

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет

вычислительной техники

Кафедра

МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

Направленность Программная инженерия

БАКАЛАВРСКАЯ РАБОТА

на тему

Программное обеспечение прохождения
психологических тестов

Студент

(подпись, дата)

Елисеев Владимир Андреевич

(ФИО полностью)

Руководитель

(подпись, дата)

Козлов А.Ю.

(фамилия, инициалы)

Нормоконтролёр

(подпись, дата)

Такташкин Д.В.

(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой

(подпись)

Козлов А.Ю.

(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК

(подпись)

Попова Н.А.

(фамилия, инициалы)

Пенза, 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
А.Ю. Козлов
«20» декабря 2024 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Елисеев Владимир Андреевич гр. 21ВП1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Программное обеспечение прохождения психологических тестов

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

3. Руководитель работы д.т.н., профессор, заведующий кафедрой «МО и ПЭВМ» Козлов А.Ю.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) *Назначение: Программное обеспечение прохождения психологических тестов*

2) *Среда выполнения: операционная система Windows 10*

Языковые средства разработки: C#, UML

Инструментальные средства: MS Visual Studio 2022, SQL Server, Draw.io,

Технологии разработки: ООП, принципы SOLID, шаблоны проектирования.

3) *Основные функции приложения:*

а) пройти психологическое тестирование;

б) просмотреть результаты психологического тестирования;

в) просмотреть интерпретацию результатов тестирования;

г) управлять учётными записями пользователей;

5. Перечень подлежащих разработке вопросов:

1) *Анализ предметной области и требований к системе;*

2) *Проектирование программных средств;*

3) *Реализация программных средств;*

4) Тестирование программных средств;

5) Оформление пояснительной записки.

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	08.04.2025 – 14.04.2025	
2	Проектирование программных средств	20%	15.04.2025 – 25.04.2025	
3	Реализация программных средств	30%	26.04.2025 – 12.05.2025	
4	Тестирование программных средств	20%	13.05.2025 – 22.05.2025	
5	Оформление пояснительной записки	20%	23.05.2025 – 03.06.2025	

Дата выдачи задания «20» декабря 2024г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 128 листов, 31 рисунок, 14 таблиц и 20 использованных источников.

ПСИХОЛОГИЧЕСКОЕ ТЕСТИРОВАНИЕ, ИНТЕРПРЕТАЦИЯ, ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ.

Объект исследования – программное обеспечение прохождения психологического тестирования.

Цель выпускной квалификационной работы – спроектировать и разработать программные средства, позволяющие проходить психологическое тестирование, получать результаты, а также интерпретировать их результаты.

Технологии разработки – разработка всех программных средств производится в среде разработки Microsoft Visual Studio 2022 на объектно-ориентированном языке программирования C#. Развертывание основного приложения осуществляется на операционных системах семейства Windows версии 10 и выше. В рамках обработки и хранения данных приложение взаимодействует с СУБД Microsoft SQL Server. В качестве среды исполнения программных средств выступает персональный компьютер пользователя.

Результаты работы – в ходе выполнения выпускной квалификационной работы были спроектированы и разработаны программные средства, позволяющие пройти психологическое тестирование и получить результаты.

					ПГУ 09.03.04– 8BP211.06 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.		Елисеев В.А.			Программное обеспечение прохождения психологических тестов.	Лит.	Лист	Листов
Пров.		Козлов А.Ю.					4	128
Н. Контр.		Такташкин Д.В			Пояснительная записка	Группа 21ВП1		
Утв.		Козлов А.Ю.						

Содержание

Введение.....	7
1 Анализ предметной области и постановка задачи на разработку программного обеспечения для прохождения психологических тестов	8
1.1 Анализ предметной области психологического тестирования.....	8
1.2 Анализ методик психологического тестирования.....	9
1.3 Анализ программного обеспечения аналогичного назначения	14
1.4 Постановка задачи на разработку программных средств.....	19
1.5 Анализ функциональных требований	20
1.6 Анализ нефункциональных требований	22
1.7 Планирование разработки и оценка бюджета.....	23
2 Проектирование программного обеспечения для прохождения психологических тестов	26
2.1 Архитектура приложения	26
2.2 Проектирование структур данных.....	28
2.2.1 Концептуальная модель базы данных	28
2.2.2 Логическая модель базы данных	29
2.3 Проектирование программных средств.....	30
2.3 Проектирование пользовательского интерфейса.....	34
3 Реализация программных средств психологического тестирования.....	36
3.1 Реализация базы данных	36
3.2 Реализация алгоритмической части обработки результатов	38
3.3 Реализация основного приложения.....	42
4 Контроль стандартов качества программных решений	54
4.1 Функциональное тестирование	54

	6
4.2 Анализ сложности кода модуля создания узлов	57
4.2.1 Вычисление и определение метрики SLOC	57
4.2.2 Вычисление и определение метрики Холстеда	59
4.3 Системное тестирование	61
4.3.1 Тестирование совместимости.....	61
Заключение	63
Список использованных источников	64
Приложение А (справочное) Глоссарий	67
Приложение Б (обязательное) Листинг приложения	69

Введение

В современном мире психологическое благополучие человека становится одним из ключевых факторов его успешной профессиональной деятельности и адаптации в социуме. Особую значимость приобретают программные средства, позволяющие оперативно и эффективно проводить психологическое тестирование различных категорий людей, включая обучающихся в образовательных организациях [1]. Такие инструменты становятся незаменимыми помощниками в работе профессиональных психологов, позволяя автоматизировать процесс сбора и анализа результатов психологических тестов.

Разрабатываемое программное обеспечение для прохождения психологических тестов представляет собой комплексный инструмент, реализованный на языке программирования C# с использованием технологии Windows Forms и базы данных SQL Server. Система предназначена для проведения психологической диагностики и определения личностных характеристик пользователей через специализированные тесты, включающие определение доминирующих инстинктов, геометрических психотипов и направленности личности.

Целью выпускной квалификационной работы является разработка надежного и удобного программного обеспечения для проведения психологического тестирования, обеспечивающего валидность и конфиденциальность данных, простоту использования и эффективность обработки результатов. Разработка включает в себя анализ требований, проектирование, реализацию и всестороннее тестирование программного обеспечения с учетом современных стандартов качества.

Программное решение должно обеспечивать прохождение тестов пользователями, обработку результатов, а также предоставлять администратору (профессиональному психологу) инструменты для анализа данных и выработки рекомендаций.

1 Анализ предметной области и постановка задачи на разработку программного обеспечения для прохождения психологических тестов

1.1 Анализ предметной области психологического тестирования

С развитием информационных технологий в области психологической диагностики значительно выросла необходимость автоматизации процессов проведения и обработки результатов психологических тестов [2]. На текущий момент использование специализированного программного обеспечения для психологического тестирования становится неотъемлемой частью работы психологических служб образовательных организаций и других учреждений. Конечное использование подобных технологий сводится к упрощенному пользовательскому интерфейсу с доступным набором психологических методик, удобной системой регистрации пользователей и наглядного отображения результатов тестирования.

Развитие современных программных платформ позволяет создавать надежные локальные системы тестирования без необходимости передачи конфиденциальных психологических данных на удаленные серверы, что особенно важно при работе с персональными данными в образовательных учреждениях. Однако разработка специализированного программного обеспечения для психологического тестирования должна учитывать особенности методик, требования к валидности и надежности получаемых результатов, а также обеспечивать защиту конфиденциальных данных пользователей.

В рамках психологической диагностики основное предназначение автоматизированных систем тестирования заключается в точном и стандартизированном сборе данных о личностных особенностях тестируемых, эффективной обработке результатов и предоставлении профессиональному психологу инструментов для квалифицированной интерпретации. Решение данной задачи требует разработки интуитивно понятного интерфейса как для

тестируемых, так и для администраторов системы, а также создания надежной базы данных для хранения результатов тестирования.

В современных образовательных организациях существует потребность в системах, позволяющих проводить массовое психологическое тестирование с целью выявления индивидуальных особенностей обучающихся и формирования персонализированного подхода к их обучению и развитию. Особенно важным становится определение доминирующих инстинктов, психологических типов и направленности личности, что позволяет преподавателям и курсовым офицерам адаптировать педагогические методы к индивидуальным особенностям учащихся.

Наиболее эффективным подходом для разработки такой системы является создание специализированного программного обеспечения на базе языка C# с использованием Windows Forms для интерфейса и SQL Server для хранения данных. Такое решение обеспечивает необходимую гибкость, надежность и функциональность при относительно невысоких затратах на разработку и внедрение.

1.2 Анализ методик психологического тестирования

В рамках разрабатываемого программного обеспечения для психологического тестирования используется методика выявления доминирующего инстинкта, разработанная В. Гарбузовым, которая является одной из научно обоснованных методик определения базовых типологических особенностей личности [3]. Данная методика предназначена для определения ведущих инстинктов, лежащих в основе типологии индивидуальности. Вилен Гарбузов выделяет семь основных инстинктов, которые формируют типологическую структуру личности.

Эгофильный тип характеризуется преобладанием инстинкта самосохранения. С раннего детства у человека этого типа проявляется склонность к повышенной осторожности. Такие люди демонстрируют нетерпимость к боли, эгоцентричность, тревожную мнительность, склонность

при неблагоприятных обстоятельствах к навязчивым страхам, фобиям или истерическим реакциям.

Ведущие качества «эгофилов»:

- эгоцентричность;
- консерватизм;
- готовность поступиться социальными потребностями ради собственной безопасности;
- отрицание риска;
- тревожность в отношении своего здоровья и благополучия.

Кредо этого типа: «Безопасность и здоровье – превыше всего!», «Жизнь одна и больше не будет». Эволюционная целесообразность наличия данного типа заключается в том, что его носители, сохраняя себя, являются хранителями генофонда рода, племени.

Генофильный тип характеризуется своеобразной разновидностью эгоцентризма, когда «Я» замещается понятием «Мы» (под «Мы» подразумевается семья) вплоть до отрицания «Я». Ценности, цели, жизненный замысел подчинены интересам детей и семьи.

Ведущие качества «генофилов»:

- сверхлюбовь к своим детям;
- семейственность;
- сверхзабота о безопасности и здоровье своих детей;
- тенденция к отрицанию своего «Я» в пользу «Мы» (семьи);
- сверхтревожность относительно будущего своих детей.

Кредо: «Интересы семьи превыше всего», «Мой дом – моя крепость». Эволюционная целесообразность данного типа заключается в том, что его носители являются хранителями семьи и генофонда своего рода.

Альтруистический тип характеризуется добротой, эмпатией, заботливостью к близким, особенно к пожилым. Люди этого типа способны отдать другим последнее, даже необходимое им самим.

Ведущие качества «Альтруистов»:

- доброта;
- сопереживание, понимание людей;
- бескорыстность в отношениях с людьми;
- забота о слабых, больных;
- миролюбие.

Кредо: «Доброта спасет мир, доброта превыше всего». Эволюционная роль такого типа заключается в сохранении доброты, мира, жизни.

У людей исследовательского типа с раннего детства отмечается любознательность, стремление во всем добраться до сути, склонность к творчеству. Путешественники, изобретатели, ученые – лица этого типа.

Ведущие качества «исследователей»:

- склонность к исследовательской деятельности;
- склонность к поиску нового, новаторство в науке, искусстве и т. д.;
- способность без колебаний оставлять обжитое место, налаженное дело при появлении новых, требующих риска, но интересных дел и задач;
- устремленность к творчеству;
- самоотверженность в реализации творческих устремлений.

Кредо: «Творчество и прогресс – превыше всего». Эволюционная целесообразность этого типа очевидна – они двигают прогресс человечества.

С раннего детства у доминантного типа наблюдается стремление к лидерству, умение организовать деятельность, поставить цель, проявить волю для ее достижения. Формируется личность, знающая, чего она хочет и как достигнуть желаемого, упорная в достижении цели, готовая к продуманному риску, умеющая разбираться в людях и вести их за собой.

Ведущие качества «доминантов»:

- склонность к лидерству, к власти;
- предрасположенность к решению сложных организационных задач;
- приоритет перспектив служебного роста над материальными стимулами;

- готовность к жесткой борьбе за лидерство, за первое место;
- приоритет общего (интересов дела, коллектива) над частным (интересами одного человека).

Кредо: «Дело и порядок превыше всего», «Один – ничто, все – все», «Будет хорошо всем – будет хорошо каждому». Эволюционная целесообразность наличия этого типа заключается в формировании лидеров, организаторов, политиков – хранителей интересов и чести всего рода.

Либертофильный тип характеризуется склонностью к протесту против любого ограничения свободы. Для людей этого типа характерны стремление к самостоятельности, отрицание авторитетов, терпимость к боли, склонность рано покидать отчий дом, предрасположенность к риску, упрямству, негативизму, нетерпимость рутины и бюрократизма.

Ведущие качества «либертофилов»:

- склонность к протесту, бунтарству;
- предрасположенность к перемене мест (отрицание будничности);
- стремление к независимости;
- склонность к реформаторству, революционным преобразованиям;
- нетерпимость к любым формам ограничений, к цензуре, к подавлению «Я».

Кредо: «Свобода превыше всего». Эволюционная роль – хранители интересов и свободы каждого, индивидуальности, естественные ограничители тенденций лиц «Доминантного типа».

В раннем детстве человек дигнилофильного типа способен уловить иронию, насмешку и абсолютно нетерпим к любой форме унижения. Характерна безоглядность, готовность поступиться всем в отстаивании своих прав, непоколебимая позиция «Честь превыше всего».

Ведущие качества «дигнилофилов»:

- нетерпимость к любым формам унижения;
- готовность поступиться благополучием и социальным статусом во имя собственного достоинства;

- приоритет чести и гордости над безопасностью;
- бескомпромиссность и прямота в отношениях с лидерами;
- нетерпимость ко всем формам ущемления прав человека.

Кредо: «Честь превыше всего». Эволюционная целесообразность наличия такого типа заключается в том, что его носители – хранители чести и достоинства личности, а с этим – жизни, достойной человека.

Опросник В. Гарбузова состоит из 15 вопросов, примерное время тестирования составляет 3 – 5 минут. Вопросы направлены на выявление преобладающих инстинктов, формирующих типологическую структуру личности [7].

При обработке результатов подсчитывается сумма баллов по каждому из семи инстинктов. Наибольшая сумма баллов указывает на доминирующий инстинкт и на типовую принадлежность. Если равное количество баллов оказывается у двух инстинктов, то тестируемому свойственен смешанный тип. Особенно информативным считается отсутствие баллов по инстинктам альтруизма, самосохранения, продолжения рода.

Результаты тестирования представляются в виде диаграммы и текстовой интерпретации. Диаграмма показывает выраженность каждого из семи инстинктов, а в текстовой интерпретации указывается доминирующий инстинкт с его основными характеристиками и ведущими качествами личности.

Помимо методики В. Гарбузова, в разрабатываемом программном обеспечении также должен быть реализован тест на определение геометрического психотипа личности, который дополняет общую картину психологической диагностики. Этот тест основан на выборе геометрических фигур и определяет пять основных типов личности: Треугольник (лидерство), Квадрат (трудолюбие), Круг (коммуникабельность), Зигзаг (творчество) и Прямоугольник (переходное состояние).

Также необходимо реализовать тест на направленность личности, который будет показывать основную направленность личности тестируемого среди нескольких вариантов: на себя, на общество и на дело.

Реализация методики В. Гарбузова, геометрического и теста на направленность личности в разрабатываемом программном обеспечении предполагает:

- 1 – создание системы ввода данных пользователя для регистрации;
- 2 – разработку интерфейса для представления вопросов теста;
- 3 – реализацию алгоритмов обработки ответов и подсчета результатов;
- 4 – создание системы визуализации результатов в виде диаграмм;
- 5 – формирование текстовых интерпретаций результатов;
- 6 – разработку базы данных для хранения результатов тестирования;
- 7 – обеспечение доступа администратора к результатам тестирования для последующего анализа и консультирования.

Методика В. Гарбузова широко применяется в психологической практике и показала свою эффективность в выявлении базовых типологических особенностей личности. Типология инстинктов обладает прогностической ценностью и позволяет не только понять текущие особенности личности, но и предсказать поведение в различных ситуациях.

Геометрический тест также является проверенным инструментом психологической диагностики, который дополняет картину, полученную с помощью методики В. Гарбузова, и позволяет сформировать более полное представление о личностных особенностях тестируемого.

С помощью тестирования на направленность личности можно будет получить еще более полную картину о тестируемом, в том числе получить информацию о его поведении в обществе.

Интеграция этих методик в программное обеспечение позволяет автоматизировать процесс тестирования, сделать его более эффективным и доступным, а также обеспечить надежное хранение и анализ результатов.

1.3 Анализ программного обеспечения аналогичного назначения

В настоящее время на рынке существует несколько программных продуктов для психологического тестирования. Для анализа выбраны два

наиболее популярных решения: Psychometric Expert и Effecton Studio, которые имеют схожую с разрабатываемым ПО функциональность, но отличаются подходами к реализации и целевой аудиторией.

Psychometric Expert представляет собой многофункциональную компьютерную диагностическую систему, разработанную для психологов, HR-менеджеров и специалистов по подбору персонала. Система позволяет проводить психологическую диагностику личности и группы, вести учет, осуществлять психологический отбор, оценку и сопровождение персонала организации [4]. Графический интерфейс приложения «Psychometric Expert» представлен на рисунке 1.

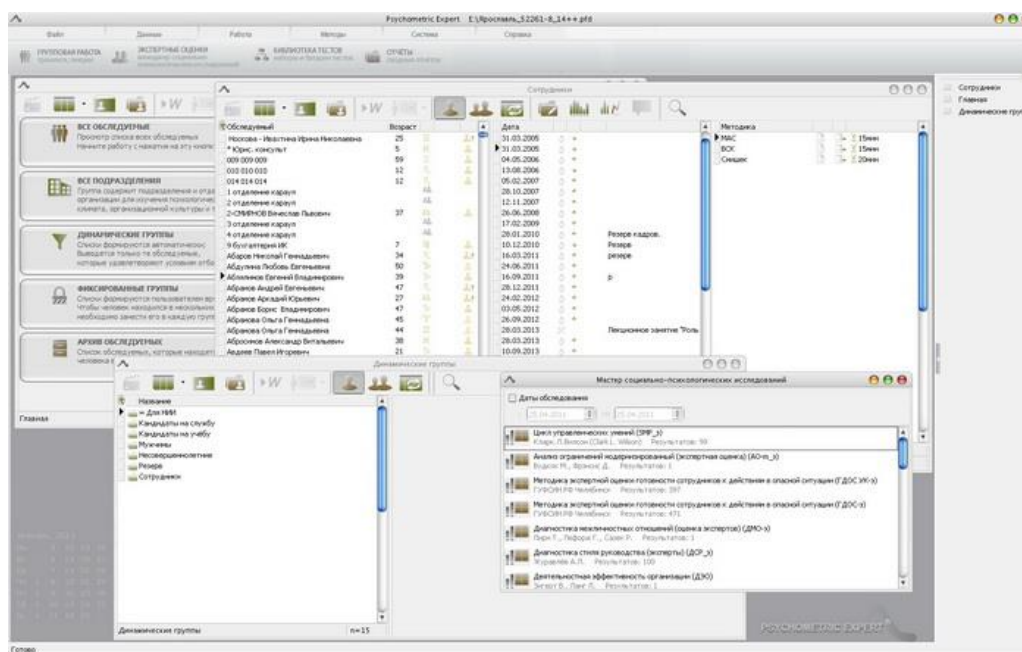


Рисунок 1 – Графический интерфейс приложения «Psychometric Expert»

Основные возможности:

- библиотека из более чем 600 психодиагностических методик;
- электронный учебник по экспериментальной психологии и психодиагностике;
- статистический анализ данных;
- среда управления компетенциями;
- разработка интерпретаторов, моделей, профессиограмм;
- система контроля исполнения обязательств.

Преимущества:

- 1 – обширная библиотека тестов – содержит более 600 различных методик, что позволяет проводить комплексную психологическую диагностику;
- 2 – модульная структура – возможность расширения функциональности через дополнительные модули;
- 3 – открытая система – пользователи могут самостоятельно разрабатывать и распространять специализированные компоненты.

Недостатки:

- 1 – высокая сложность освоения – требуется значительное время для изучения всех возможностей системы;
- 2 – избыточность функциональности для базовых задач психологического тестирования;
- 3 – высокая стоимость полной версии программы.

Effecton Studio – программный комплекс для автоматизации психологических служб образовательных и социальных учреждений. Содержит более 150 психологических тестов и упражнений для развития различных когнитивных функций и диагностики личности [5]. Графический интерфейс приложения «Effecton Studio» представлен на рисунке 2.

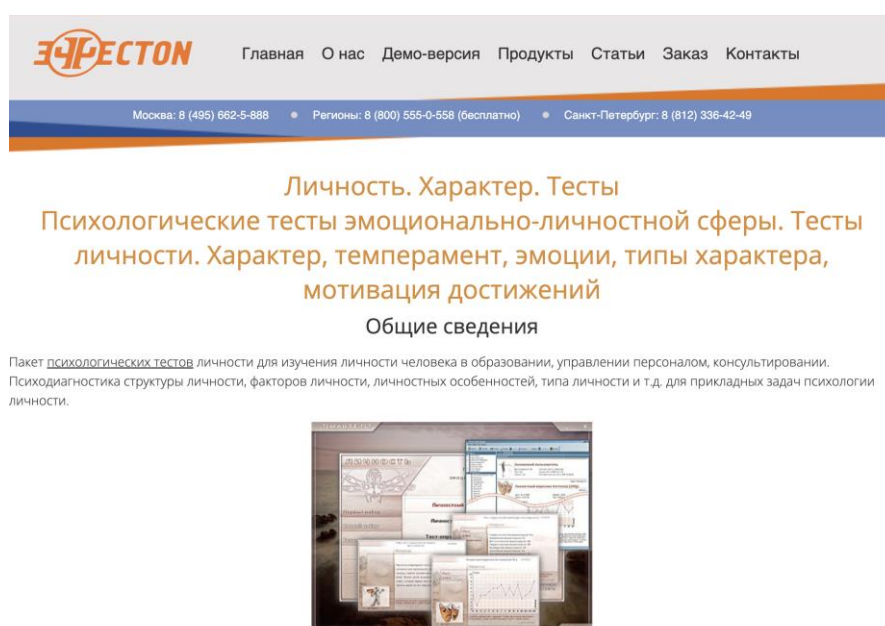


Рисунок 2 – Графический интерфейс приложения «Effecton Studio»

Основные возможности:

- комплекс психологических тестов и методик;
- упражнения для развития внимания, памяти, интеллекта, восприятия, моторики;
- специализированные версии для различных образовательных учреждений;
- анализ результатов психологического тестирования;
- групповая статистика.

Преимущества:

- 1 – специализация на образовательных учреждениях – хорошо адаптирован для школ, вузов, дошкольных учреждений;
- 2 – наличие развивающих упражнений – комплекс предлагает не только диагностику, но и средства развития психических функций;
- 3 – доступная демоверсия – возможность протестировать функциональность перед покупкой;
- 4 – интуитивно понятный интерфейс – требует меньше времени на освоение;
- 5 – модификации для различных целей – специальные версии для школы, вуза, дошкольного образования, социальной работы и управления персоналом.

Недостатки:

- 1 – меньшее количество методик по сравнению с Psychometric Expert;
- 2 – ограниченные возможности для кастомизации – пользователи не могут так свободно разрабатывать собственные методики;
- 3 – недостаточная гибкость настройки для специфических задач;
- 4 – отсутствие глубокого статистического анализа данных;
- 5 – невозможность создания собственных интерпретаций результатов тестирования.

Анализ аналогов показал, что существующие решения для психологического тестирования можно разделить на две основные категории:

1 – профессиональные комплексные системы (Psychometric Expert) – функционально богатые, но сложные в освоении и дорогостоящие;

2 – специализированные решения для образовательных учреждений (Effecton Studio) – более доступные, но с ограниченными возможностями настройки.

Сравнительный анализ приложений аналогов представлен в таблице 1.

Таблица 1 – Сравнительный анализ аналогов с разрабатываемым программным обеспечением

Критерий	Psychometric Expert	Effecton Studio	Разрабатываемое ПО
Целевая аудитория	HR-менеджеры, психологи кадровых служб	Психологи образовательных учреждений	Психологи образовательных организаций
Количество методик	>600	~150	Ограниченный набор с фокусом на методику В. Гарбузова
Возможность доработки	Высокая	Средняя	Средняя
Стоимость	Высокая	Средняя	Планируется низкая
Простота освоения	Сложная	Средняя	Простая
Статистический анализ	Расширенный	Базовый	Базовый
Ориентация на конкретные методики	Широкий спектр	Широкий спектр	Специализированные методики
Интерфейс	Сложный, профессиональный	Современный, интуитивный	Простой, минималистичный
Требования к системе	Высокие	Средние	Низкие
Модульность	Высокая	Средняя	Низкая

Разрабатываемое программное обеспечение должно объединить сильные стороны обоих типов решений:

- простой и интуитивно понятный интерфейс как в Effecton Studio;
- фокус на конкретные психологические методики;
- минимальные требования к системе и простота установки;
- доступная стоимость;
- ориентация на специфические потребности образовательных организаций.

При этом необходимо учесть недостатки аналогов и избежать:

- чрезмерного усложнения функциональности;
- устаревшего интерфейса;
- избыточности данных;
- высоких требований к аппаратному обеспечению;
- сложности настройки и использования.

Разрабатываемое программное обеспечение будет отличаться от существующих решений большей простотой в использовании, фокусом на конкретных психологических методиках и оптимизацией для образовательных организаций с учетом их специфических потребностей.

1.4 Постановка задачи на разработку программных средств

Целью разработки является создание программного обеспечения для прохождения психологических тестов, ориентированного на образовательные организации и эффективно решающего задачи психологической диагностики обучающихся. Необходимость разработки обусловлена потребностью автоматизации процесса психологического тестирования с использованием методики В. Гарбузова, геометрического и теста на направленность личности для определения психотипа личности.

Основной задачей разрабатываемого программного обеспечения является создание удобного инструмента для психологов-администраторов и пользователей, позволяющего проводить тестирование, сохранять результаты и формировать отчеты. Система должна обеспечивать разграничение прав доступа между ролями администратора (психолога) и пользователей (тестируемых), для чего необходима надежная аутентификация пользователей с валидацией вводимых данных.

В рамках разработки программного обеспечения требуется реализовать логику прохождения психологических тестов, алгоритмы анализа и интерпретации результатов, а также создать эргономичный интерфейс для взаимодействия с системой. Важным аспектом является организация хранения

данных с использованием SQL Server без привлечения внешних библиотек, что обеспечит независимость и устойчивость работы приложения.

1.5 Анализ функциональных требований

Диаграмма вариантов использования используется для визуализации функциональных требований к системе или программному продукту [6]. Она позволяет понять, как пользователи будут взаимодействовать с системой, какие действия они будут выполнять, какие варианты использования системы существуют, а также кто участвует в этих вариантах использования. Диаграмма вариантов использования помогает лучше понять потребности пользователей и определить функциональные требования к системе. Разработанная диаграмма вариантов использования для программного обеспечения психологического тестирования приведена на рисунке 3.

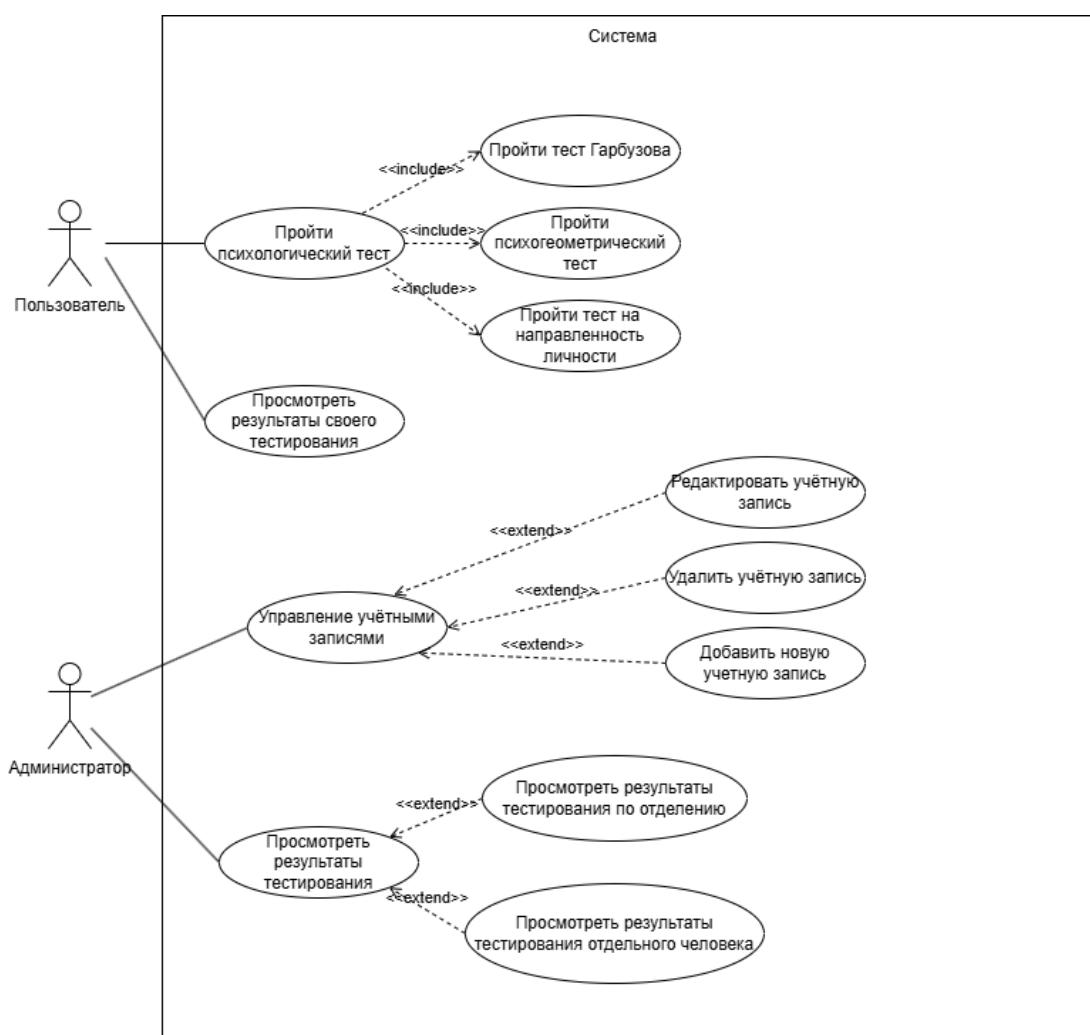


Рисунок 3 – Диаграмма вариантов использования приложения

На диаграмме представлены два основных актора системы: Пользователь (тестируемый) и Администратор (психолог). Каждый из них взаимодействует с системой посредством определенного набора функций.

Спецификация варианта использования «Пройти психологический тест» приведена на рисунке 4.

Прецедент: Пройти психологический тест
Краткое описание: Пользователь проходит выбранный психологический тест через интерфейс программы
Актеры: Пользователь (тестируемый)
Предусловия: Пользователь авторизован в системе
Основной поток: <ol style="list-style-type: none"> 1. Пользователь выбирает тест из списка доступных. 2. Система отображает инструкции к тесту. 3. Пользователь отвечает на вопросы теста. 4. Система обрабатывает ответы и отображает результаты тестирования. 5. Система сохраняет результаты в базе данных.
Постусловия: Результаты тестирования сохранены в системе и доступны для просмотра

Рисунок 4 – Спецификация варианта использования «Пройти психологический тест»

Спецификация варианта использования «Просмотреть результаты тестирования» приведена на рисунке 5.

Прецедент: Просмотреть результаты тестирования
Краткое описание: Администратор просматривает результаты пройденных пользователями тестов
Актеры: Администратор (психолог)
Предусловия: Администратор авторизован в системе
Основной поток: <ol style="list-style-type: none"> 1. Администратор выбирает раздел «Результаты тестирования». 2. Система отображает список пользователей, прошедших тестирование. 3. Администратор выбирает конкретного пользователя или группу пользователей. 4. Система отображает подробную информацию о результатах тестирования. 5. Администратор может экспортировать результаты в формате отчета.
Постусловия: Администратор получает доступ к результатам тестирования пользователей

Рисунок 5 – Спецификация варианта использования «Просмотреть результаты тестирования»

Особенным вариантом использования является «Редактировать данные пользователя». Администратор любой системы должен иметь доступ к данным и управлять доступом пользователей к системе. В реализуемой программе администратор может удалить или редактировать данные созданных пользователей, а также создавать аккаунты пользователей.

Спецификация варианта использования «Редактировать данные пользователя» приведена на рисунке 6.

Прецедент: Редактировать данные пользователя
Краткое описание: Администратор редактирует данные пользователя в системе
Актеры: Администратор (психолог)
Предусловия: Администратор авторизован в системе
Основной поток: <ol style="list-style-type: none"> 1. Администратор выбирает раздел «Управление пользователями». 2. Система отображает список существующих пользователей. 3. Администратор нажимает кнопку «Редактировать данные» возле ФИО пользователя. 4. Администратор вводит новые данные о пользователе и нажимает кнопку «Сохранить». 5. Система сохраняет изменённые данные в базе данных.
Постусловия: Администратор отредактировал данные пользователя.

Рисунок 6 – Спецификация варианта использования «Редактировать данные пользователя»

Таким образом, проанализированные функциональные возможности приложения определяют основные способы взаимодействия с пользователями. Это позволяет выделить бизнес-процессы приложения и сформировать минимальный набор реализуемых функций для корректной работы программного обеспечения психологического тестирования.

1.6 Анализ нефункциональных требований

Разработка программного обеспечения для прохождения психологических тестов должна быть выполнена в многофункциональной среде разработки Microsoft Visual Studio 2022 на объектно-ориентированном языке программирования C#. Графический интерфейс должен быть построен с использованием технологии Windows Forms. В качестве СУБД должна

выступать Microsoft SQL Server. Приложение должно разворачиваться и исполняться на операционных системах семейства Windows версии 10 и выше.

1.7 Планирование разработки и оценка бюджета

Для оценки временных затрат, а также наглядного планирования процесса разработки была построена диаграмма Ганта с использованием инструментов Microsoft Office Project 2007. Общий процесс разработки приложения содержит пять вех. Каждый этап включает в себя определенный набор подзадач, которые необходимо выполнить для перехода на новый этап [7]. Для разработки данного приложения целесообразно выделить следующие этапы:

1 – анализ предметной области и постановка задачи на разработку программного обеспечения для прохождения психологических тестов;

2 – проектирование программного обеспечения для прохождения психологических тестов;

3 – реализация программного обеспечения для прохождения психологических тестов;

4 – контроль качества программного обеспечения;

5 – оформление отчета.

Краткое описание задач каждого этапа приведено на рисунке 7.

Название задачи	Длительность	Начало	Окончание
Начало проекта	0 дней	Пн 07.04.25	Пн 07.04.25
▣ Анализ предметной области и постановка задачи на разработку про	5 дней?	Пн 07.04.25	Пт 11.04.25
Анализ предметной области	0,5 дней?	Пн 07.04.25	Пн 07.04.25
Анализ методик психологического тестирования	0,5 дней?	Пн 07.04.25	Пн 07.04.25
Анализ программного обеспечения аналогичного назначения	1 день?	Вт 08.04.25	Вт 08.04.25
Постановка задачи на разработку программных средств	1 день?	Ср 09.04.25	Ср 09.04.25
Анализ функциональных требований	0,5 дней	Чт 10.04.25	Чт 10.04.25
Анализ нефункциональных требований	0,5 дней?	Чт 10.04.25	Чт 10.04.25
Планирование разработки и оценка бюджета	1 день?	Пт 11.04.25	Пт 11.04.25
▣ Проектирование программного обеспечения для прохождения психс	9 дней	Пн 14.04.25	Чт 24.04.25
Архитектура приложения	3 дней	Пн 14.04.25	Ср 16.04.25
Проектирование структур данных	2 дней	Чт 17.04.25	Пт 18.04.25
Проектирование программных средств	2 дней	Пн 21.04.25	Вт 22.04.25
Проектирование пользовательского интерфейса	2 дней	Ср 23.04.25	Чт 24.04.25
▣ Реализация программных средств психологического тестирования	11 дней	Пт 25.04.25	Пт 09.05.25
Реализация базы данных	2 дней	Пт 25.04.25	Пн 28.04.25
Реализация алгоритмической части приложения	2 дней	Вт 29.04.25	Ср 30.04.25
Реализация основного приложения	7 дней	Чт 01.05.25	Пт 09.05.25
▣ Контроль стандартов качества программных решений	8 дней?	Пн 12.05.25	Ср 21.05.25
Функциональное тестирование приложения	3 дней	Пн 12.05.25	Ср 14.05.25
Анализ сложности кода приложения	4 дней	Чт 15.05.25	Вт 20.05.25
Системное тестирование приложения	1 день?	Ср 21.05.25	Ср 21.05.25
▣ Оформление отчёта	8 дней	Чт 22.05.25	Пн 02.06.25
Оформление пояснительной записки	8 дней	Чт 22.05.25	Пн 02.06.25
Конец проекта	0 дней	Пн 02.06.25	Пн 02.06.25

Рисунок 7 – Сведения об этапах разработки приложения

На первом этапе содержатся задачи, направленные на проведение анализа предметной области психологического тестирования, изучение методик В. Гарбузова и геометрического теста, анализ аналогичного программного обеспечения и формирование исходных требований к разработке. Второй этап предполагает проектирование программного обеспечения, разработку архитектуры приложения, проектирование базы данных и пользовательского интерфейса. На третьем этапе предполагается конечная реализация разрабатываемого программного обеспечения, включая создание базы данных, реализацию алгоритмической части обработки результатов тестирования и основного приложения. Четвертый этап предполагает функциональное тестирование средствами причинно-следственного графа и попарного тестирования, анализ сложности кода, а также системное тестирование и проверку совместимости с различными операционными системами и устройствами. Пятый этап является заключительным и предполагает оформление отчёта по выполненной работе.

Для наглядного отражения временных характеристик разработки программного обеспечения была построена диаграмма Ганта. Данная диаграмма приведена на рисунке 8.

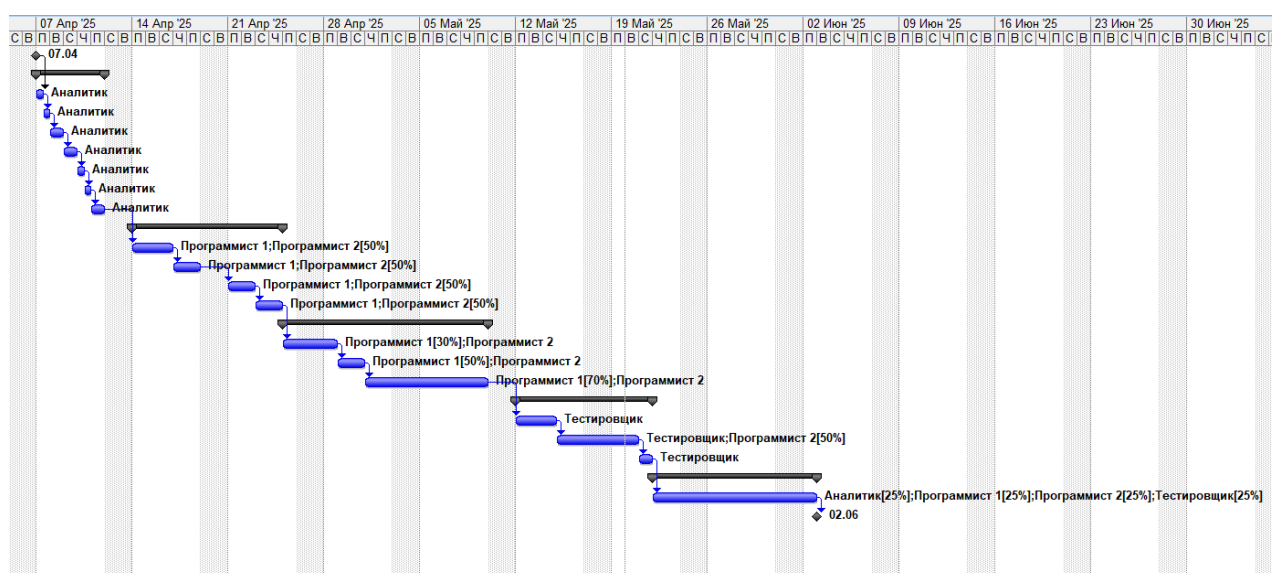


Рисунок 8 – Этапы разработки приложения на диаграмме Ганта

Для оценки финансовых затрат необходимо назначить цены ресурсов. Стоимость используемых ресурсов представлена на рисунке 9.

	1	Название ресурса	Тип	Единицы измерения материалов	Краткое название	Группа	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Затраты на испол.	Начисление	Базовый календарь
1		Программист 1	Трудовой		П		100%	350,00 Р/ч	0,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
2		Программист 2	Трудовой		П		100%	200,00 Р/ч	0,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
3		Аналитик	Трудовой		А		100%	200,00 Р/ч	0,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
4		Тестировщик	Трудовой		Т		100%	170,00 Р/ч	0,00 Р/ч	0,00 Р	Пропорциональное	Стандартный

Рисунок 9 – Стоимость задействованных ресурсов

Общая оценка финансовых затрат проекта представлена на рисунке 10.

Название задачи	Затраты
Начало проекта	0,00 Р
Анализ предметной области и постановка задачи на разработку про	8 000,00 Р
Анализ предметной области	800,00 Р
Анализ методик психологического тестирования	800,00 Р
Анализ программного обеспечения аналогичного назначения	1 600,00 Р
Постановка задачи на разработку программных средств	1 600,00 Р
Анализ функциональных требований	800,00 Р
Анализ нефункциональных требований	800,00 Р
Планирование разработки и оценка бюджета	1 600,00 Р
Проектирование программного обеспечения для прохождения психс	32 400,00 Р
Архитектура приложения	10 800,00 Р
Проектирование структур данных	7 200,00 Р
Проектирование программных средств	7 200,00 Р
Проектирование пользовательского интерфейса	7 200,00 Р
Реализация программных средств психологического тестирования	35 800,00 Р
Реализация базы данных	4 880,00 Р
Реализация алгоритмической части приложения	6 000,00 Р
Реализация основного приложения	24 920,00 Р
Контроль стандартов качества программных решений	14 080,00 Р
Функциональное тестирование приложения	4 080,00 Р
Анализ сложности кода приложения	8 640,00 Р
Системное тестирование приложения	1 360,00 Р
Оформление отчёта	14 720,00 Р
Оформление пояснительной записки	14 720,00 Р
Конец проекта	0,00 Р
	Итого: 105000 руб.

Рисунок 10 – Общая оценка финансовых затрат

Таким образом, исходя из оценки финансовых и временных затрат, видно, что предполагаемые затраты на разработку составляют 105 000 рублей, а общая продолжительность – 41 день. Следовательно, мы можем сделать вывод, что проект может быть успешно реализован в довольно сжатые сроки при указанных затратах.

2 Проектирование программного обеспечения для прохождения психологических тестов

2.1 Архитектура приложения

Архитектура разрабатываемого программного обеспечения для психологического тестирования построена на основе многоуровневой структуры, обеспечивающей четкое разделение ответственности между компонентами системы. Приложение спроектировано с учетом разграничения прав доступа между двумя основными типами пользователей: тестируемыми и администраторами (профессиональными психологами). Основа архитектуры состоит из трёх уровней.

Уровень представления является набором форм Windows Forms, реализующих графический интерфейс приложения. Основные компоненты этого уровня:

- MainForm – главное окно приложения, обеспечивающее навигацию между функциональными разделами;
- LoginForm – форма авторизации, содержащая поля для ввода персональных данных пользователя;
- TestForm – форма для прохождения психологических тестов с отображением вопросов и вариантов ответов;
- ResultForm – форма отображения результатов тестирования с диаграммами и текстовыми интерпретациями;
- AdminPanelForm – специальная форма для административного доступа с расширенной функциональностью.

На уровне представления также реализован ScalingModule, обеспечивающий корректное масштабирование интерфейса под различные размеры экранов и разрешения, что важно для обеспечения совместимости с разнообразными устройствами.

Уровень бизнес-логики содержит классы, реализующие основные алгоритмы и логику работы приложения:

- TestManager – класс, отвечающий за управление процессом тестирования, включая выбор теста, отображение вопросов и сбор ответов;
- ResultProcessor – класс, реализующий алгоритмы обработки результатов тестирования по методикам В. Гарбузова, направленностью личности и геометрического теста;
- UserManager – класс для управления данными пользователей и их авторизации в системе;
- AdminManager – класс, обеспечивающий функциональность административного интерфейса.

На уровне доступа к данным база данных SQL Server структурирована в соответствии с разработанной физической моделью и включать таблицы Users, Administrators, Results и Department. Структура оптимизирована для обеспечения целостности данных и эффективного доступа к информации.

Процесс взаимодействия пользователя с системой выстроен следующим образом:

- 1 – при запуске приложения пользователь видит форму авторизации, где вводит свои персональные данные (фамилия, имя, отчество, отделение);
- 2 – после валидации введенных данных создается пользовательская сессия и пользователь перенаправляется в главное меню;
- 3 – в главном меню пользователь может выбрать один из доступных тестов для прохождения;
- 4 – TestManager отображает последовательность вопросов на TestForm и обрабатывает ответы пользователя;
- 5 – по завершении тестирования ResultProcessor анализирует ответы пользователя, формирует интерпретацию результатов и сохраняет их в базе данных.

Администратор (психолог) имеет доступ к расширенному функционалу через специальный режим входа. В административном интерфейсе реализованы функции просмотра результатов всех пользователей, управления учетными записями, формирования статистических отчетов и экспорта/импорта данных.

2.2 Проектирование структур данных

2.2.1 Концептуальная модель базы данных

Концептуальная модель данных представляет собой наглядное отражение организации данных на высоком уровне. Данное представление может демонстрировать описание сущностей, их атрибуты и отношения без указания технических деталей реализации [8]. Она служит основой для дальнейшего перехода к логическому и физическому уровням моделирования, облегчая коммуникацию между разработчиками и заказчиками. На этой стадии также можно выявить потенциальные несоответствия требований и своевременно скорректировать структуру данных.

Согласно выделенным требованиям к приложению по хранению и организации данных была разработана диаграмма типа «сущность-связь». Данная диаграмма представлена на рисунке 11.

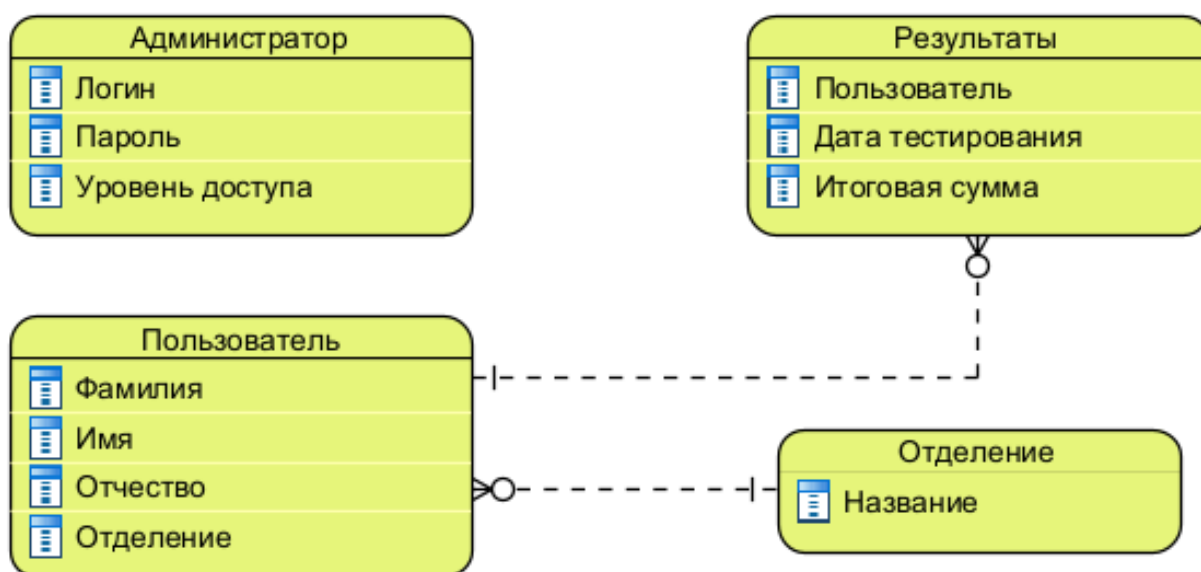


Рисунок 11 – Графическое представление концептуальной модели

Разработанная диаграмма наглядно отражает хранимые элементы приложения и их связи. В таблице 2 представлены сущности, сформированные согласно входным и выходным данным бизнес-процессов.

Таблица 2 – Описание сущностей

Наименование сущности	Идентификатор сущности	Назначение сущности
Пользователь	User	Содержит данные о пользователях системы, их личную информацию и учетные данные
Администратор	Administrator	Содержит данные об администраторах системы (профессиональных психологах), их учетную информацию и права доступа
Результат	Result	Содержит обработанные результаты тестирования, интерпретации и рекомендации
Отделение	Department	Содержит информацию о названии отделения

В таблице 3 представлено описание связей между сущностями.

Таблица 3 – Описание связей

Наименование связи	Описание	Мощность
Department-User	Определяет принадлежность пользователя к отделению	1:M
User-Result	Определяет принадлежность результатов тестирования к пользователям	1:M

Концептуальная модель данных построена в графической среде Visual Paradigm 17.0 в нотации IDEF1X [9].

2.2.2 Логическая модель базы данных

Логическая модель данных является расширением концептуальной модели, в которой сущности, атрибуты и связи дополнительно детализированы и организованы. Логическая модель данных является основой физической модели, и раскрывает представление данных более детально с технической точки зрения [10]. На этом уровне можно выявить потенциальные оптимизации структуры данных ещё до начала реализации СУБД. Логическая модель данных позволяет определить типы атрибутов, а также ключевые поля для каждой сущности. В данной модели уточняются связи между сущностями, включая типы связей и правила целостности. Также происходит нормализация данных для устранения избыточности и повышения эффективности хранения. Логическая модель не зависит от конкретной СУБД, но учитывает технические требования к реализации. Таким образом, она играет ключевую роль в переходе от абстрактной структуры к реализуемому проекту базы данных.

На основе концептуальной модели данных была построена логическая модель, представленная на рисунке 12.

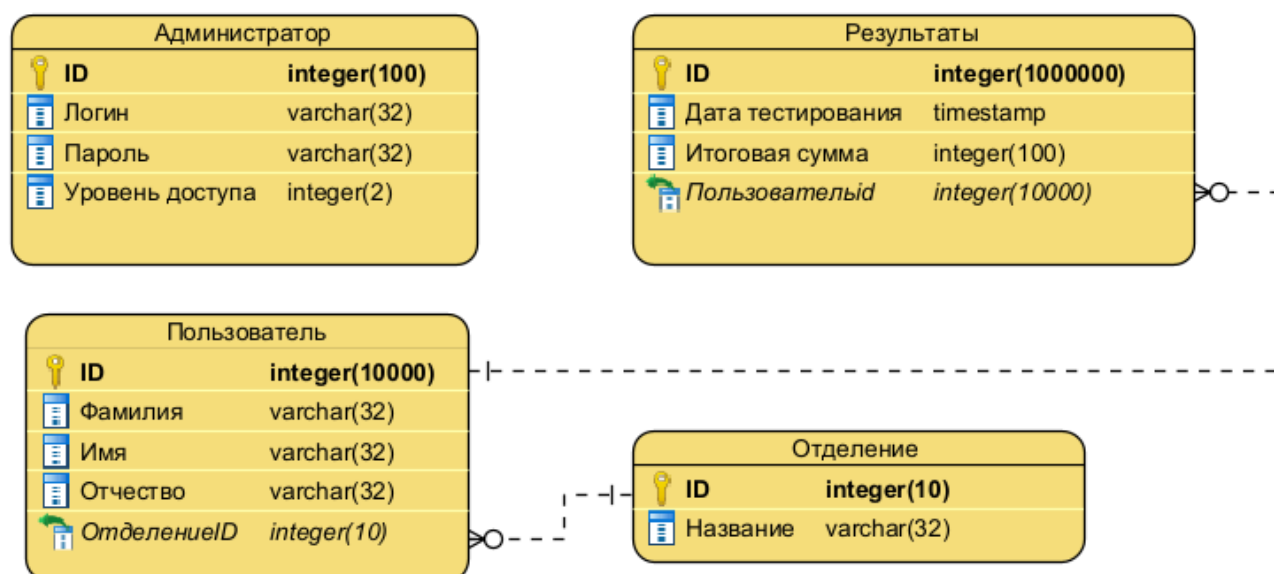


Рисунок 12 – Графическое представление логической модели

Логическая модель была построена в среде Visual Paradigm 17.0 в нотации моделирования Crow's Foot [11].

Таким образом, спроектированные графические представления организации и хранения данных позволяют выполнить техническую реализацию обращения к базе данных приложения, а также наглядно отразить структурные связи между данными.

2.3 Проектирование программных средств

Диаграмма классов описывает структуру системы через классы, их свойства, методы и взаимосвязи. Способствует формализации архитектурных решений и облегчает процесс командной разработки.

Также диаграмма помогает выявить дублирующие элементы и определить зоны ответственности компонентов. С помощью визуализации упрощается переход от проектирования к программной реализации. На основе архитектуры приложения была спроектирована диаграмма классов, приведенная на рисунке 13. Данная диаграмма позволяет наглядно отобразить основные технические решения, которые будут применяться в рамках дальнейшей разработки приложения.

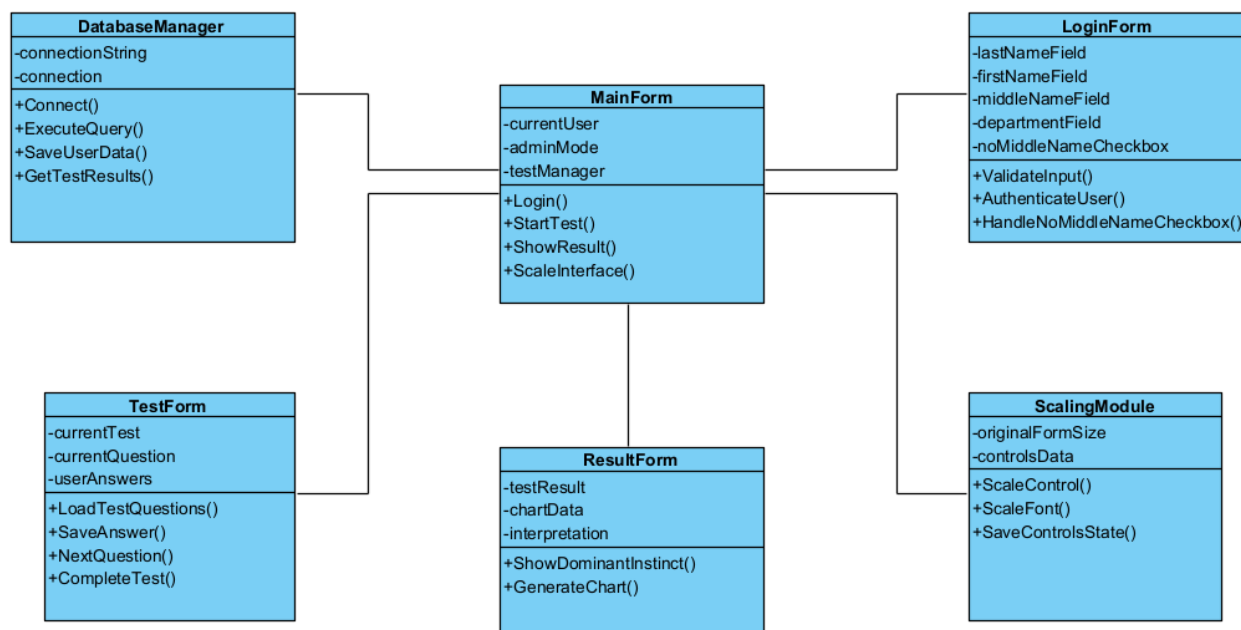


Рисунок 13 – Диаграмма классов основного приложения

В таблице 4 представлена спецификация основных классов приложения.

Класс MainForm является центральным компонентом, обеспечивающим взаимодействие пользователя с системой через графический интерфейс приложения.

Спецификация классов отражает четкое разделение обязанностей: интерфейсы отвечают за взаимодействие с пользователем, сервисы – за бизнес-логику, а репозитории – за работу с данными. Такая структура обеспечивает гибкость при добавлении новых типов тестов и методик анализа, не нарушая работу существующих компонентов. Взаимодействие классов реализовано через события и вызовы методов, что минимизирует прямые зависимости между модулями.

Таблица 4 – Спецификация классов

Класс	Описание класса	Поля	Описание полей	Методы	Описание методов
MainForm	Класс основного окна приложения	currentUser	Текущий пользователь	Login()	Вход в систему
		adminMode	Режим администратора	StartTest()	Начать тестирование
		testManager	Менеджер тестирования	ShowResults()	Показать результаты
				ScaleInterface()	Масштабировать интерфейс

Продолжение таблицы 4

Класс	Описание класса	Поля	Описание полей	Методы	Описание методов
LoginForm	Класс формы входа в систему	lastNameField	Поле для фамилии	ValidateInput()	Проверка входных данных
		firstNameField	Поле для имени	AuthenticateUser()	Аутентификация пользователя
		middleNameField	Поле для отчества	HandleNoMiddleNameCheckbox()	Обработка чекбокса «Нет отчества»
		departmentField	Поле для отделения		
		noMiddleNameCheckbox	Чекбокс «Нет отчества»		
TestForm	Класс формы прохождения теста	currentTest	Текущий тест	LoadTestQuestions()	Загрузка вопросов теста
		currentQuestion	Текущий вопрос	SaveAnswer()	Сохранить ответ
		userAnswers	Ответы пользователя	NextQuestion()	Перейти к следующему вопросу
				CompleteTest()	Завершить тест
ResultForm	Класс формы отображения результатов	testResult	Результат теста	ShowDominantInstinct()	Показать доминирующий инстинкт
		chartData	Данные для диаграммы	GenerateChart()	Создать диаграмму
		interpretation	Данные для интерпретации		
ScalingModule	Класс масштабирования интерфейса	originalFormSize	Исходный размер формы	ScaleControl()	Масштабирование элемента управления
		controlsData	Данные о элементах управления	ScaleFont()	Масштабирование шрифта
				SaveControlsState()	Сохранение положения на экране элементов управления
DatabaseManager	Класс управления базой данных	connectionString	Строка подключения	Connect()	Подключение к базе данных
		connection	Подключение	ExecuteQuery()	Выполнить запрос
				SaveUserData()	Сохранить данные пользователя
				GetTestResults()	Получить результаты тестирования пользователя

Спроектированная диаграмма классов основного приложения позволяет организовать структуру программного обеспечения для психологического тестирования, а также уточнить технические детали реализации. Классы разделены на логические группы в соответствии с функциональными требованиями системы. Объединенная диаграмма деятельности для программы психологического тестирования изображена на рисунке 14.

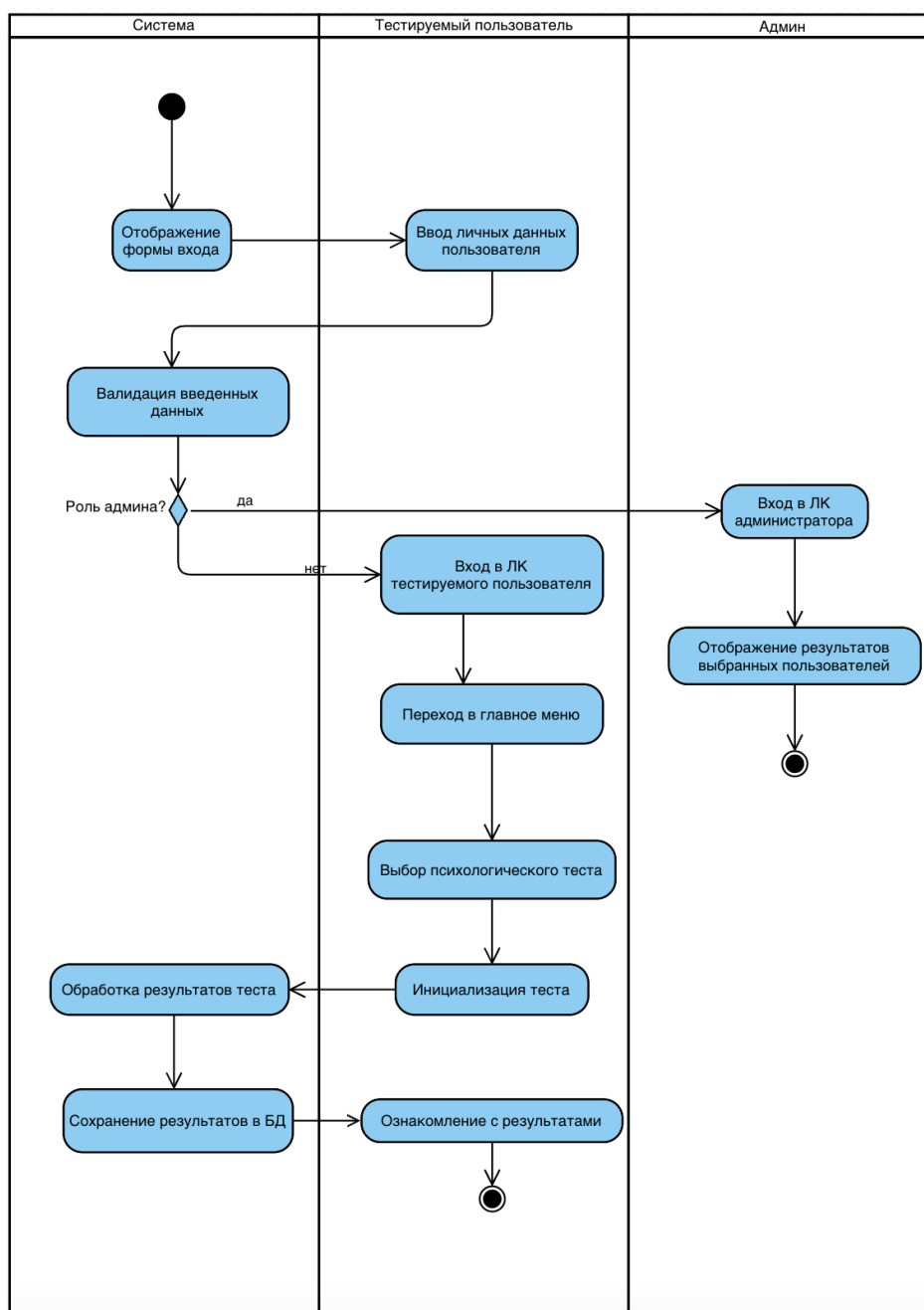


Рисунок 14 – Диаграмма деятельности

Пользовательский интерфейс представлен классами MainForm, LoginForm, TestForm и ResultForm, которые отвечают за взаимодействие с пользователем. Основная бизнес-логика реализована в классе TestManager, который обеспечивает функциональность системы тестирования. Доступ к данным осуществляется через класс DatabaseManager, который абстрагирует работу с базой данных от остальной части приложения.

Особое внимание уделено классу ScalingModule, который отвечает за масштабирование интерфейса в зависимости от размера экрана, что обеспечивает корректное отображение на различных устройствах. Это решение повышает удобство использования программы и расширяет круг поддерживаемых конфигураций оборудования.

2.3 Проектирование пользовательского интерфейса

Интерфейс приложения должен предоставлять беспрепятственный доступ к основным возможностям программного обеспечения для психологического тестирования, а также располагать эффективными и простыми инструментами взаимодействия с приложением.

Проектируемое приложение является многостраничным. Это подразумевает наличие нескольких разделов доступа к отдельным модулям приложения. Данный выбор архитектуры пользовательского интерфейса обусловлен умеренным количеством размещаемых элементов управления по различным частям приложения, которые требуют уникального взаимодействия с пользователем. Интерфейс программы важен для обеспечения удобного взаимодействия с системой психологического тестирования. Он должен включать страницы входа, регистрации и главную страницу с навигационным меню. На главной странице будут реализованы вкладки для доступа к тестам, результатам и управлению пользователями и результатами для администраторов.

Основные страницы:

- страница входа/регистрации: поля для ввода ФИО, выбора отделения, логина и пароля (для администраторов), чекбокс «Нет отчества» скрывает соответствующее поле;

- главная страница: содержит список доступных тестов, историю результатов с фильтрами по дате, для администраторов – разделы управления тестами, пользователями и аналитики.

Модальные окна:

- прохождение теста: отображает вопросы по одному в формате вопрос – варианты ответов;

- детали результата: показывает текстовую интерпретацию психотипа и диаграмму по результатам тестирования;

- управление тестами (для администраторов): окна добавления/редактирования пользователей, данных и импорт/экспорт базы данных.

Спроектированный пользовательский интерфейс должен позволять пользователю эффективно использовать основной функционал программного обеспечения для психологического тестирования в комфортном для представления виде. Интерфейс будет учитывать различные сценарии использования как для обычных пользователей, так и для администраторов системы.

3 Реализация программных средств психологического тестирования

3.1 Реализация базы данных

Физическая модель базы данных представляет собой логическую модель, выраженную в терминологии языка описания конкретной СУБД [12]. В рамках данного приложения используется СУБД Microsoft SQL Server.

Физические модели баз данных определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне.

Согласно концептуальной и логической модели, была разработана физическая модель данных, представленная на рисунке 15.

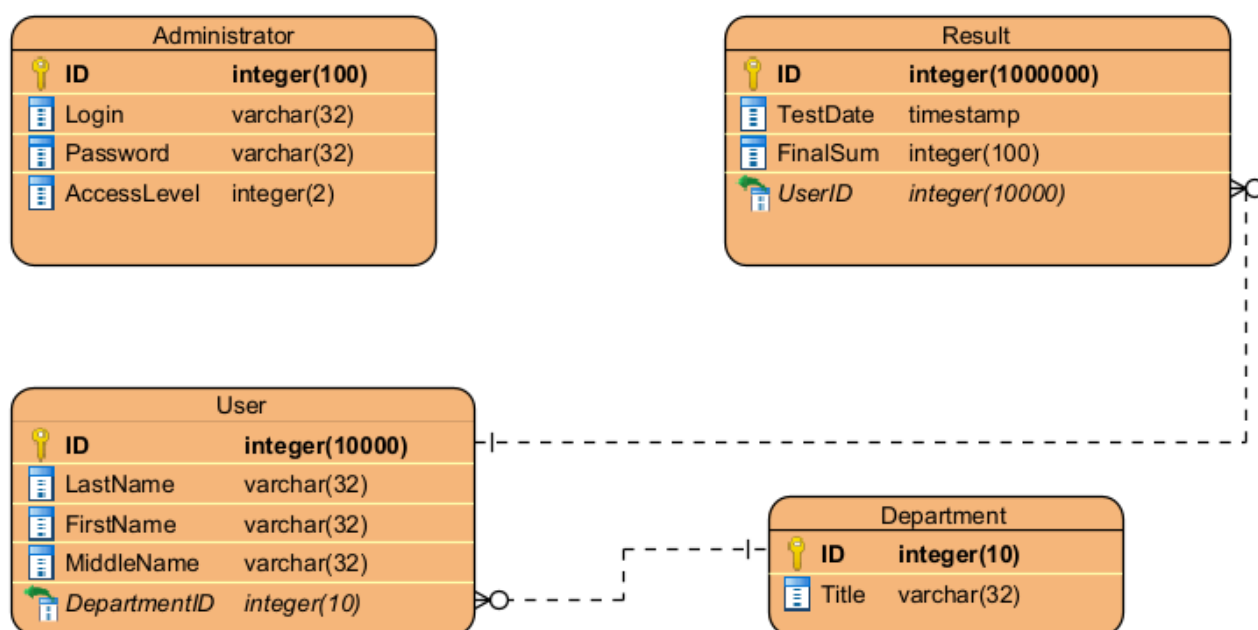


Рисунок 15 – Физическая модель базы данных

Физическая модель базы данных была разработана с помощью Visual Paradigm Community Edition 17.1. На основе физической модели, была разработана база данных с использованием современной СУБД Microsoft SQL Server 2019 [13].

В таблице 5 представлена спецификация сущности «Пользователь».

Таблица 5 – Спецификация сущности «Пользователь»

Наименование	Описание	Тип данных	NULL	ОДЗ	Значение по умолчанию	Роль
ID	Идентификатор	INTEGER(10000)	Нет	> 0	-	PK
LastName	Фамилия	VARCHAR(32)	Нет	2 – 32 символа	-	-
FirstName	Имя	VARCHAR(32)	Нет	2 – 32 символа	-	-
MiddleName	Отчество	VARCHAR(32)	Да	2 – 32 символа	NULL	-
DepartmentID	Идентификатор отделения	INTEGER(10)	Нет	> 0	-	FK

В таблице 6 представлена спецификация сущности «Администратор».

Таблица 6 – Спецификация сущности «Администратор»

Наименование	Описание	Тип данных	NULL	ОДЗ	Значение по умолчанию	Роль
ID	Идентификатор	INTEGER(100)	Нет	> 0	-	PK
Login	Логин	VARCHAR(32)	Нет	Уник.	-	-
Password	Пароль	VARCHAR(32)	Нет	-	-	-
AccessLevel	Уровень доступа	INTEGER(2)	Нет	1 – 2	1	-

В таблице 7 представлена спецификация сущности «Отделение».

Таблица 7 – Спецификация сущности «Отделение»

Наименование	Описание	Тип данных	NULL	ОДЗ	Значение по умолчанию	Роль
ID	Идентификатор	INTEGER(10)	Нет	> 0	-	PK
Title	Название	VARCHAR(32)	Нет	2 – 32 символа	-	-

В таблице 8 представлена спецификация сущности «Результаты».

Таблица 8 – Спецификация сущности «Результаты»

Наименование	Описание	Тип данных	NULL	ОДЗ	Значение по умолчанию	Роль
ID	Идентификатор	INTEGER(100000)	Нет	> 0	-	PK
UserID	Идентификатор пользователя	INTEGER(10000)	Нет	> 0	-	FK
TestDate	Дата тестирования	TIMESTAMP	Нет	-	-	-
FinalSum	Итоговая сумма за тестирование	INTEGER(100)	Нет	-	-	-

Таким образом, реализованная физическая схема базы данных позволяет эффективно хранить сведения о пользователях и результатах тестирования в рамках программного обеспечения для психологического тестирования.

3.2 Реализация алгоритмической части обработки результатов

Алгоритмическая часть обработки результатов тестирования является ключевым компонентом разрабатываемого программного обеспечения для психологического тестирования. В данном разделе рассматривается реализация алгоритмов обработки ответов пользователей и формирования интерпретации по методикам В. Гарбузова (определение доминирующего инстинкта) и геометрического теста для определения психотипа личности.

Методика В. Гарбузова для определения доминирующего инстинкта основана на системе из 15 вопросов, ответы на которые позволяют определить выраженность семи базовых инстинктов человека [14].

На первом этапе сбора ответов пользователя система собирает и сохраняет ответы пользователя на все вопросы теста. Каждый ответ однозначно связывается с определенным вопросом и сохраняется в базе данных с указанием времени ответа и идентификатора пользователя. Это позволяет не только получить итоговый результат, но и сохранить всю историю тестирования для возможного последующего анализа.

Каждый вопрос в методике В. Гарбузова связан с одним или несколькими инстинктами [15]. Алгоритм анализирует полученные ответы и классифицирует их в соответствии с заложенной в систему матрицей соответствия вопросов и инстинктов. Для каждого из семи инстинктов (эгофильный, генофильный, альтруистический, исследовательский, доминантный, либертофильный, дигниофильный) ведется отдельный подсчет баллов.

Алгоритм подсчитывает сумму баллов по каждому из семи инстинктов на основе ответов пользователя. Каждый ответ, связанный с определенным инстинктом, увеличивает счетчик соответствующего инстинкта на

установленное значение. Суммы баллов нормализуются для обеспечения сопоставимости результатов между различными инстинктами.

После подсчета баллов алгоритм находит инстинкт с наибольшим количеством баллов, который и определяется как доминирующий. В случае если два или более инстинкта набирают одинаковое количество баллов, система фиксирует смешанный тип с указанием всех доминирующих инстинктов. Особое внимание уделяется случаям, когда по инстинктам альтруизма, самосохранения или продолжения рода отсутствуют баллы, что является диагностически значимым показателем.

На основе определенного доминирующего инстинкта система формирует текстовую интерпретацию результата. Для каждого инстинкта в базе данных хранится подробное описание, включая ведущие качества, характеристики поведения и рекомендации. Алгоритм динамически составляет интерпретацию, включая общее описание типа и список ведущих качеств личности. При смешанном типе в интерпретацию включаются характеристики всех доминирующих инстинктов.

Для наглядного представления результатов реализован алгоритм построения диаграммы, отображающей выраженность каждого из семи инстинктов. Диаграмма генерируется с использованием библиотеки для работы с графикой, встроенной в Windows Forms [16]. Каждый инстинкт представлен отдельным столбцом, высота которого пропорциональна количеству набранных баллов.

Полная информация о результатах тестирования сохраняется в базе данных SQL Server. Это включает в себя информацию о доминирующем инстинкте, баллах по каждому из семи инстинктов, сформированную интерпретацию и дату проведения тестирования. Такой подход обеспечивает возможность последующего анализа результатов администратором системы.

Геометрический тест для определения психотипа личности основан на выборе геометрических фигур: треугольник, квадрат, круг, зигзаг,

прямоугольник. Алгоритм обработки результатов по данной методике реализован следующим образом:

Алгоритм выбора пользователя анализирует первый и второй выбор пользователя среди предложенных геометрических фигур [17]. Первый выбор имеет большее значение для определения основного психотипа, в то время как второй выбор дает дополнительную информацию о вторичных характеристиках личности.

На основе выбранных геометрических фигур система определяет основной психотип личности. Каждой фигуре соответствует определенный психологический тип:

- треугольник – лидерство, решительность, соперничество;
- квадрат – трудолюбие, усердие, терпение, аккуратность;
- круг – коммуникабельность, эмпатия, доброжелательность;
- зигзаг – творчество, энтузиазм, мечтательность;
- прямоугольник – состояние перехода и изменения.

Алгоритм формирует комбинированную характеристику личности на основе первого и второго выбора пользователя. Например, если первый выбор – треугольник, а второй – круг, то формируется характеристика, включающая основные качества лидера с элементами коммуникабельности и эмпатии. Система использует заранее определенные шаблоны интерпретаций для различных комбинаций психотипов.

На основе определенного психотипа и его комбинации алгоритм генерирует подробную текстовую интерпретацию. Для каждого психотипа и их комбинаций в базе данных хранятся развернутые описания, которые включают характеристики личности, сильные и слабые стороны, особенности поведения и рекомендации.

Для наглядного представления результата реализован алгоритм отображения выбранных геометрических фигур с соответствующими пояснениями. Система визуализирует первый и второй выбор пользователя с указанием их значимости и соответствующих психологических характеристик.

Результаты геометрического теста сохраняются в базе данных аналогично результатам теста В. Гарбузова. Это включает информацию о выбранных фигурах, определенном психотипе, сформированной интерпретации и дате проведения тестирования.

Тест на направленность личности определяет преобладающие мотивационные ориентации личности и является одним из ключевых компонентов разрабатываемой системы психологического тестирования.

На первом этапе система собирает и сохраняет ответы пользователя на все вопросы теста направленности личности. Каждый ответ однозначно связывается с определенным вопросом и сохраняется в базе данных с указанием времени ответа и идентификатора пользователя.

Каждый вопрос в тесте направленности личности связан с определенными мотивационными ориентациями. Алгоритм анализирует полученные ответы и классифицирует их в соответствии с заложенной в систему матрицей соответствия вопросов и типов направленности.

Алгоритм подсчитывает сумму баллов по каждому типу направленности на основе ответов пользователя. Каждый ответ, связанный с определенным типом направленности, увеличивает счетчик соответствующей категории на установленное значение. Суммы баллов нормализуются для обеспечения сопоставимости результатов между различными типами направленности личности.

На основе определенного доминирующего типа направленности система формирует текстовую интерпретацию результата. В ней можно увидеть, сколько баллов получил тестируемый в каждом из направлений личности.

Результаты тестирования на направленность личности сохраняются в базе данных аналогично остальным результатам теста. Результаты включают информацию об ответах пользователя, баллы за каждый из вопросов, сформированную интерпретацию и дату проведения тестирования.

Алгоритмическая часть обработки результатов тестирования реализована в виде отдельного модуля ResultProcessor, который взаимодействует с другими

компонентами системы. Этот модуль получает ответы пользователя от менеджера тестирования (TestManager), обрабатывает их с использованием описанных выше алгоритмов и передает результаты для отображения в пользовательском интерфейсе и сохранения в базе данных.

Таким образом, реализованная алгоритмическая часть обработки результатов тестирования обеспечивает эффективный анализ ответов пользователей, формирование психологической характеристики личности и наглядное представление результатов тестирования.

3.3 Реализация основного приложения

Основное приложение было реализовано в среде разработки Microsoft Visual Studio 2022 в качестве исполняемого проекта Windows Forms. Windows Forms – это технология для построения клиентских приложений Windows с графическим интерфейсом пользователя, являющаяся частью .NET Framework [18]. Окно авторизации модуля авторизации представлено на рисунках 16 и 17.

Рисунок 16 – Графический интерфейс раздела «Авторизация» для пользователя

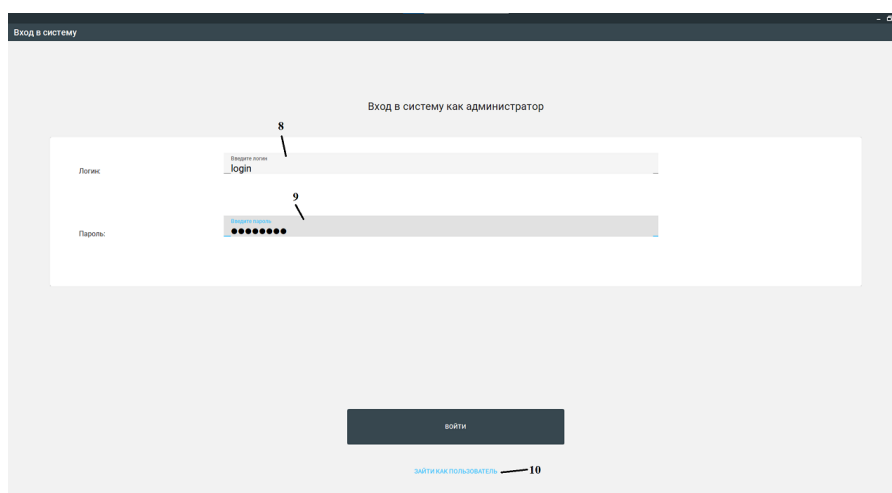


Рисунок 17 – Графический интерфейс раздела «Авторизация» для администратора

Окно авторизации представлено следующими графическими элементами:

- 1 – поле ввода фамилии;
- 2 – поле ввода имени;
- 3 – поле ввода отчества;
- 4 – поле ввода отделения;
- 5 – чекбокс «Нет отчества»;
- 6 – кнопка входа в систему;
- 7 – кнопка входа в режим администратора;
- 8 – поле ввода логина;
- 9 – поле ввода пароля;
- 10 – кнопка входа в режим пользователя.

Раздел главного меню представлен на рисунке 18.

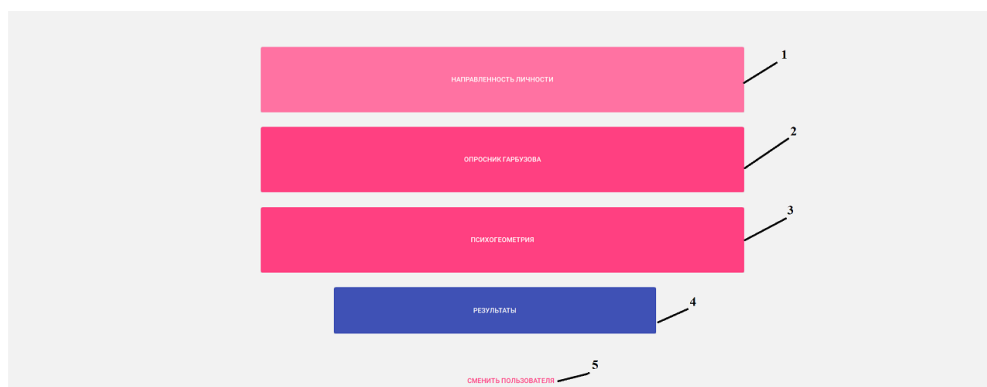


Рисунок 18 – Графический интерфейс раздела «Главное меню»

Данный раздел представлен следующими графическими элементами:

- 1 – кнопка выбора теста на направленность личности;
- 2 – кнопка выбора теста опросника Гарбузова;
- 3 – кнопка выбора психометрического теста;
- 4 – кнопка просмотра результатов предыдущих тестирований;
- 5 – кнопка смены пользователя.

Раздел прохождения теста модуля прохождения психологических тестов представлен на рисунке 19.

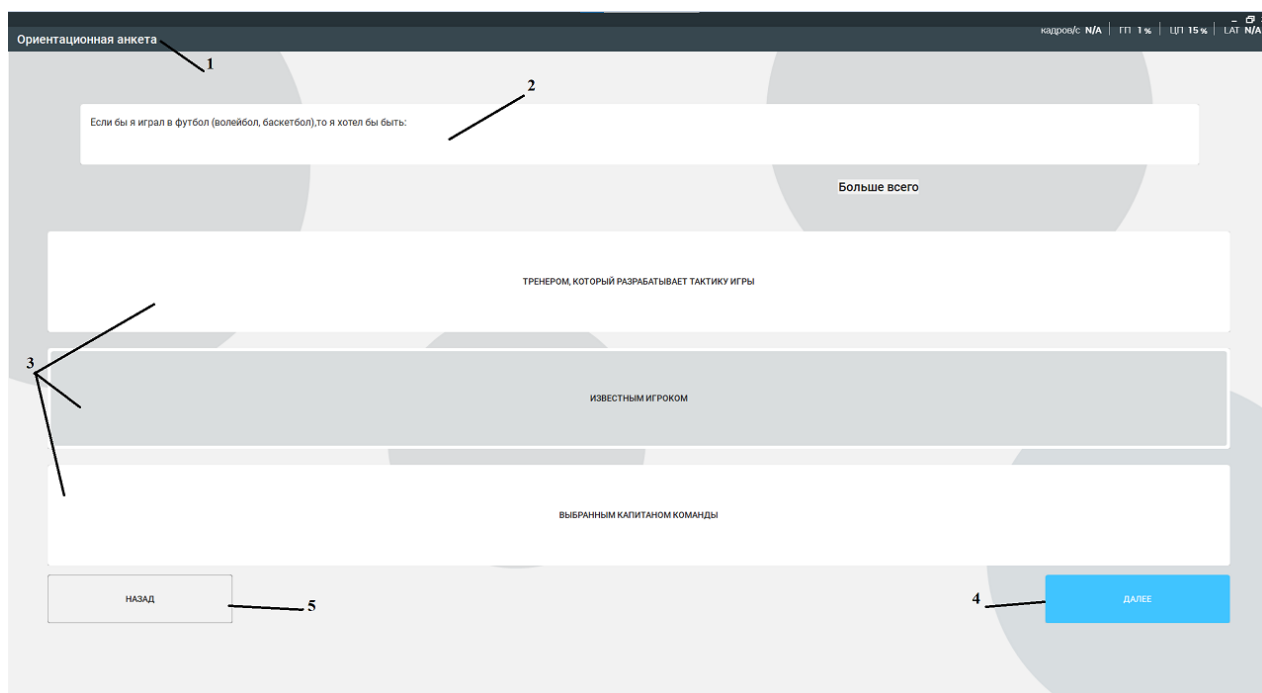


Рисунок 19 – Графический интерфейс раздела «Прохождение теста»

Страница прохождения теста представлена следующими графическими элементами:

- 1 – информационная панель с названием анкеты;
- 2 – область отображения текущего вопроса;
- 3 – варианты ответов на текущий вопрос;
- 4 – кнопка перехода к следующему вопросу;
- 5 – кнопка возврата к предыдущему вопросу.

Раздел результатов тестирования модуля результатов психологического тестирования приведен на рисунке 20 и 21.

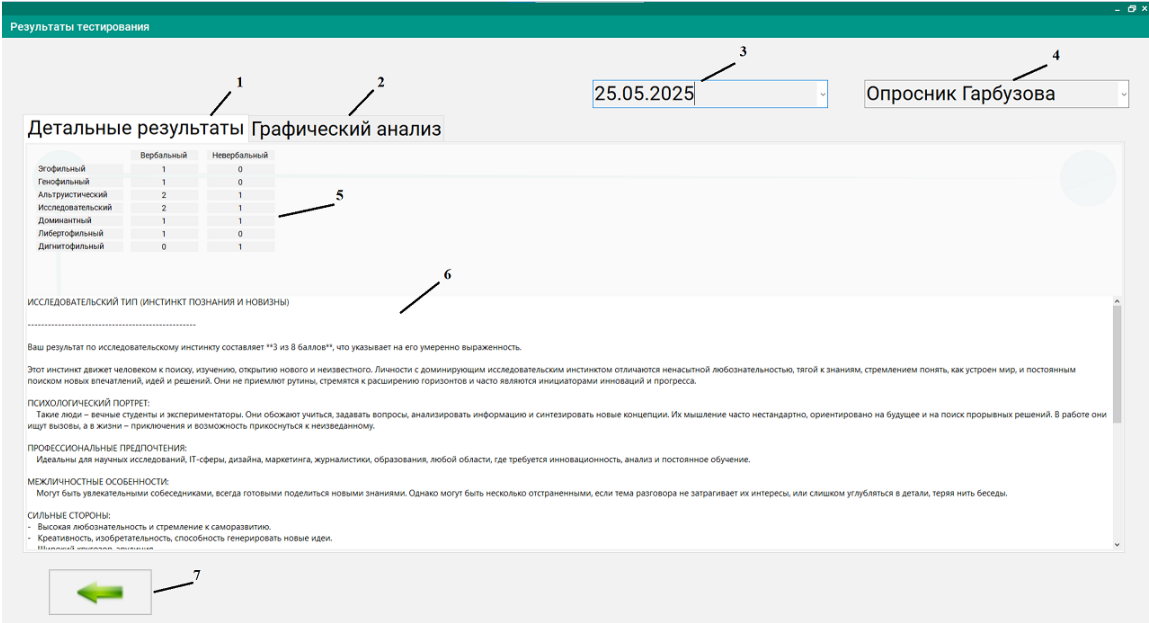


Рисунок 20 – Детальные результаты раздела «Результаты тестирования»

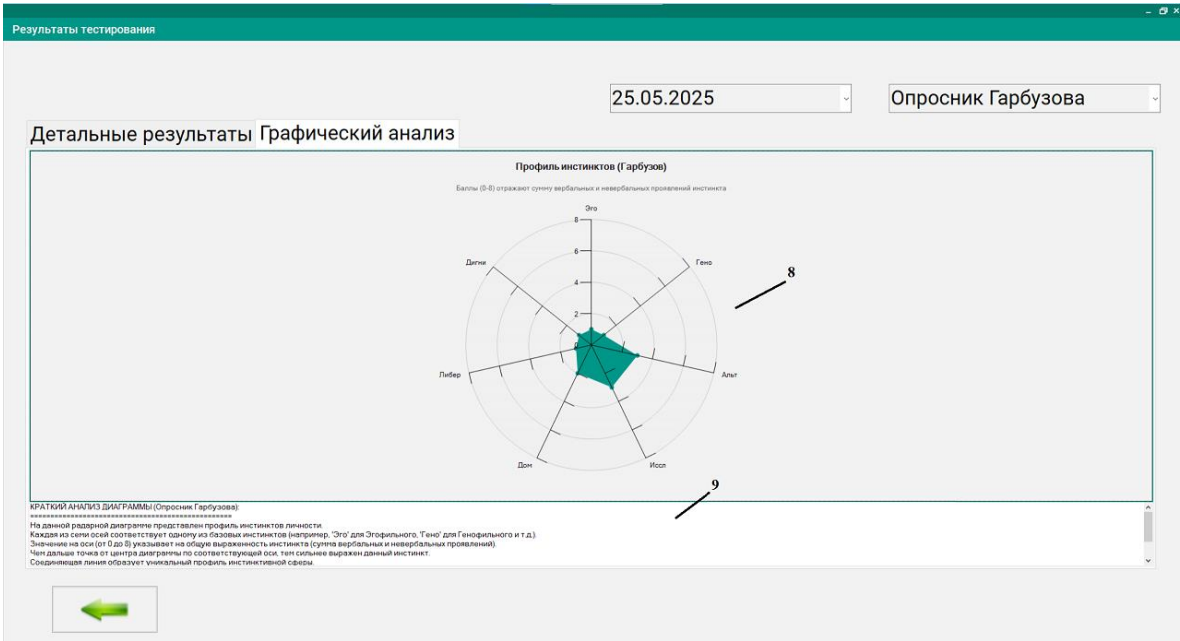


Рисунок 21 – Графический анализ раздела «Результаты тестирования»

- Данный раздел представлен следующими графическими элементами:
- 1 – вкладка с детальными результатами выбранного тестирования;
 - 2 – вкладка с графическим анализом выбранного тестирования;
 - 3 – дата прохождения тестирования;
 - 4 – название выбранного тестирования;

- 5 – таблица результатов тестирования по баллам;
- 6 – текстовая интерпретация результатов тестирования;
- 7 – кнопка возврата в главное меню;
- 8 – диаграмма по результатам тестирования;
- 9 – информационная панель с кратким анализом диаграммы.

Для администратора (профессионального психолога) предусмотрен дополнительный раздел – панель администратора, в которой реализованы расширенные функциональные возможности приложения, в том числе редакторы данных и тестов, а также подробные результаты тестирования, меню раздела представлено на рисунке 22.

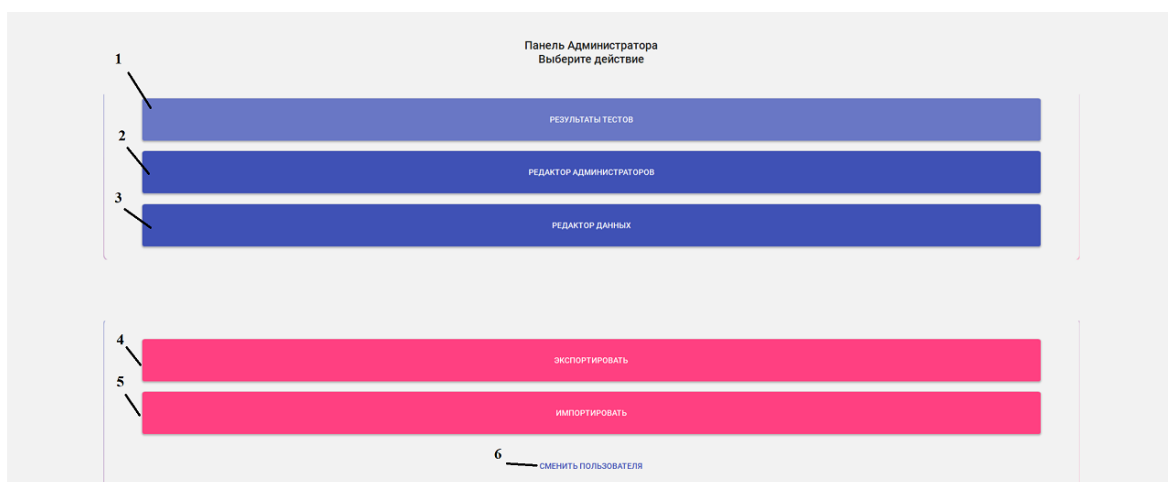


Рисунок 22 – Меню раздела «Панель администратора»

Окно «Панель администратора» содержит следующие графические элементы:

- 1 – кнопка перехода к разделу «Регистрация тестов»;
- 2 – кнопка перехода к разделу «Редактировать администраторов»;
- 3 – кнопка перехода к разделу «Редактировать данные»;
- 4 – кнопка перехода к разделу «Экспортировать»;
- 5 – кнопка перехода к разделу «Импортировать»;
- 6 – кнопка выхода из административной панели.

На рисунке 23 изображен интерфейс отображения детальных результатов тестирования административного модуля.

Результаты тестирования

7 test123 8 Test Test Test 9 Опросник Гарбузова 10 24.05.2025

11 Детальные результаты 12 Графический анализ

	Вербальный	Невербальный
Эгофильный	3	0
Генофильный	4	0
Альтруистический	3	1
Исследовательский	3	3
Доминантный	2	0
Либертофильный	2	0
Дигнилофильный	1	0

13

14 ОБЩИЙ АНАЛИЗ РЕЗУЛЬТАТОВ: Опросник Гарбузова

Эта вкладка предназначена для подробной интерпретации результатов тестирования по 7 базовым инстинктам (Эгофильный, Генофильный, Альтруистический, Исследовательский, Доминантный, Либертофильный, Дигнилофильный).
 Чтобы получить 'детальное психологическое описание' каждого инстинкта, пожалуйста, нажмите на соответствующее название инстинкта выше.
 Вы узнаете об особенностях его проявлений, сильных сторонах и потенциальных зонах развития, связанных с этим инстинктом.

15 ← 16 Сформировать таблицу по уч. отд. 17 ? 18 🗑️

Рисунок 23 – Результаты тестов раздела «Детальные результаты»

Раздел содержит следующие элементы:

- 7 – поле с названием отделения тестируемого;
- 8 – поле с ФИО тестируемого;
- 9 – выпадающий список выбора тестов;
- 10 – поле выбора даты проведения тестирования;
- 11 – заголовок секции с отображением детальных результатов;
- 12 – заголовок секции с отображением графического анализа;
- 13 – таблица с результатами по каждому из параметров теста;
- 14 – текстовая область с аналитическим описанием результатов;
- 15 – кнопка возврата к предыдущему экрану;
- 16 – кнопка формирования отчета по выбранному учебному отделению;
- 17 – кнопка для отображения подробной информации об интерпретации результатов;
- 18 – кнопка удаления выбранного результата тестирования.

Интерфейс раздела результатов тестирования административного модуля представлен на рисунке 24.

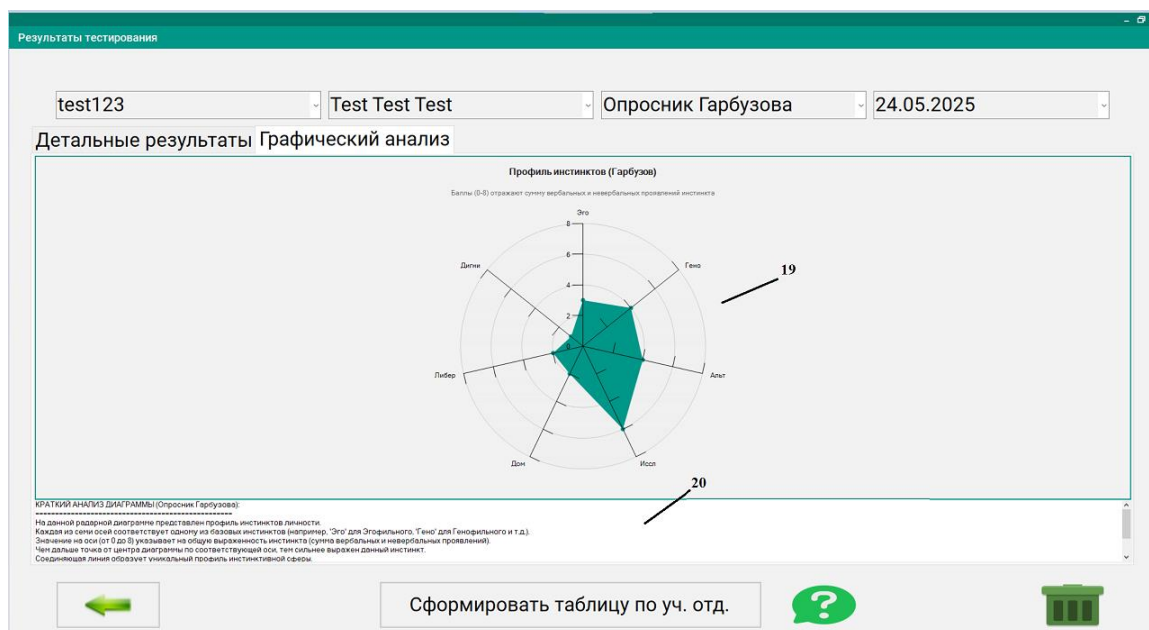


Рисунок 24 – Результаты тестов раздела «Результат тестирования»

Раздел содержит следующие графические элементы:

19 – график «Профиль инстинктов (Гарбузова)», визуализирующий индивидуальные баллы по шкале инстинктов в виде радиальной диаграммы;

20 – текстовое поле с кратким пояснением и интерпретацией значений графика, включающее выводы и рекомендации.

Графический интерфейс раздела управления администраторами представлена на рисунке 25.

Рисунок 25 – Раздел «Управление администраторами»

Данный раздел содержит следующие элементы:

- 21 – поле ввода логина администратора;
- 22 – поле ввода пароля администратора;
- 23 – поле для ввода уровня доступа выбранного администратора;
- 24 – кнопка для добавления нового администратора или изменения данных текущего;
- 25 – возврат к предыдущему окну панели администратора;
- 26 – кнопка для удаления выбранного администратора;
- 27 – информационное сообщение об уровнях доступа администраторов;
- 28 – счетчик всех зарегистрированных администраторов.

Интерфейс раздела редактирования данных административного модуля представлен на рисунке 26.

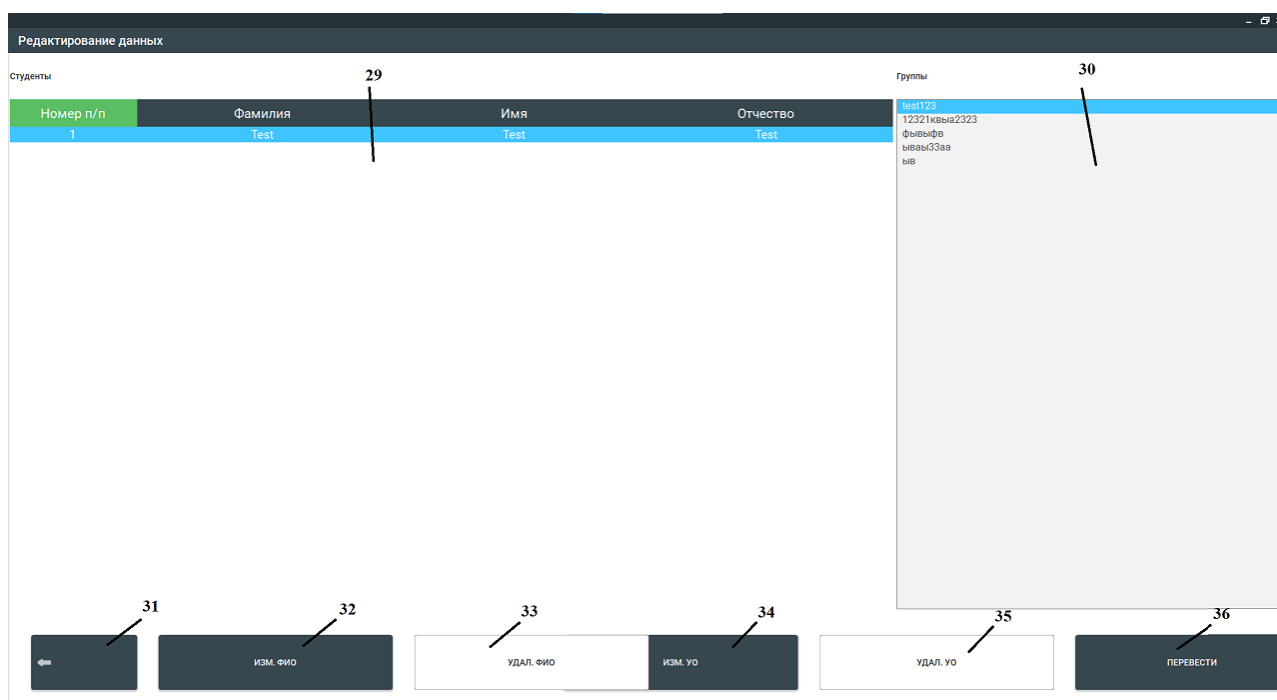


Рисунок 26 – Раздел «Редактирование данных»

Раздел включает в себя следующие графические элементы:

- 29 – таблица с персональными данными (столбцы: номер по порядку, фамилия, имя, отчество);
- 30 – список групп для выбора обучающихся;
- 31 – кнопка возврата на предыдущее окно;

- 32 – кнопка для изменения ФИО тестируемого;
- 33 – удаления тестируемого из учебного отделения;
- 34 – кнопка изменения учебного отделения для выбранного тестируемого;
- 35 – кнопка для удаления учебного отделения;
- 36 – кнопка перевода тестируемого в другое учебное отделение.

Данный раздел предназначен для управления пользовательскими данными в системе. Внесенные изменения сохраняются только после нажатия соответствующей кнопки, что позволяет избежать случайной потери информации.

Окно экспорта базы данных административного модуля представлено на рисунке 27.

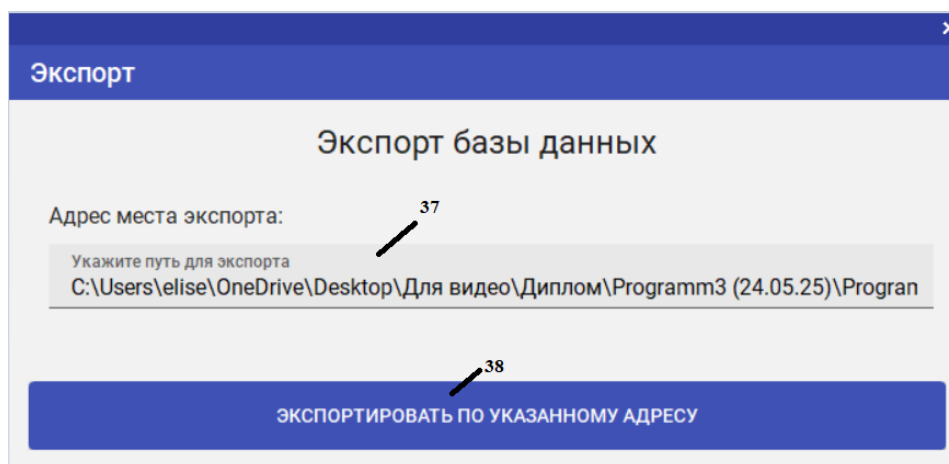


Рисунок 27 – Экспортирование базы данных раздела «Панель администратора»

Данное окно содержит следующие графические элементы:

- 37 – поле для указания пути, по которому будет сохранен файл экспорта;
- 38 – кнопка «Экспортировать по указанному адресу» для запуска процесса сохранения данных.

Данный функционал позволяет администратору сохранить резервную копию базы данных в выбранное расположение. Это обеспечивает надежность хранения информации и возможность последующего восстановления.

Окно импорта базы данных административного модуля представлено на рисунке 28.

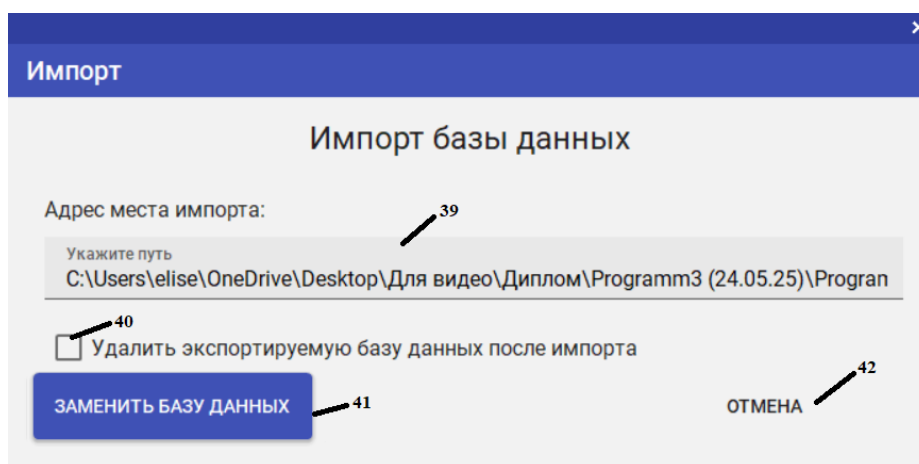


Рисунок 28 – Импортирование базы данных раздела «Панель администратора»

Окно содержит следующие графические элементы:

39 – поле для указания пути к файлу базы данных, предназначенной для импорта;

40 – чекбокс для удаления экспортируемой базы данных после импорта;

41 – кнопка для выполнения импорта и замены текущей базы;

42 – кнопка для выхода из окна без выполнения действий.

Функционал предназначен для восстановления или обновления текущей базы данных. Возможность удаления предыдущей базы после успешного импорта позволяет избежать дублирования и перегрузки системы.

Для наглядного отражения взаимодействия модулей приложения была разработана диаграмма компонентов, приведенная на рисунке 29.

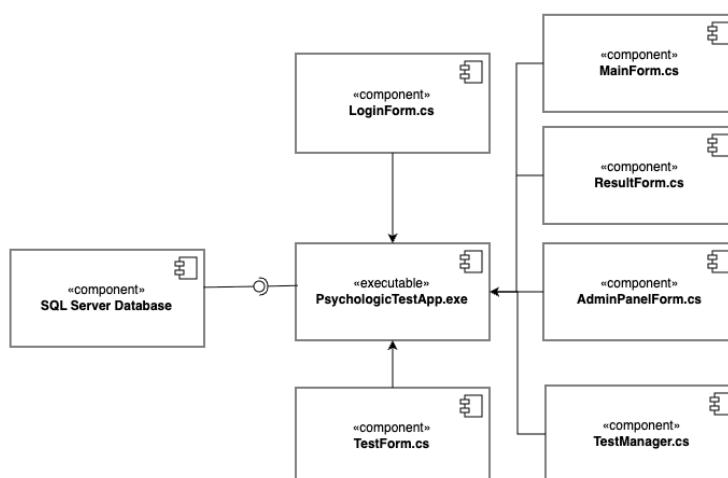


Рисунок 29 – Диаграмма компонентов разворачиваемого приложения

Для демонстрации разворачивания используемых компонентов в приложении, была разработана диаграмма развертывания, приведенная на рисунке 30.

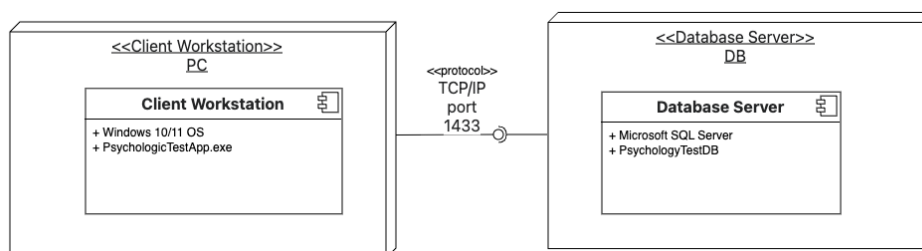


Рисунок 30 – Диаграмма развертывания приложения

Данная диаграмма отражает развертывание элементов разрабатываемого приложения в пространстве. Разработанное приложение может быть развернуто на персональных компьютерах с операционной системой семейства Windows версии 10 и выше.

В процессе разработки основного приложения были реализованы все ключевые функциональные возможности программы, которые планировались при проектировании.

Реализована форма входа в систему, обеспечивающая валидацию вводимых данных в соответствии со спецификацией (проверка полей «Имя», «Фамилия», «Отчество», «Отделение» на соответствие требуемым форматам). Обеспечена возможность указания отсутствия отчества с помощью специального флага, что скрывает соответствующее поле ввода.

Реализован интерфейс для последовательного отображения вопросов психологических тестов, сбора и сохранения ответов пользователя. Система поддерживает три основных типа тестов: тест на определение доминирующего инстинкта по методике В. Гарбузова, тест на направленность личности и геометрический тест для определения психотипа личности.

Разработана форма для наглядного представления результатов тестирования, включающая диаграмму распределения баллов по различным инстинктам или психотипам, а также текстовую интерпретацию результатов.

Для пользователей с правами администратора (психологов) реализован дополнительный модуль управления, позволяющий просматривать результаты тестирования пользователей, формировать статистические отчеты и управлять учетными записями в системе.

Разработан специальный модуль, обеспечивающий корректное масштабирование элементов интерфейса в зависимости от размера окна и разрешения экрана. Это позволяет обеспечить комфортную работу с приложением на устройствах с различными характеристиками дисплея.

Реализованы функции для эффективной работы с базой данных, включая создание, чтение, обновление и удаление записей о пользователях и результатах тестирования. Обеспечена целостность данных и оптимизация запросов для повышения производительности приложения.

Для обеспечения надежности и удобства сопровождения приложения код организован в соответствии с принципами объектно-ориентированного программирования.

Особое внимание при разработке было уделено обработке исключительных ситуаций и валидации данных. Реализована система логирования, которая фиксирует ключевые события в работе приложения и помогает в диагностике возможных проблем.

Совокупность разработанных программных средств позволяет эффективно проводить психологическое тестирование, сохранять и анализировать результаты, предоставляя пользователю и администратору системы удобный и функциональный интерфейс. Исходный код разработанных программных средств приведен в приложении Б.

4 Контроль стандартов качества программных решений

4.1 Функциональное тестирование

При выходе в программу пользователь должен ввести данные о себе, поля «Имя», «Фамилия» и «Отчество» должны содержать 2 – 32 символа: кириллицу или дефис. Поле «Отделение» должно содержать 2 - 16 символов: цифры или кириллицу. Чекбокс «Нет отчества» может быть либо поставлен, либо нет, при его постановке поле «Отчество» пропадает. В зависимости от введенных в поля данных, программа может выдавать различные сообщения, в которые входят:

- сообщение о некорректной фамилии;
- сообщение о некорректном имени;
- сообщение о некорректном отчестве;
- сообщение о некорректном отделении;
- сообщение об успешном входе в программу.

В таблице 9 представлен набор причин и следствий для определения зависимости между входными данными пользователя и реакцией системы.

Таблица 9 – Набор причин и следствий

Причины	
П1	Фамилия корректна
П2	Имя корректно
П3	Отчество корректно
П4	Чек бокс «Нет отчества» установлен
П5	Отделение корректно
Следствия	
С1	Сообщение: «Некорректная фамилия»
С2	Сообщение: «Некорректное имя»
С3	Сообщение: «Некорректное отчество»
С4	Сообщение: «Некорректное отделение»
С5	Сообщение: «Вход выполнен»

Набор причин и следствий определяет функциональную диаграмму, которая отражает зависимости между входными данными пользователя и

реакцией системы при различных условиях. Диаграмма позволяет визуализировать возможные комбинации состояний интерфейса и соответствующих сообщений, генерируемых программой. Она служит основой для построения набора тестовых сценариев и анализа корректности функционирования логики обработки данных. Такая визуализация упрощает отладку и верификацию поведения интерфейса на этапе тестирования. Набор причин и следствий определяет функциональную диаграмму, представленную на рисунке 31.

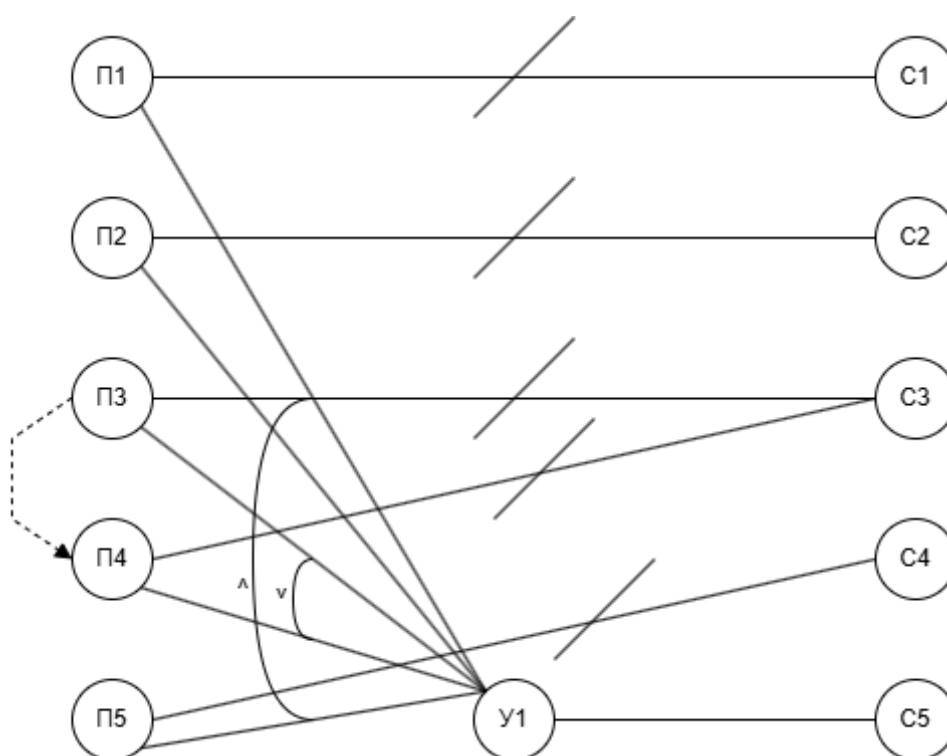


Рисунок 31 – Функциональная диаграмма

При комбинаторном переборе всех возможных вариантов входных данных получается 216 тестовых сценариев. С целью сокращения количества вхождений во множество допустимых тестов используется комбинаторный метод тестирования программного обеспечения, называемый попарным тестированием, который для каждой пары входных параметров системы проверяет все возможные дискретные комбинации этих параметров. Данная техника позволяет уменьшить количество тестовых сценариев с минимальным вмешательством в качество проводимого тестирования.

Результат применения техники попарного тестирования оставил из 216 тестовых сценариев лишь 14. Тестовые сценарии после применения техники попарного тестирования представлены на таблице 10.

Таблица 10 – Тестовые сценарии после применения техники попарного тестирования

№	П1	П2	П3	П4	П5	С1	С2	С3	С4	С5
1	Иванов	О	НетОтчества	да	Ъ	0	1	0	1	0
2	Ив@нов	Петр	НетОтчества	да	!!!!!!	1	0	0	1	0
3	Ъ	Петр	Ив@нов!!ч	нет	22BB12	1	0	1	0	0
4	Ив@нов	П!тр	Ы	нет	Ъ	1	1	1	1	0
5	Ъ	П!тр	НетОтчества	да	22BB12	1	1	0	0	0
6	Ъ	О	Ы	нет	!!!!!!	1	1	1	1	0
7	Ив@нов	О	Ив@нов!!ч	нет	22BB12	1	1	1	0	0
8	Иванов	Петр	Иванович	нет	Ъ	0	0	0	1	0
9	Иванов	П!тр	Иванович	нет	22BB12	0	1	0	0	0
10	Иванов	П!тр	Ив@нов!!ч	нет	!!!!!!	0	1	1	1	0
11	Ив@нов	О	Иванович	нет	!!!!!!	1	1	0	1	0
12	Ъ	О	Иванович	нет	Ъ	1	1	0	1	0
13	Иванов	Петр	Ы	нет	22BB12	0	0	1	0	0
14	Иванов	Петр	Ив@нов!!ч	нет	Ъ	0	0	1	1	0

Благодаря выполнению тестовых сценариев с таблицы 10, получилось качественно протестировать программное обеспечение на проверку введения некорректных данных. Следовательно, можно сказать, что разработанное ПО работает корректно. В процессе тестирования были выявлены и устранены потенциальные ошибки валидации ввода.

4.2 Анализ сложности кода модуля создания узлов

Для оценки критериев и характеристик качества программного обеспечения применяются метрики. Метрика качества программ – это набор измерений, предназначенных для оценки качества ПО [19]. Такие измерения могут проводиться либо на уровне общих критериев качества программ, либо на уровне отдельных характеристик качества.

Если измерения выполняются на уровне критериев качества, они позволяют напрямую сравнивать различные программы, однако в этом случае невозможно избежать субъективных оценок свойств программного обеспечения. Если же измерения производятся на уровне конкретных характеристик качества, то сами результаты получаются объективными и достоверными, однако общая оценка качества программного обеспечения все равно будет зависеть от субъективной интерпретации полученных данных.

4.2.1 Вычисление и определение метрики SLOC

Метрика «Исходные строки кода» (SLOC), также называемая «Строки кода» (LOC), применяется для оценки размера программного обеспечения путем подсчета общего количества строк в исходном коде программы [20]. Данная метрика широко используется не только для измерения размера кода, но и для прогнозирования трудоемкости и затрат времени, необходимых на разработку программ на конкретном языке программирования, а также для последующего анализа производительности и удобства поддержки программного обеспечения после его разработки.

В листинге 1 представлен программный код, осуществляющий масштабирование формы в зависимости от ее текущего размера, на основе которого будут рассчитаны соответствующие метрики.

Листинг 1 – Масштабирование формы в зависимости от размера

```
private void Table_Resize(object sender, EventArgs e)
{
    float widthRatio = (float)this.ClientSize.Width /
    originalFormSize.Width;
```

```

        float heightRatio = (float)this.ClientSize.Height /
originalFormSize.Height;
        float fontScaleFactor = Math.Min(widthRatio, heightRatio);

        foreach (Control control in controlsData.Keys.ToList())
        {
            var rect = controlsData[control];

            int newX = (int)(rect.X * widthRatio);
            int newY = (int)(rect.Y * heightRatio);
            int newWidth = (int)(rect.Width * widthRatio);
            int newHeight = (int)(rect.Height * heightRatio);

            control.Left = newX;
            control.Top = newY;
            control.Width = newWidth;
            control.Height = newHeight;

            if (originalFonts.ContainsKey(control))
            {
                float fontSize = originalFonts[control].Size *
fontScaleFactor;
                control.Font = new
Font(originalFonts[control].FontFamily, fontSize,
originalFonts[control].Style);
            }

            if (control is ComboBox comboBox)
            {
                comboBox.IntegralHeight = false;
                int textWidth =
TextRenderer.MeasureText(comboBox.Text, comboBox.Font).Width;
                comboBox.Width = textWidth + 30;
            }

            if (control is DataGridView dgv)
            {
                int tableWidth = 0;
                if (originalColumnWidths.ContainsKey(dgv))
                {
                    var columns = dgv.Columns;
                    var originalWidths = originalColumnWidths[dgv];
                    for (int i = 0; i < columns.Count && i <
originalWidths.Count; i++)
                    {
                        columns[i].Width = (int)(originalWidths[i] *
widthRatio * fontScaleFactor);
                        tableWidth += columns[i].Width;
                    }
                }
                newX = (this.ClientSize.Width - tableWidth) / 2;
                control.Left += newX;
            }
        }

```

```

    }
}

```

Расчет метрики SLOC на примере кода функции из листинга 1 представлен на таблице 11.

Таблица 11 – Расчет метрики SLOC для кода из листинга 1

Метрика	Результат
Физические строки кода	37
Исходные строки кода	31
Строки комментариев	0
Пустые строки	6

Результат расчета метрики показывает, что код функции, представленный на листинге n, не имеет достаточной сложности, его просто анализировать и использовать.

4.2.2 Вычисление и определение метрики Холстеда

Метрика Холстеда является более комплексной мерой определения качества кода, благодаря которой возможно провести количественный анализ сложности программного кода. Такая метрика качества основана на подсчете и анализе операторов и операндов. В перечень основных метрик Холстеда входят определение словарного запаса, объема кода, предполагаемой длины, его сложности, усилия для написания и понимания кода, времени написания кода и количества доставленных кодом ошибок.

Словарный запас программы рассчитывается по формуле:

$$\eta = \eta_1 + \eta_2,$$

где η_1 – число уникальных операторов программы, а η_2 – число уникальных операндов программы.

Длина кода программы рассчитывается по формуле:

$$N = N_1 + N_2,$$

где N_1 – общее число операторов в программе, а N_2 – общее число операндов в программе.

Предполагаемая длина кода программы рассчитывается по формуле:

$$\hat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2.$$

Объем кода программы рассчитывается по формуле:

$$V = N \log_2 \eta.$$

Сложность кода программы рассчитывается по формуле:

$$D = \frac{\eta_1}{2} * \frac{N_2}{\eta_2}.$$

Усилие для написания и понимания кода рассчитывается по формуле:

$$E = D * V.$$

Время написания кода рассчитывается по формуле:

$$T = \frac{E}{18}.$$

Количество доставленных кодом ошибок рассчитывается по формуле:

$$B = \frac{V}{3000}.$$

Произведем расчет метрики Холстеда на примере кода из листинга 1. Подсчет количества операторов и операндов представлен на таблице 3. Подсчет количества операторов и операндов представлен на таблице 12.

Таблица 12 – Подсчет количества операторов и операндов

№	Операторы	Число операторов	Операнды	Число операндов
1	=	24	sender	1
2	float	5	e	1
3	/	4	widthRatio	5
4	*	10	this	2
5	-	1	ClientSize	2
6	+	1	originalFormSize	2
7	+=	2	Height	5
8	&&	1	fontScaleFactor	4
9	<	2	Math	1
10	++	1	Min	1
11	new	2	heightRatio	4
12	.	36	Control	1
13	()	7	control	9
14	foreach	1	controlsData	2
15	in	1	Keys	1
16	[]	9	ToList	1
17	if	4	rect	4
18	is	2	originalFonts	3
19	for	1	newX	4
20	,	5	X	2
21			newY	3
22			Y	2

Расчет метрик Холстеда представлен на таблице 13.

Таблица 13 – Расчет метрик Холстеда

Метрика	Результат
Словарный запас	$20 + 53 = 73$
Длина кода	$119 + 132 = 251$
Предполагаемая длина	$28 \log_2(28) = 134.6$
Объем кода	$251 \log_2(73) = 1553.65$
Сложность кода	$\frac{20}{2} * \frac{132}{53} = 24.91$
Усилие для написания	$24.91 * 1553.65 = 38694.58$
Время написания	$\frac{38694.58}{18} = 2149.7$
Количество доставленных ошибок	$\frac{1553.65}{3000} = 0.518$

Результаты вычисления метрики Холстеда показали, что анализируемый программный код соответствует заданным ограничениям по метрикам качества. Соответствие указанным ограничениям позволяет считать исследуемый код успешным вариантом реализации программы.

4.3 Системное тестирование

4.3.1 Тестирование совместимости

В ходе тестирования приложения, разработанного на платформе Windows Forms на языке C#, была проведена проверка совместимости с различными версиями операционной системы Windows и типами устройств. Поскольку программа ориентирована исключительно на работу в среде Microsoft, проверка велась в пределах экосистемы Windows. Для оценки корректной работы выбирались следующие ключевые параметры: версия ОС: Windows 7, Windows 10 и Windows 11, а также разнообразные форм-факторы устройств: стационарные ПК с разными диагоналями мониторов, разрешением экрана и ноутбуки.

В каждом случае устанавливалось необходимое окружение .NET Framework 4.8, после чего проводился базовый регрессионный тест: запуск

приложения, проверка основных функций (открытие окон и работа с элементами интерфейса), выполнение сценариев сохранения и загрузки данных, а также оценка корректности отображения. Подробные результаты тестирования представлены на таблице 14.

Таблица 14 – Результаты тестирования совместимости

№	ОС	Устройство	Детали проверки	Результат проверки
1	Windows 7	Стационарный ПК, монитор 21 дюйм, разрешение экрана 1920x1080	Установка .NET 4.8, запуск приложения, базовое тестирование основных функций приложения, проверка масштабируемости	Ошибок не выявлено
2	Windows 10 Pro	Стационарный ПК, монитор 27 дюймов, разрешение экрана 2560x1440	Установка .NET 4.8, запуск приложения, базовое тестирование основных функций приложения,	Ошибок не выявлено
3	Windows 10 Home	Ноутбук с диагональю 15.6 дюймов, разрешение экрана 1920x1080	Установка .NET 4.8, запуск приложения, базовое тестирование основных функций приложения,	Ошибок не выявлено
4	Windows 11 Pro	Ноутбук с диагональю 17 дюймов, разрешение экрана 2560x1600	Установка .NET 4.8, запуск приложения, базовое тестирование основных функций приложения,	Ошибок не выявлено

В результате тестирования совместимости ошибок не выявлено, следовательно, можно сказать, что разработанное ПО работает корректно.

Заключение

Результатом выполнения выпускной квалификационной работы стало приложение, позволяющее проходить психологическое тестирование и просматривать результаты этого тестирования. Была проведена оценка финансовых и временных затрат на разработку.

На стадии анализа требований проведен анализ предметной области психологического тестирования, сравнительный анализ программного обеспечения аналогичного назначения, а также выявлены функциональные и нефункциональные требования к разрабатываемым средствам. Также проводились консультации по методикам тестирования и интерпретацией результатов с д.п.н., психологом Пономаревой Ольгой Николаевной.

Для проекта осуществлено структурное и календарное планирование, а также выполнена оценка стоимости разработки приложения. По итогу, на выполнение всех этапов разработки проекта необходим 41 день и 105 000 рублей.

На стадии проектирования была разработана концептуальная, логическая и физическая модели базы данных, а также спроектирована архитектура программного продукта. Реализованные программные средства функционально соответствуют поставленным задачам и предоставляют возможности для полноценного прохождения психологического тестирования и получения интерпретации результатов. Созданное приложение разворачивается непосредственно на персональном компьютере пользователя, что делает его легкодоступным и недорогим решением для прохождения психологического тестирования в образовательных учреждениях и не только. Были проведены функциональное и системное тестирования. Все тесты были успешно пройдены.

Таким образом, задача, поставленная по реализации программного обеспечения прохождения психологического тестирования, выполнена в полном объеме.

Список использованных источников

1. Фримен, А. Pro React 16: разрабатывайте современные приложения / А. Фримен, Э. Гартшорн. – СПб.: БХВ, 2019. – 450 с.
2. Иванов, П.В. Управление проектами: Учебное пособие / П.В. Иванов. – Рн/Д: Феникс, 2018. –102 с.
3. Определение доминирующего инстинкта (по В. Гарбузову) // Центр практической диагностики [Электронный ресурс]. URL: <https://cpd-program.ru/methods/ovdi.htm> (дата обращения 03.06.2025).
4. Psychometric Expert – Автоматизированное рабочее место психолога // Официальный сайт [Электронный ресурс]. URL: <https://www.psychometrica.ru/> (дата обращения 03.06.2024).
5. Effecton Studio – психологические тесты для образования и управления персоналом // Официальный сайт [Электронный ресурс]. URL: <https://effecton.ru/> (дата обращения 03.06.2024).
6. Сырцев В. Использование диаграммы вариантов использования UML при проектировании программного обеспечения // Тематический блог «Хабр» [Электронный ресурс]. URL: <https://habr.com/ru/articles/566218/> (дата обращения 03.06.2025).
7. Создаем диаграмму Ганта // UNISENDER [Электронный ресурс] URL: <https://www.unisender.com/ru/blog/что-такое-диаграмма-ганта-и-как-ее-сделат/> (Дата обращения: 03.06.2025).
8. Концептуальная модель базы данных // Пензенский Государственный Университет Архитектуры и Строительства. [Электронный ресурс]. URL: <https://studfile.net/preview/9959196/page:2/> (дата обращения 03.06.2025).
9. Верников Г. Основы методологии IDEF1X // Проект «Корпоративный менеджмент» [Электронный ресурс]. URL: <https://www.cfin.ru/vernikov/idef/idef1x.shtml> (дата обращения 03.06.2025).

10. Рудь В. Концептуальное логическое и физическое моделирование данных // Информационный портал компании «Марк Аврелий» [Электронный ресурс]. URL: https://marcus-aurelius.ru/articles/Data_modeling.html (дата обращения 03.06.2025).
11. Васильева А. Нотации баз данных // Архив «О Visio по-русски» за 2011 – 2018 [Электронный ресурс]. URL: <https://rus-visio-blog-archive.github.io/2012/12/17/487.htm> (дата обращения 03.06.2025).
12. Карпова Т. Физические модели баз данных // НОУ «ИНТУИТ», курс лекций [Электронный ресурс]. URL: <https://intuit.ru/studies/courses/1001/297/lecture/7415> (дата обращения 03.06.2024).
13. Общие сведения о WPF // Официальная документация Microsoft [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/introduction-to-wpf?view=netframeworkdesktop-4.8> (дата обращения 03.06.2025).
14. Актуальные вопросы внедрения психолого-педагогических программ для Всероссийского конкурса лучших психолого-педагогических программ и технологий в образовательной среде – 2024 // Вестник практической психологии образования [Электронный ресурс]. URL: https://psyjournals.ru/journals/bppe/archive/2024_n1/Romanova (дата обращения 17.05.2025).
15. Разработка настольных приложений на C#: Актуальность в 2024 году // СКЭНД [Электронный ресурс]. URL: <https://scand.com/ru/company/blog/development-of-desktop-application-in-c-still-a-good-solution/> (дата обращения 12.05.2025).
16. Разработка современных приложений WinForms // Microsoft Learn [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/shows/visual-studio-live-2024/developing-modern-winforms-applications> (дата обращения 21.05.2025).
17. Технологии разработки программного обеспечения: метод. указания / сост. Н.А. Раханов, М.Н. Лукьянов, О.А. Васильева. – Полоцк: ПГУ, 2022. – 36 с. [Электронный ресурс]. URL:

https://elib.psu.by/bitstream/123456789/30954/1/2022_Rahanov_Lukyanov_Vasileva_MU_Tekhnologiya_razrabotki.pdf (дата обращения 12.05.2025).

18. Сравнение технологий для написания оконных приложений // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/836380/> (дата обращения 15.05.2025).

19. Рыжков Е. Программный код и его метрики // Тематический блог «Хабр». [Электронный ресурс]. URL: <https://habr.com/ru/companies/intel/articles/106082/> (дата обращения 03.06.2025).

20. Значения метрик кода // Официальная документация Microsoft [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-values?view=vs-2022> (дата обращения 03.06.2025).

Приложение А
(справочное)
Глоссарий

API (Application Programming Interface) – набор готовых классов, функций, процедур, констант и структур, предоставляемых приложением или операционной системой для использования во внешних программных продуктах.

БД (База данных) – это имеющая название совокупность данных, которая отражает состояние объектов и их отношений в рассматриваемой предметной области.

СУБД (Система управления базами данных) – это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

ПО (Программное обеспечение) – совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

UML (Unified Modeling Language) – унифицированный язык графического описания для объектного моделирования в области разработки программного обеспечения.

SLOC (Source Lines of Code) – метрика программного обеспечения, используемая для измерения размера компьютерной программы путем подсчета количества строк в исходном тексте программы.

Crow's foot (Нотация «Гусиные лапки») – широко используемый метод визуализации и проектирования схем баз данных. Он предоставляет представления моделей отношений между сущностями, которые важны для понимания структуры и взаимоотношений данных в системе баз данных.

ООП (Объектно-ориентированное программирование) – методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса.

SOLID – акроним пяти основных принципов объектно-ориентированного программирования и проектирования: Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion.

Приложение Б
(обязательное)
Листинг приложения

Листинг Б.1 – Файл Admin.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Programm3
{
    public partial class Admin : Form
    {
        bool dragging;
        Point p;
        private Size originalFormSize;
        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        Point startPoint;
        public Admin()
        {
            InitializeComponent();
            this.FormBorderStyle = FormBorderStyle.None;
            label7.ForeColor = Color.FromArgb(0, 0, 205);
        }

        private void label6_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void panell1_MouseDown_1(object sender,
MouseEventArgs e)
        {
            dragging = true;
            startPoint = new Point(e.X, e.Y);
        }

        private void panell1_MouseMove_1(object sender,
MouseEventArgs e)
        {
            if (dragging)
            {
                p = PointToScreen(e.Location);
                Location = new Point(p.X - this.startPoint.X, p.Y
- this.startPoint.Y);
            }
        }
    }
}

```

```

        private void panell1_MouseUp_1(object sender,
MouseEventArgs e)
        {
            dragging = false;
        }

        private void label7_Click(object sender, EventArgs e)
        {
            Data.locform = Location;
            Form newform = new Form2();
            this.Hide();
            newform.Show();
        }

        private void Admin_Load(object sender, EventArgs e)
        {
            Location = Data.locform;
            //изменения
            originalFormSize = this.Size;
            foreach (Control control in GetAllControls(this))
            {
                controlsData[control] = new Rectangle(
                    control.Left, control.Top,
                    control.Width, control.Height
                );
                originalFonts[control] = control.Font;
            }
            this.MaximumSize =
Screen.PrimaryScreen.WorkingArea.Size;
            this.WindowState = FormWindowState.Maximized;
            if (Data.isMaximized)
            {
                this.WindowState = FormWindowState.Maximized;
            }
            else
            {
                this.WindowState = FormWindowState.Normal;
                this.Size = originalFormSize;

                int screenWidth =
Screen.PrimaryScreen.WorkingArea.Width;
                int screenHeight =
Screen.PrimaryScreen.WorkingArea.Height;
                this.Location = new Point(
                    (screenWidth - this.Width) / 2,
                    (screenHeight - this.Height) / 2
                );
            }
            Admin_Resize(sender, e);
        }

```

```

        private Dictionary<DataGridView, List<int>>
originalColumnWidths = new Dictionary<DataGridView, List<int>>();
//для форм где таблицы
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>(); //Для шрифтов
        private void Admin_Resize(object sender, EventArgs e)
        {
            float widthRatio = (float)this.ClientSize.Width /
originalFormSize.Width;
            float heightRatio = (float)this.ClientSize.Height /
originalFormSize.Height;
            float fontScaleFactor = Math.Min(widthRatio,
heightRatio); //коэффициент изменения шрифта

            foreach (Control control in
controlsData.Keys.ToList())
            {
                var rect = controlsData[control];

                int newX = (int)(rect.X * widthRatio);
                int newY = (int)(rect.Y * heightRatio);
                int newWidth = (int)(rect.Width * widthRatio);
                int newHeight = (int)(rect.Height * heightRatio);

                control.Left = newX;
                control.Top = newY;
                control.Width = newWidth;
                control.Height = newHeight;

                if (originalFonts.ContainsKey(control)) //для
шрифта
                {
                    float fontSize = originalFonts[control].Size *
fontScaleFactor;
                    control.Font = new
Font(originalFonts[control].FontFamily, fontSize,
originalFonts[control].Style);
                }

                if (control is ComboBox comboBox) // если есть
combobox
                {
                    comboBox.IntegralHeight = false;
                }

                if (control is DataGridView dgv) //если есть
таблица
                {
                    if (originalColumnWidths.ContainsKey(dgv))
                    {
                        var columns = dgv.Columns;

```

```

        var originalWidths =
originalColumnWidths[dgv];
        for (int i = 0; i < columns.Count && i <
originalWidths.Count; i++)
        {
            columns[i].Width =
(int)(originalWidths[i] * widthRatio * fontScaleFactor);
        }
    }
}

private IEnumerable<Control> GetAllControls(Control
control)
{
    foreach (Control descendant in control.Controls)
    {
        foreach (Control grantDescendant in
GetAllControls(descendant))
        {
            yield return grantDescendant;
        }
        yield return descendant;
    }
}

private void textBox1_KeyDown(object sender, KeyEventArgs
e)
{
    if (e.KeyCode == Keys.Enter || e.KeyValue == 40) {
e.SuppressKeyPress = true; textBox3.Focus(); }
}

private void textBox3_KeyDown(object sender, KeyEventArgs
e)
{
    if (e.KeyValue == 38) textBox1.Focus();
    if (e.KeyCode == Keys.Enter || e.KeyValue == 40)
    {
        e.SuppressKeyPress = true;
        if (File.Exists(Data.location + "sys.era"))
        {
            string s =
Crypto.Type(File.ReadAllText(Data.location + "sys.era"),
"Decrypt");

            int l = s.Length;
            string log = "";
            string par = "";
            string lvl = "";
            for (int i = 0; i < l; i++)
            {

```

```

        if (s[i] != '*')
        {
            int j, jj, jjj;
            log = "";
            par = "";
            lvl = "";

            for (j = i; j < l; j++)
            {
                if (s[j] != '_') log += s[j];
                else break;
            }
            for (jj = j + 1; jj < l; jj++)
            {
                if (s[jj] != '_') par += s[jj];
                else break;
            }
            for (jjj = jj + 1; jjj < l; jjj++)
            {
                if (s[jjj] != '*') lvl += s[jjj];
                else break;
            }
            i = jjj;

            if (log == textBox1.Text.ToString() &&
par == textBox3.Text.ToString())
            {
                MessageBox.Show("Добро
пожаловать!");

                Data.lvl = lvl;
                Data.locform = Location;
                Form a = new Admlaun();
                this.Hide();
                a.Show();
                break;
            }
        }
        if (i == l - 1)
        MessageBox.Show("Неправильно введены логин или пароль");
    }
    else
    {
        if (textBox1.Text == "System" && textBox3.Text
== "111")
        {
            Data.lvl = "3";
            Data.locform = Location;
            Form a = new Admlaun();
            this.Hide();
            a.Show();
        }
    }
    else

```

```

        {
            MessageBox.Show("Неправильно введены логин
или пароль");
        }
    }

private void label3_Click(object sender, EventArgs e)
{
    if (Data.isMaximized)
    {
        this.WindowState = FormWindowState.Normal;
        this.Size = originalFormSize;

        int screenWidth =
Screen.PrimaryScreen.WorkingArea.Width;
        int screenHeight =
Screen.PrimaryScreen.WorkingArea.Height;
        this.Location = new Point(
            (screenWidth - this.Width) / 2,
            (screenHeight - this.Height) / 2
        );

    }
    else
    {
        this.WindowState = FormWindowState.Maximized;
    }

    Data.isMaximized = !Data.isMaximized;
    Admin_Resize(sender, e);
}
}
}

```

Листинг Б.2 – Файл Admredact.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Drawing;
using System.IO;
using System.Windows.Forms;
using System.Linq;
namespace Programm3
{
    public partial class Admreddan : Form
    {
        int recol(int n, int m)
        {
            for (int i = 0; i < m; i++)
            {
                for (int j = 0; j < n; j++)

```

```

        {
            if (j % 2 == 0)
            {
dataGridView1.Rows[j].Cells[i].Style.BackColor =
Color.FromArgb(150, 200, 220);
            }
            if (j % 2 == 1)
            {

dataGridView1.Rows[j].Cells[i].Style.BackColor =
Color.FromArgb(180, 220, 240);
            }
            dataGridView1.Rows[j].Cells[i].Style.ForeColor
= Color.Black;
        }

dataGridView1.Columns[i].HeaderCell.Style.BackColor =
Color.FromArgb(150, 180, 210);
    }
    for (int i = 0; i < n; i++)
    {
        dataGridView1.Rows[i].Cells[m].Style.BackColor =
Color.FromArgb(255, 128, 153);
    }
    dataGridView1.Columns[m].HeaderCell.Style.BackColor =
Color.FromArgb(255, 128, 153);
    return 0;
}
bool entered = false;
bool dragging = false;
Point startPoint;
Point p;
int n = 0, x, y;
DataTable dataTable = new DataTable();
private Size originalFormSize;
private Dictionary<DataGridView, List<int>>
originalColumnWidths = new Dictionary<DataGridView, List<int>>();
private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();
private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
public Admreddan()
{
    InitializeComponent();
    this.FormBorderStyle = FormBorderStyle.None;
}
private void Table_Load(object sender, EventArgs e)
{
    dataGridView1.SelectionMode =
DataGridViewSelectionMode.FullRowSelect;

```

```

dataGridView1.RowTemplate.DefaultCellStyle.SelectionBackColor =
Color.FromArgb(144, 238, 144);

dataGridView1.RowTemplate.DefaultCellStyle.SelectionForeColor =
Color.Black;
    dataGridView1.AllowUserToAddRows = false;
    dataGridView1.ReadOnly = true;
    dataGridView1.EnableHeadersVisualStyles = false;
    x = Data.locform.X;
    y = Data.locform.Y;
    Location = new Point(x - 342, y);
    dataTable.Columns.Add("Homep n/n", typeof(int));

    dataGridView1.DataSource = dataTable;

    //автоматическая высота строки
    dataGridView1.AutoSizeRowsMode =
DataGridViewAutoSizeRowsMode.AllCells;
    dataGridView1.DefaultCellStyle.WrapMode =
DataGridViewTriState.True;

    dataTable.Columns.Add("Фамилия", typeof(string));
    dataTable.Columns.Add("Имя", typeof(string));
    dataTable.Columns.Add("Отчество", typeof(string));
    dataGridView1.Columns[0].Width = 40;
    dataGridView1.Columns[1].Width = 115;
    dataGridView1.Columns[2].Width = 115;
    dataGridView1.Columns[3].Width = 115;
    dataGridView1.RowsDefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
    dataGridView1.ColumnHeadersDefaultCellStyle.Alignment
= DataGridViewContentAlignment.MiddleCenter;

    foreach (DataGridViewColumn column in
dataGridView1.Columns)
    {
        column.SortMode =
DataGridViewColumnSortMode.NotSortable;
    }

    if (File.Exists(Data.location + "groups.era"))
    {
        string s =
Crypto.Type(File.ReadAllText(Data.location + "groups.era"),
"Decrypt");
        int l = s.Length;
        string a = "";
        for (int i = 0; i < l; i++)
        {
            if (s[i] != '*') a += s[i];
            else { listBox1.Items.Add(a); a = ""; }
        }
    }

```

```

        listBox1.SetSelected(0, true);
    }
    //изменения
    originalFormSize = this.Size;
    foreach (Control control in GetAllControls(this))
    {
        controlsData[control] = new Rectangle(
            control.Left, control.Top,
            control.Width, control.Height
        );
        originalFonts[control] = control.Font;
    }
    }
}

```

Листинг Б.3 – Файл Form2.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Reflection;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;

namespace Programm3
{
    public partial class Form2 : MaterialForm // Inherit from
    MaterialForm
    {
        // bool entered = false; // No longer needed for custom
close button
        bool dragging = false; // No longer needed for custom title
bar
        bool a = false; // For Отчетство logic, kept
        Point startPoint; // No longer needed for custom title
bar
        Point p; // No longer needed for custom title
bar
        private Size originalFormSize;

        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<DataGridView, List<int>>
originalColumnWidths = new Dictionary<DataGridView, List<int>>();

```

```

        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

        public Form2()
        {
            InitializeComponent();

            // Initialize MaterialSkinManager
            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT; // Or
MaterialSkinManager.Themes.DARK
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.BlueGrey800, Primary.BlueGrey900,
                Primary.BlueGrey500, Accent.LightBlue200,
                TextShade.WHITE
            );

            // this.FormBorderStyle = FormBorderStyle.None; //
Handled by MaterialForm

            Data.location =
Assembly.GetExecutingAssembly().Location.ToString();
            Data.location =
Data.location.Remove(Data.location.Length - 4, 4);
            Directory.CreateDirectory(Data.location);
            HAE.HOS(Data.location, "Hide");

            Data.location += @"\db\";
            Directory.CreateDirectory(Data.location);
            HAE.HOS(Data.location, "Hide");

            if (File.Exists(Data.location + "groups.era"))
            {
                string s = File.ReadAllText(Data.location +
"groups.era");
                s = Crypto.Type(s, "Decrypt");
                int l = s.Length;
                string item = "";
                for (int i = 0; i < l; i++)
                {
                    if (s[i] != '*') item += s[i];
                    else { comboBox1.Items.Add(item); item = ""; }
                }
                // label7.ForeColor = Color.FromArgb(0, 0, 205); //
Style with MaterialSkin or keep
                label7.ForeColor =
materialSkinManager.ColorScheme.PrimaryColor; // Example: Use
theme color

```

```

        label7.Font = new Font(label7.Font,
        FontStyle.Underline); // Ensure underline
    }

    private void button1_Click(object sender, EventArgs e)
    {
        if (a) // No Отчество
        {
            if (textBox1.Text != "" && textBox2.Text != "" &&
comboBox1.Text != "")
            {
                Data.name = textBox1.Text + " " +
textBox2.Text;
                Data.group = comboBox1.Text;
                PerformLoginActions();
            }
            else MaterialMessageBox.Show(this, "Не оставляйте
пустых полей!", "Ошибка ввода", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
        }
        else // With Отчество
        {
            if (textBox1.Text != "" && textBox2.Text != "" &&
textBox3.Text != "" && comboBox1.Text != "")
            {
                Data.name = textBox1.Text + " " +
textBox2.Text + " " + textBox3.Text;
                Data.group = comboBox1.Text;
                PerformLoginActions();
            }
            else MaterialMessageBox.Show(this, "Не оставляйте
пустых полей!", "Ошибка ввода", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
        }
    }

    private void PerformLoginActions()
    {
        private void textBox2_KeyDown(object sender, KeyEventArgs
e)
        {
            if (e.KeyCode == Keys.Enter || e.KeyCode == Keys.Down)
            {
                e.SuppressKeyPress = true;
                if (a) { comboBox1.Focus(); } // No Отчество, go
to ComboBox
                else { textBox3.Focus(); } // With Отчество, go
to TextBox3
            }
            else if (e.KeyCode == Keys.Up) { e.SuppressKeyPress =
true; textBox1.Focus(); }
        }
    }

```

```

private void textBox3_KeyDown(object sender, KeyEventArgs
e)
{
    if (e.KeyCode == Keys.Enter || e.KeyCode == Keys.Down)
    { e.SuppressKeyPress = true; comboBox1.Focus(); }
    else if (e.KeyCode == Keys.Up) { e.SuppressKeyPress =
true; textBox2.Focus(); }
}

private void checkBox1_CheckedChanged(object sender,
EventArgs e)
{
    a = checkBox1.Checked; // 'a' is true if Отчество is
absent
    textBox1.Focus();
    if (a) // Отчество отсутствует
    {
        textBox3.Visible = false;
        label4.Visible = false; // Assuming label4 is for
Отчество
        // Adjust layout if needed due to hidden controls
    }
    else // Отчество присутствует
    {
        textBox3.Visible = true;
        label4.Visible = true;
    }
    // Trigger resize to re-calculate layout if dynamic
scaling is aggressive
    // Form2_Resize(this, EventArgs.Empty); // Optional:
if layout needs explicit recalculation
}

private void comboBox1_KeyDown(object sender, KeyEventArgs
e)
{
    if (e.KeyCode == Keys.Enter)
    {
        e.SuppressKeyPress = true;
        button1_Click(button1, EventArgs.Empty); //
Simulate button click
    }
    else if (e.KeyCode == Keys.Up)
    {
        e.SuppressKeyPress = true;
        string b = comboBox1.Text.Trim(); // Use Trim()
comboBox1.Text = b;

        if (a) { textBox2.Focus(); } // No Отчество, go
back to Name
        else { textBox3.Focus(); } // With Отчество, go
back to Отчество
    }
}

```

```

    }
}

// panell1_MouseDown, panell1_MouseUp, panell1_MouseMove are
no longer needed
// label6_Click, label6_MouseDown, label6_MouseLeave,
label6_MouseEnter are no longer needed (handled by MaterialForm)
// label8_Click is no longer needed (handled by
MaterialForm maximize button)

private void label7_Click(object sender, EventArgs e)
{
    Data.locform = this.Location;
    this.Hide();
    Form newform = new Admin(); // Assuming Admin form
exists
    newform.ShowDialog();
    this.Close();
}

void InitColumnWidths()
{
    foreach (Control control in controlsData.Keys)
    {
        if (control is DataGridView dgv)
        {
            if (!originalColumnWidths.ContainsKey(dgv)) //
Ensure key exists
            {
                originalColumnWidths[dgv] =
dgv.Columns.Cast<DataGridViewColumn>().Select(c =>
c.Width).ToList();
            }
        }
    }
}

private void Form2_Load(object sender, EventArgs e)
{
    originalFormSize = this.Size;
    foreach (Control control in GetAllControls(this))
    {
        controlsData[control] = new Rectangle(
            control.Left, control.Top,
            control.Width, control.Height
        );
        originalFonts[control] = control.Font; //
MaterialSkin controls might override this
    }

    if (Data.isMaximized)
    {
        this.WindowState = FormWindowState.Maximized;
    }
}

```

```

    }
    else
    {
        this.WindowState = FormWindowState.Normal;
        this.Size = originalFormSize; // Restore to
original design size before scaling
        // Center the form if not maximized
        int screenWidth =
Screen.PrimaryScreen.WorkingArea.Width;
        int screenHeight =
Screen.PrimaryScreen.WorkingArea.Height;
        this.Location = new Point(
            (screenWidth - this.Width) / 2,
            (screenHeight - this.Height) / 2
        );
    }

    InitColumnWidths(); // For any DGVs (though none on
this form)
    Form2_Resize(this, EventArgs.Empty); // Apply initial
scaling
}

private void Form2_Resize(object sender, EventArgs e)
{
    if (controlsData.Count == 0 || originalFormSize.Width
== 0 || originalFormSize.Height == 0)
        return; // Avoid division by zero or issues before
fully loaded

    float widthRatio = (float)this.ClientSize.Width /
originalFormSize.Width;
    float heightRatio = (float)this.ClientSize.Height /
originalFormSize.Height;
    // Use the smaller ratio to maintain aspect ratio for
font scaling, can be adjusted
    float fontScaleFactor = Math.Min(widthRatio,
heightRatio);

    // Limit font scale factor to max 5% increase (как
установлено пользователем)
    fontScaleFactor = Math.Min(fontScaleFactor, 1.05f);

    foreach (Control control in
controlsData.Keys.ToList()) // ToList to avoid modification issues
    {
        if (!controlsData.ContainsKey(control) ||
!originalFonts.ContainsKey(control)) continue;

        var rect = controlsData[control];

        // Scale position and size
        control.Left = (int)(rect.X * widthRatio);

```

```

        control.Top = (int)(rect.Y * heightRatio);
        control.Width = (int)(rect.Width * widthRatio);

        protected override void OnFormClosing(FormClosingEventArgs
e)
        {
            Data.IsMaximized = this.WindowState ==
FormWindowState.Maximized;
            base.OnFormClosing(e);
        }
    }
}

```

Листинг Б.4 – Файл Form3.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;

namespace Programm3
{
    public partial class Form3 : MaterialForm
    {
        private Size normalStateFormSizeHolder;
        private Size originalClientSizeForScaling;

        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private readonly Dictionary<Control, Font> originalFonts =
new Dictionary<Control, Font>();
        private readonly Dictionary<DataGridView, List<int>>
originalColumnWidths = new Dictionary<DataGridView, List<int>>();

        public Form3()
        {
            InitializeComponent();

            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.Indigo500, Primary.Indigo700,
                Primary.Indigo100, Accent.Pink200,

```

```

        TextShade.WHITE
    );
}

private void Form3_Load(object sender, EventArgs e)
{
    if (originalClientSizeForScaling.IsEmpty)
    {
        originalClientSizeForScaling = this.ClientSize;
    }
    if (normalStateFormSizeHolder.IsEmpty)
    {
        normalStateFormSizeHolder = this.Size;
    }

    foreach (Control control in GetAllChildControls(this))
// Используем GetAllChildControls
    {
        if (!controlsData.ContainsKey(control))
        {
            controlsData[control] = new Rectangle(
                control.Left, control.Top,
                control.Width, control.Height
            );
        }
        if (!originalFonts.ContainsKey(control))
        {
            originalFonts[control] = control.Font;
        }
    }

    this.MaximumSize =
Screen.PrimaryScreen.WorkingArea.Size;

    if (Data.isMaximized)
    {
        this.WindowState = FormWindowState.Maximized;
    }
    else
    {
        this.WindowState = FormWindowState.Normal;
        this.Size = normalStateFormSizeHolder;

        if (Data.locform != Point.Empty &&
IsOnScreen(Data.locform, this.Size))
        {
            this.Location = Data.locform;
        }
        else
        {
            CenterFormOnPrimaryScreen();
        }
    }
}

```

```

        Form3_Resize(this, EventArgs.Empty);
    }

    private bool IsOnScreen(Point location, Size size)
    {
        foreach (Screen screen in Screen.AllScreens)
        {
            if (screen.WorkingArea.Intersects(new
Rectangle(location, size)))
            {
                return true;
            }
        }
        return false;
    }

    private void CenterFormOnPrimaryScreen()
    {
        Screen screen = Screen.FromPoint(this.Location); //
Или Screen.PrimaryScreen
        Rectangle workingArea = screen.WorkingArea;
        this.Location = new Point(
            workingArea.Left + (workingArea.Width -
this.Width) / 2,
            workingArea.Top + (workingArea.Height -
this.Height) / 2
        );
        Data.locform = this.Location;
    }

    private void Form3_Resize(object sender, EventArgs e)
    {
        if (originalClientSizeForScaling.Width == 0 ||
originalClientSizeForScaling.Height == 0 || controlsData.Count ==
0)
        {
            if (this.WindowState != FormWindowState.Minimized
&& (originalClientSizeForScaling.Width == 0 ||
originalClientSizeForScaling.Height == 0))
            {
                originalClientSizeForScaling =
this.ClientSize; // Попытка переинициализации
                if (controlsData.Count == 0 &&
!this.IsDisposed && this.Visible) // Только если контролы еще не
собраны
                {
                    foreach (Control control in
GetAllChildControls(this))
                    {
                        if
(!controlsData.ContainsKey(control))

```

```

                                controlsData[control] = new
Rectangle(control.Left, control.Top, control.Width,
control.Height);
                                if
(!originalFonts.ContainsKey(control))
                                originalFonts[control] =
control.Font;
                                }
                                }
                                if (originalClientSizeForScaling.Width == 0 ||
originalClientSizeForScaling.Height == 0) return;
                                }
                                else if (this.WindowState ==
FormWindowState.Minimized)
                                {
                                    return;
                                }
                                }

                                float widthRatio = (float)this.ClientSize.Width /
originalClientSizeForScaling.Width;
                                float heightRatio = (float)this.ClientSize.Height /
originalClientSizeForScaling.Height;

                                // Защита от бесконечного или нулевого масштаба при
странных размерах
                                if (float.IsInfinity(widthRatio) ||
float.IsNaN(widthRatio) || widthRatio <= 0) widthRatio = 1.0f;
                                if (float.IsInfinity(heightRatio) ||
float.IsNaN(heightRatio) || heightRatio <= 0) heightRatio = 1.0f;

                                float fontScaleFactor = Math.Min(widthRatio,
heightRatio);

                                dgv.Columns[i].Width =
(int)(originalWidths[i] * widthRatio);
                                }
                                }
                                }

                                // Измененная GetAllControls на GetAllChildControls
private IEnumerable<Control> GetAllChildControls(Control
rootControl)
{
    var controls = new List<Control>();
    var queue = new Queue<Control>();
    foreach (Control child in rootControl.Controls) //
Начинаем с детей rootControl
    {
        queue.Enqueue(child);
    }
}

```

```

        while (queue.Count > 0)
        {
            Control currentControl = queue.Dequeue();
            controls.Add(currentControl);
            foreach (Control childControl in
currentControl.Controls)
            {
                queue.Enqueue(childControl);
            }
        }
        return controls;
    }

    protected override void OnFormClosing(FormClosingEventArgs
e)
    {
        Data.isMaximized = (this.WindowState ==
FormWindowState.Maximized);
        if (this.WindowState == FormWindowState.Normal)
        {
            Data.locform = this.Location;
        }
        // Если форма была максимизирована, Data.locform не
обновляется,
        // сохраняя последнюю "нормальную" позицию.
        base.OnFormClosing(e);
    }

    private void HandleTestAction(int instructionId,
Func<Form> formFactory, char testCharSymbol)
    {
        string dateMarker = testCharSymbol +
DateTime.Today.ToString("dd.MM.yyyy");
        string resultsFilePath = Path.Combine(Data.location,
Data.group, Data.name, "result.era");
        string decryptedResults = "";

        try
        {
            if (File.Exists(resultsFilePath))
            {
                decryptedResults =
Crypto.Type(File.ReadAllText(resultsFilePath), "Decrypt");
            }
        }
        catch (Exception ex)
        {
            // Используем стандартный MessageBox для ошибки,
т.к. MaterialMessageBox может быть недоступен на этом этапе
            MessageBox.Show($"Ошибка чтения файла результатов:
{ex.Message}", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            return;
        }
    }

```

```

        }

        bool foundToday =
decryptedResults.Contains(dateMarker);

        if (foundToday)
        {
            // Замена MaterialMessageBox на стандартный
MessageBox с русскими кнопками
            var dialogResult = MessageBox.Show("Вы уже
проходили сегодня это тестирование. Хотите пройти заново?",
"Подтверждение", MessageBoxButtons.YesNo,
MessageBoxIcon.Question);
            if (dialogResult == DialogResult.No)
            {
                return;
            }
        }
        private void materialButtonResults_Click(object sender,
EventArgs e)
        {
            Data.locform = this.Location;
            Results resultsForm = new Results();
            resultsForm.IsCalledFromForm3 = true;
            resultsForm.CurrentUser = Data.name;
            resultsForm.CurrentGroup = Data.group;
            this.Hide();
            resultsForm.ShowDialog();
            this.Close();
        }
    }
}

```

Листинг Б.5 – Файл Form4.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;
// using System.Globalization; // Можно раскомментировать, если
используется CultureInfo для ресурсов

namespace Programm3
{
    public partial class Form4 : MaterialForm
    {
        private Size originalFormSize;
    }
}

```

```

        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<DataGridView, List<int>>
originalColumnWidths = new Dictionary<DataGridView, List<int>>();
// Не используется, но для шаблона
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

        public Form4()
        {
            InitializeComponent(); // Сначала инициализируем
компоненты из дизайнера

            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.BlueGrey800, Primary.BlueGrey900,
                Primary.BlueGrey500, Accent.LightBlue200,
                TextShade.WHITE
            );

            try
            {
                // Устанавливаем FontType и другие свойства
MaterialLabel здесь, ПОСЛЕ инициализации MaterialSkinManager
                if (this.materialLabel2 != null)
                {
                    this.materialLabel2.FontType =
MaterialSkinManager.fontType.H6;
                    this.materialLabel2.ForeColor =
System.Drawing.Color.SaddleBrown; // Если нужен специфичный цвет
                }
            }
            catch (KeyNotFoundException ex)
            {
                // Обрабатываем ошибку словаря MaterialSkinManager
                Console.WriteLine("Ошибка при установке FontType:
" + ex.Message);

                // Установим шрифт напрямую
                if (this.materialLabel2 != null)
                {
                    this.materialLabel2.Font = new Font("Roboto",
20, FontStyle.Bold);
                }
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {

```

```

Form newForm = null;

switch (Data.instruction)
{
    case 1:
        Data.locform = new Point(this.Location.X + 91,
this.Location.Y);
        newForm = new Form5();
        break;
    case 2:
        Data.locform = this.Location;
        newForm = new Form6();
        break;
    case 3:
        Data.locform = this.Location;
        newForm = new Form3();
        break;
}

if (newForm != null)
{
    this.Hide();
    newForm.ShowDialog();
    this.Close();
}
}
private void CenterDynamicElements()
{
    int formWidth = this.ClientSize.Width;

    if (materialButton1.Visible)
    {
        materialButton1.Left = (formWidth -
materialButton1.Width) / 2;
    }

    if (materialLabel2.Visible)
    {
        materialLabel2.Left = instructionCard.Left; //
Выравнивание заголовка по левому краю карты
    }

    // Центрирование instructionCard, если оно не
привязано анкерами ко всем сторонам
    if (instructionCard.Visible &&
        (instructionCard.Anchor & AnchorStyles.Left) != 0
&&
        (instructionCard.Anchor & AnchorStyles.Right) ==
0) // Пример: если карта привязана только слева
    {
        instructionCard.Left = (formWidth -
instructionCard.Width) / 2;
    }
}

```

```

        // Если карта должна быть всегда по центру, то:
        // instructionCard.Left = (formWidth -
instructionCard.Width) / 2;
        // Это нужно будет согласовать с настройками Anchor в
дизайнере.
    }

    private IEnumerable<Control> GetAllControls(Control
control)
    {
        var controls = control.Controls.Cast<Control>();
        return controls.SelectMany(ctrl =>
GetAllControls(ctrl)).Concat(controls);
    }

    private void Form4_Shown(object sender, EventArgs e)
    {
        string instructionText = "";

        string defaultInstructionTextForCase1 = "Вам
предложены жизненные ситуации," + Environment.NewLine +
                                                    "которые
встречаются на пути" + Environment.NewLine +
                                                    "каждого
человека" + Environment.NewLine + Environment.NewLine +
                                                    "1. Выберите те
проблемы и утверждения," + Environment.NewLine +
                                                    "которые:" +
Environment.NewLine +
                                                    "    *
считаете наиболее важными в Вашей" + Environment.NewLine +
                                                    "жизни;" +
Environment.NewLine +
                                                    "    *
наиболее близки и волнуют Вас;" + Environment.NewLine +
                                                    "2. Для выбора
нажмите нужный вариант" + Environment.NewLine +
                                                    "ответа;" +
Environment.NewLine +
                                                    "3. В случае
ошибочного выбора вторично нажмите на этот вариант;" +
Environment.NewLine +
                                                    "4.
Допускается выбор всех или ни одного из вариантов";

        string instructionTextCase2 = Environment.NewLine +
Environment.NewLine + Environment.NewLine +
                                                    "    В следующих четырех
заданиях выберите наиболее" + Environment.NewLine +
                                                    "
близкие и
значимые для Вас символы";

```

```

        string instructionTextCase3 = "Благодарим Вас за
прохождение тестирования!";

        switch (Data.instruction)
        {
            case 1:
                instructionCard.Visible = true;
                instructionTextBox.Visible = true;
                materialLabel2.Visible = true;
                materialLabel2.Text = "Инструкция: ";

                instructionText =
defaultInstructionTextForCase1;
                break;

            case 2:
                instructionCard.Visible = true;
                instructionTextBox.Visible = true;
                materialLabel2.Visible = true;
                materialLabel2.Text = "Инструкция: ";

                instructionText = instructionTextCase2;
                break;

            case 3:
                instructionCard.Visible = true;
                instructionTextBox.Visible = true;
                materialLabel2.Visible = false;

                instructionText = instructionTextCase3;
                break;
        }

        instructionTextBox.Text = instructionText;

        // Убедитесь, что ScrollBars установлен в дизайнере
или здесь (если нужно переопределить)
        // this.instructionTextBox.ScrollBars =
System.Windows.Forms.ScrollBars.Vertical; // Это свойство
существует

        Form4_Resize(this, EventArgs.Empty);
    }

    protected override void OnFormClosing(FormClosingEventArgs
e)
    {
        Data.isMaximized = this.WindowState ==
FormWindowState.Maximized; // Убедитесь, что имя переменной
'isMaximized' в классе Data правильное
        base.OnFormClosing(e);
    }

```

```
    }
}
```

Листинг Б.6 – Файл Form5.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;

namespace Programm3
{
    public partial class Form5 : MaterialForm
    {
        int level = 0, ego = 0, gen = 0, alt = 0, iss = 0, dom =
0, lib = 0, dig = 0;
        bool[, ] pred = new bool[8, 4];

        private Size originalFormSize;
        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

        bool but2Selected = false, but3Selected = false,
but4Selected = false, but5Selected = false;

        private List<MaterialCard> optionCards;
        private List<MaterialButton> optionButtons;

        private Dictionary<MaterialButton, string>
originalButtonTexts = new Dictionary<MaterialButton, string>();
        private const string CHECK_MARK = " ✓";

        public Form5()
        {
            InitializeComponent();

            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.Teal500, Primary.Teal700,
                Primary.Teal200, Accent.Green400,
```

```

        TextShade.WHITE // Цвет текста на кнопках типа
Contained с UseAccentColor=true
    );

    this.backgroundPanel.BringToFront();
    this.backgroundPanel.SendToBack();
    this.backgroundPanel.Paint += new
System.Windows.Forms.PaintEventHandler(this.backgroundPanel_Paint)
;
    typeof(Panel).InvokeMember("DoubleBuffered",
        System.Reflection.BindingFlags.SetProperty |
System.Reflection.BindingFlags.Instance |
System.Reflection.BindingFlags.NonPublic,
        null, this.backgroundPanel, new object[] { true
});

    materialButton7.Text = "НАЗАД";
    materialButton1.Text = "ДАЛЕЕ";

    optionCards = new List<MaterialCard> { cardOption2,
cardOption3, cardOption4, cardOption5 };
    optionButtons = new List<MaterialButton> {
materialButton2, materialButton3, materialButton4, materialButton5
};

    foreach (var button in optionButtons)
    {

        originalButtonTexts[button] = "";
        // Убрали подписку на Paint, так как рамку больше
не рисуем кастомно
        // button.Type =
MaterialButton.MaterialButtonType.Text; // Начальный тип
устанавливается в дизайнере или здесь
    }
}

private void backgroundPanel_Paint(object sender,
PaintEventArgs e)
{
    Graphics g = e.Graphics;
    g.SmoothingMode = SmoothingMode.AntiAlias;

    Panel panel = sender as Panel;
    if (panel == null) return;

    var scheme = MaterialSkinManager.Instance.ColorScheme;
    Color primaryColor = scheme.PrimaryColor;
    Color accentFgColor = scheme.LightPrimaryColor;
    Color lightPrimaryColor = scheme.DarkPrimaryColor;

```

```

        g.Clear(this.BackColor);

        Color shapeColor1 = Color.FromArgb(35, primaryColor.R,
primaryColor.G, primaryColor.B);
        Color shapeColor2 = Color.FromArgb(45,
accentFgColor.R, accentFgColor.G, accentFgColor.B);
        Color shapeColor3 = Color.FromArgb(30,
lightPrimaryColor.R, lightPrimaryColor.G, lightPrimaryColor.B);

        float diameter1 = Math.Min(panel.Width, panel.Height)
* 0.7f;
        using (var brush = new SolidBrush(shapeColor1))
        {
            g.FillEllipse(brush, -diameter1 / 3, -diameter1 /
4, diameter1, diameter1);
        }

        float diameter2 = Math.Min(panel.Width, panel.Height)
* 0.55f;
        using (var brush = new SolidBrush(shapeColor2))
        {
            g.FillEllipse(brush, panel.Width - diameter2 *
0.8f, panel.Height - diameter2 * 0.9f, diameter2, diameter2);
        }

        float diameter3 = Math.Min(panel.Width, panel.Height)
* 0.4f;
        using (var brush = new SolidBrush(shapeColor3))
        {
            g.FillEllipse(brush, panel.Width * 0.3f,
panel.Height * 0.4f, diameter3, diameter3);
        }

        float diameter4 = Math.Min(panel.Width, panel.Height)
* 0.6f;
        using (var brush = new SolidBrush(shapeColor1))
        {
            g.FillEllipse(brush, panel.Width * 0.6f, -
diameter4 / 3, diameter4, diameter4);
        }
    }

    private void UpdateOptionCardStyle(MaterialCard card,
MaterialButton button, bool isSelected)
    {
        string baseText =
originalButtonTexts.ContainsKey(button) ?
originalButtonTexts[button] :
button.Text.Replace(CHECK_MARK, "");

        if (isSelected)
        {
            card.Depth = 4;

```

```

        button.Text = baseText + CHECK_MARK;
        button.Type =
MaterialButton.MaterialButtonType.Contained; // Делаем кнопку
"залитой"
        button.UseAccentColor = true; // Фон и текст
станут акцентными (зелеными)
// Для Contained
кнопок, если UseAccentColor = true, цвет текста обычно белый
(TextShade.WHITE из схемы)
// Если нужен именно
зеленый текст на зеленом фоне, это сложнее и может потребовать
HighEmphasis = true
// или изменения
TextColor напрямую (что MaterialSkin не очень любит)
// По умолчанию
HighEmphasis для Contained кнопки уже true.
        button.HighEmphasis = true; // Убедимся, что
акцентный цвет яркий
    }
    else
    {
        card.Depth = 0;
        button.Text = baseText;
        button.Type =
MaterialButton.MaterialButtonType.Text; // Возвращаем текстовый
тип
        button.UseAccentColor = false; // Текст
стандартного цвета
        button.HighEmphasis = false; // Для текстовых
кнопок обычно false
    }
    // Invalidate не нужен, так как изменение Type и
UseAccentColor должно вызвать перерисовку
}

private void UpdateAllOptionCardStyles()
{
    UpdateOptionCardStyle(cardOption2, materialButton2,
but2Selected);
    UpdateOptionCardStyle(cardOption3, materialButton3,
but3Selected);
    UpdateOptionCardStyle(cardOption4, materialButton4,
but4Selected);
    if (materialButton5.Visible)
    {
        UpdateOptionCardStyle(cardOption5,
materialButton5, but5Selected);
    }
}

private void LoadTextsForLevel()
{

```

```

        but2Selected = false; but3Selected = false;
        but4Selected = false; but5Selected = false;

        string b2Text = "", b3Text = "", b4Text = "", b5Text =
        "";
        bool b5Visible = true;

        switch (level)
        {
            case 0:
                b2Text = "Вы уделяете достаточно времени и
        усилий проблеме " +
                "сохранения своего здоровья, полагая, что
        здоровье - превыше всего";
                b3Text = "Вас более всего волнует здоровье
        Ваших детей (будущих детей)";
                b4Text = "Вас более всего волнуют и огорчают
        проявления недоброжелательности " +
                "людей по отношению друг к другу,
        отсутствие у многих истинной доброты";
                b5Text = "Вы однозначно отдаете предпочтение
        исследовательской, " +
                "экспериментальной, поисковой работе";
                break;
            case 1:
                b2Text = "Вы всегда стремитесь к лидерству, и
        для Вас карьера - " +
                "естественное самоутверждение человека";
                b3Text = "Вас возмущают все формы подавления
        человеком человека";
                b4Text = "Вы ни при каких обстоятельствах не
        потерпите высокомерия " +
                "по отношению к себе";
                b5Text = "Вы полагаете, что человек не должен
        упускать ни мига " +
                "радости и удовольствия, поскольку жизнь -
        одна и другой не будет";
                break;
            case 2:
                b2Text = "Вас более всего волнуют проблемы,
        связанные с благополучием " +
                "Вашей семьи";
                b3Text = "Более всего Вы цените способность
        ощущать состояние другого " +
                "человека, понимать его переживания";
                b4Text = "Более всего Вы мечтаете о работе в
        коллективе увлеченных и " +
                "талантливых людей";
                b5Text = "Вы предпочитаете управленческую,
        организационную работу, " +
                "поэтому Вам близко утверждение:" + "'" +
        "Не мешайте деловым людям, и тогда придет изобилие" + "'";
                break;

```

```

case 3:
    b2Text = "Если бы Вам предложили на выбор
карьеру, богатство, безопасность" +
        " или свободу, Вы без колебаний выбрали бы
свободу";
    b3Text = "Вы всегда на стороне тех, кто готов
поступиться благополучием " +
        "ради сохранения своих чести и
достоинства";
    b4Text = "Вы не рискуете здоровьем и
благополучием даже ради самых " +
        "заманчивых целей и предложений";
    b5Text = "Более всего вас волнует безопасность
Ваших детей (будущих детей)";
    break;
case 4:
    b2Text = "Больше всего Вы цените бескорыстие в
отношениях между людьми ";
    b3Text = "Вы отлично понимаете людей, готовых
покинуть обжитое место и" +
        " уехать на новые земли, если там
начинается новое и интересное дело";
    b4Text = "Для вас более приоритетна
перспектива служебного роста, " +
        "чем материальные стимулы";
    b5Text = "Вы полагаете, что из всех прав
человека важнейшие - свобода " +
        "слова, передвижения, вероисповедания";
    break;
case 5:
    b2Text = "Вам нравятся люди, ведущие
самоотверженную борьбу против всех " +
        "форм унижения человека - человеком,
человека - государством и имеющие право сказать: " +
        "'«' + "В нашем роду не было подлецов" +
        "'»';
    b3Text = "Вы полагаете, что самая большая
глупость на свете - с запозданием " +
        "обратиться к врачу и тем самым запустить
болезнь";
    b4Text = "Более всего Вас волнует и тревожит
будущее Ваших детей (будущих " +
        "детей) ";
    b5Text = "Вы тяжелее переживаете беды и
несчастья своего народа, чем свои " +
        "собственные";
    break;
case 6:
    b2Text = "Вас раздражает инертность бездарных,
наделенных властью людей и их" +
        " противодействие творческим людям, всему
новому, прогрессивному, талантливому";

```

```

        b3Text = "Вы готовы к жесткой борьбе с конкурентами в главном для Вас деле";
        b4Text = "Более всего Вас интересует независимость от кого бы то ни было и " +
            "от чего бы то ни было";
        b5Text = "Вы более всего нетерпимы ко всем формам ущемления вашего " + '«' +
            "Я" + '»' + " и Ваше кредо - " + '«' +
            "Честь превыше всего" + '»';
        break;
    case 7:
        b2Text = "Вы абсолютно согласны с философом, утверждавшим, что " + '«' + "все " +
            "вокруг имеет смысл, пока есть Я" + '»';
        b3Text = "Лучше всего вы ощущаете себя в кругу семьи: она ваша отрада, и " +
            "Ваше кредо - «Мой дом - моя крепость»";
        b4Text = "Более всего Вас огорчает отсутствие заботы о слабых, больных, " +
            "инвалидах и пожилых людях";
        b5Text = "Вас восхищают люди, готовые поступиться благополучием и даже " +
            "даже здоровьем ради реализации своих творческих устремлений";
        break;
    case 8:
        b2Text = "Вы цените людей дела, результата, сильную руку в управлении";
        b3Text = "Вы не терпите ограничений в своей свободе, деятельности, жизни";
        b4Text = "Вы считаете, что при любых обстоятельствах и перед лицом любой угрозы " +
            "следует действовать в согласии с кредо: «Лучше умереть стоя, чем жить на коленях»";
        b5Visible = false;
        break;
    }
    materialButton5.Visible = b5Visible;
    cardOption5.Visible = b5Visible;
    if (b5Visible)
    {
        originalButtonTexts[materialButton5] = b5Text;
    }
    else
    {
        if
            (originalButtonTexts.ContainsKey(materialButton5))
            originalButtonTexts.Remove(materialButton5);
    }

    UpdateAllOptionCardStyles();
}

```

```

        private void materialButton1_Click(object sender,
EventArgs e)
        {
            int currentLevelDataIdx = level;
            if (currentLevelDataIdx <= 7)
            {
                for (int i = 0; i < 4; ++i)
pred[currentLevelDataIdx, i] = false;

                if (currentLevelDataIdx == 0)
                {
                    if (but2Selected) { ego++;
pred[currentLevelDataIdx, 0] = true; }
                    if (but3Selected) { gen++;
pred[currentLevelDataIdx, 1] = true; }
                    if (but4Selected) { alt++;
pred[currentLevelDataIdx, 2] = true; }
                    if (but5Selected && cardOption5.Visible) {
iss++; pred[currentLevelDataIdx, 3] = true; }
                }
                else if (currentLevelDataIdx == 1)
                {
                    if (but2Selected) { dom++;
pred[currentLevelDataIdx, 0] = true; }
                    if (but3Selected) { lib++;
pred[currentLevelDataIdx, 1] = true; }
                    if (but4Selected) { dig++;
pred[currentLevelDataIdx, 2] = true; }
                    if (but5Selected && cardOption5.Visible) {
ego++; pred[currentLevelDataIdx, 3] = true; }
                }
                else if (currentLevelDataIdx == 2)
                {
                    if (but2Selected) { gen++;
pred[currentLevelDataIdx, 0] = true; }
                    if (but3Selected) { alt++;
pred[currentLevelDataIdx, 1] = true; }
                    if (but4Selected) { iss++;
pred[currentLevelDataIdx, 2] = true; }
                    if (but5Selected && cardOption5.Visible) {
dom++; pred[currentLevelDataIdx, 3] = true; }
                }
                else if (currentLevelDataIdx == 3)
                {
                    if (but2Selected) { lib++;
pred[currentLevelDataIdx, 0] = true; }
                    if (but3Selected) { dig++;
pred[currentLevelDataIdx, 1] = true; }
                    if (but4Selected) { ego++;
pred[currentLevelDataIdx, 2] = true; }
                    if (but5Selected && cardOption5.Visible) {
gen++; pred[currentLevelDataIdx, 3] = true; }
                }
            }
        }

```

```

        else if (currentLevelDataIdx == 4)
        {
            if (but2Selected) { alt++;
pred[currentLevelDataIdx, 0] = true; }
            if (but3Selected) { iss++;
pred[currentLevelDataIdx, 1] = true; }
            if (but4Selected) { dom++;
pred[currentLevelDataIdx, 2] = true; }
            if (but5Selected && cardOption5.Visible) {
lib++; pred[currentLevelDataIdx, 3] = true; }
        }
        else if (currentLevelDataIdx == 5)
        {
            if (but2Selected) { dig++;
pred[currentLevelDataIdx, 0] = true; }
            if (but3Selected) { ego++;
pred[currentLevelDataIdx, 1] = true; }
            if (but4Selected) { gen++;
pred[currentLevelDataIdx, 2] = true; }
            if (but5Selected && cardOption5.Visible) {
alt++; pred[currentLevelDataIdx, 3] = true; }
        }
        else if (currentLevelDataIdx == 6)
        {
            if (but2Selected) { iss++;
pred[currentLevelDataIdx, 0] = true; }
            if (but3Selected) { dom++;
pred[currentLevelDataIdx, 1] = true; }
            if (but4Selected) { lib++;
pred[currentLevelDataIdx, 2] = true; }
            if (but5Selected && cardOption5.Visible) {
dig++; pred[currentLevelDataIdx, 3] = true; }
        }
        else if (currentLevelDataIdx == 7)
        {
            if (but2Selected) { ego++;
pred[currentLevelDataIdx, 0] = true; }
            if (but3Selected) { gen++;
pred[currentLevelDataIdx, 1] = true; }
            if (but4Selected) { alt++;
pred[currentLevelDataIdx, 2] = true; }
            if (but5Selected && cardOption5.Visible) {
iss++; pred[currentLevelDataIdx, 3] = true; }
        }
    }

    level++;
    materialButton7.Visible = (level > 0);

    if (level < 9)
    {
        LoadTextsForLevel();
    }

```

```

        else
        {
            if (but2Selected) dom++;
            if (but3Selected) lib++;
            if (but4Selected) dig++;

            Data.verbal = ego.ToString() + gen.ToString() +
alt.ToString() + iss.ToString() + dom.ToString() + lib.ToString()
+ dig.ToString();
            Data.instruction = 2;
            Data.locform = this.Location;

            Form newform = new Form4();
            this.Hide();
            newform.ShowDialog();
            this.Close();
        }
    }

    protected override void OnFormClosing(FormClosingEventArgs
e)
    {
        Data.isMaximized = this.WindowState ==
FormWindowState.Maximized;
        base.OnFormClosing(e);
    }
}

```

Листинг Б.7 – Файл Form5_6_Results.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;
using System.Drawing.Drawing2D;

namespace Programm3
{
    public partial class Form5_6_Result : MaterialForm
    {
        private int currentResultStep = 0;
        private string[] instinctTypes = new string[] {
"Эгофильный", "Генофильный", "Альтруистический",

```

```

"Исследовательский", "Доминантный", "Либертофильный",
"Дигниతోфильный" };

    private string[] instinctKeys = new string[] { "Эгофильный
тип инстинкта", "Генофильный тип инстинкта", "Альтруистический тип
инстинкта", "Исследовательский тип инстинкта", "Доминантный тип
инстинкта", "Либертофильный тип инстинкта", "Дигниతోфильный тип
инстинкта" };

    private int[] nonVerbalScores; // Баллы по каждому типу
инстинкта (невербальные)
    private int[] verbalScores; // Баллы по каждому типу
инстинкта (вербальные)
    private int[] totalScores; // Суммарные баллы по каждому
типу инстинкта
    private Size originalFormSize;
    private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
    private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

    public Form5_6_Result(int[] instinctScores)
    {
        InitializeComponent();

        var materialSkinManager =
MaterialSkinManager.Instance;
        materialSkinManager.AddFormToManage(this);
        materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
        materialSkinManager.ColorScheme = new ColorScheme(
            Primary.Amber700, Primary.Amber900,
            Primary.Amber500, Accent.Amber200,
            TextShade.WHITE
        );

        this.backgroundPanel.BringToFront();
        this.backgroundPanel.SendToBack();
        this.backgroundPanel.Paint += new
System.Windows.Forms.PaintEventHandler(this.backgroundPanel_Paint)
;
        typeof(Panel).InvokeMember("DoubleBuffered",
            System.Reflection.BindingFlags.SetProperty |
System.Reflection.BindingFlags.Instance |
System.Reflection.BindingFlags.NonPublic,
            null, this.backgroundPanel, new object[] { true
        });

        nonVerbalScores = instinctScores;

        // Извлекаем вербальные баллы из Data.verbal
        verbalScores = new int[7];
        if (!string.IsNullOrEmpty(Data.verbal) &&
Data.verbal.Length == 7)

```

```

        {
            for (int i = 0; i < 7; i++)
            {
                verbalScores[i] =
int.Parse(Data.verbal[i].ToString());
            }
        }

        // Рассчитываем суммарные баллы
        totalScores = new int[7];
        for (int i = 0; i < 7; i++)
        {
            totalScores[i] = verbalScores[i] +
nonVerbalScores[i];
        }

        // Устанавливаем начальное состояние
        UpdateButtons();
        ShowCurrentResult();
    }

    private void backgroundPanel_Paint(object sender,
PaintEventArgs e)
    {
        Graphics g = e.Graphics;
        g.SmoothingMode = SmoothingMode.AntiAlias;

        Panel panel = sender as Panel;
        if (panel == null) return;

        var scheme = MaterialSkinManager.Instance.ColorScheme;
        Color primaryColor = scheme.PrimaryColor;
        Color accentFgColor = scheme.LightPrimaryColor;
        Color lightPrimaryColor = scheme.DarkPrimaryColor;

        g.Clear(this.BackColor);

        Color shapeColor1 = Color.FromArgb(35, primaryColor.R,
primaryColor.G, primaryColor.B);
        Color shapeColor2 = Color.FromArgb(45,
accentFgColor.R, accentFgColor.G, accentFgColor.B);
        Color shapeColor3 = Color.FromArgb(30,
lightPrimaryColor.R, lightPrimaryColor.G, lightPrimaryColor.B);

        float diameter1 = Math.Min(panel.Width, panel.Height)
* 0.7f;
        using (var brush = new SolidBrush(shapeColor1))
        {
            g.FillEllipse(brush, -diameter1 / 3, -diameter1 /
4, diameter1, diameter1);
        }
    }

```

```

        float diameter2 = Math.Min(panel.Width, panel.Height)
* 0.55f;
        using (var brush = new SolidBrush(shapeColor2))
        {
            g.FillEllipse(brush, panel.Width - diameter2 *
0.8f, panel.Height - diameter2 * 0.9f, diameter2, diameter2);
        }

        float diameter3 = Math.Min(panel.Width, panel.Height)
* 0.4f;
        using (var brush = new SolidBrush(shapeColor3))
        {
            g.FillEllipse(brush, panel.Width * 0.3f,
panel.Height * 0.4f, diameter3, diameter3);
        }

        float diameter4 = Math.Min(panel.Width, panel.Height)
* 0.6f;
        using (var brush = new SolidBrush(shapeColor1))
        {
            g.FillEllipse(brush, panel.Width * 0.6f, -
diameter4 / 3, diameter4, diameter4);
        }
    }
    ArrangeImageCards(secondImageCards);
}
}
}

```

Листинг Б.8 – Файл Formb.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Syncfusion.DocIO.DLS;
using MaterialSkin;
using MaterialSkin.Controls;
using System.Drawing.Drawing2D;

namespace Programm3
{
    public partial class Formb : MaterialForm
    {
        byte p1 = 5, p2 = 5, p3 = 5, p4 = 5, p5 = 5, i = 1;
        string result;
        private Size originalFormSize;
    }
}

```

```

        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

        public Formb()
        {
            InitializeComponent();

            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.Green700, Primary.Green900,
                Primary.Green500, Accent.Green200,
                TextShade.WHITE
            );

            // Настраиваем порядок слоев для правильного
отображения
            this.backgroundPanel.BackColor = Color.White; //
Устанавливаем белый цвет фона
            this.backgroundPanel.SendToBack(); // Отправляем
фоновую панель на задний план

            switch (i)
            {
                case (2):
                    MessageBox.Show("Осталось четыре объекта.
Какой из них более других отражает грани Вашего характера? Нажмите
на это изображение");
                    label1.Text = "Осталось четыре объекта. Какой
из них более других отражает грани Вашего характера? Нажмите на
это изображение";
                    break;
                case (3):
                    MessageBox.Show("Осталось три объекта. Какой
из них подходит Вам больше, чем другие? Нажмите на это
изображение");
                    label1.Text = "Осталось три объекта. Какой из
них подходит Вам больше, чем другие? Нажмите на это изображение";
                    break;
                case (4):
                    MessageBox.Show("Осталось два объекта. Нажмите
на тот из них, который частично отражает Ваш характер, в отличие
от другого (100% не Вы)");
                    label1.Text = "Осталось два объекта. Нажмите
на тот из них, который частично отражает Ваш характер, в отличие
от другого (100% не Вы)";
                    break;
            }

```

Листинг Б.9 – Файл FormB1.cs

```

using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.Windows.Forms;
using System.Linq;
using MaterialSkin;
using MaterialSkin.Controls;
using System.Drawing.Drawing2D;

namespace Programm3
{
    public partial class FormB1 : MaterialForm
    {
        int lvl = 0;
        int[,] choises = { { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 },
        { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, {
        1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, {
        1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, {
        1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, {
        1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 }, { 1, 1, 1 } };
        bool entered = false;
        bool dragging = false;
        bool turn = false;
        Point startPoint;
        Point p;
        private Size originalFormSize;
        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();

        private List<MaterialCard> optionCards;
        private List<MaterialButton> optionButtons;

        public FormB1()
        {
            InitializeComponent();

            var materialSkinManager =
MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.BlueGrey800, Primary.BlueGrey900,
                Primary.BlueGrey500, Accent.LightBlue200,
                TextShade.WHITE
            );

            try

```

```

        {
            // Стилизация карточки заголовка
            if (titleCard != null)
            {
                // Применяем легкий фон и рамку для отделения
                titleCard.BackColor = Color.FromArgb(245, 245,
от фона
                250);
            }

            // Устанавливаем FontType и другие свойства для
MaterialLabel после инициализации MaterialSkinManager
            if (materialLabelSelectionMode != null)
            {
                materialLabelSelectionMode.FontType =
MaterialSkinManager.fontType.H6;
                materialLabelSelectionMode.ForeColor =
Color.FromArgb(0, 153, 0); // Начальный зеленый цвет
            }

            if (materialLabelTitle != null)
            {
                materialLabelTitle.Font = new Font("Roboto",
14F, FontStyle.Bold);
                materialLabelTitle.TextAlign =
ContentAlignment.MiddleCenter;
            }
        }
        catch (KeyNotFoundException ex)
        {
            // Обрабатываем ошибку словаря MaterialSkinManager
            Console.WriteLine("Ошибка при установке FontType:
" + ex.Message);

            // Устанавливаем шрифты напрямую
            if (materialLabelSelectionMode != null)
            {
                materialLabelSelectionMode.Font = new
Font("Roboto Medium", 20F, FontStyle.Bold);
            }

            if (materialLabelTitle != null)
            {
                materialLabelTitle.Font = new Font("Roboto",
14F, FontStyle.Bold);
            }
        }

        this.backgroundPanel.BringToFront();
        this.backgroundPanel.SendToBack();
        this.backgroundPanel.Paint += new
System.Windows.Forms.PaintEventHandler(this.backgroundPanel_Paint)
;

```

```

        typeof (Panel).InvokeMember("DoubleBuffered",
            System.Reflection.BindingFlags.SetProperty |
            System.Reflection.BindingFlags.Instance |
            System.Reflection.BindingFlags.NonPublic,
            null, this.backgroundPanel, new object[] { true
        });

        materialButtonBack.Text = "НАЗАД";
        materialButtonNext.Text = "ДАЛЕЕ";

        optionCards = new List<MaterialCard> { cardOption1,
        cardOption2, cardOption3 };
        optionButtons = new List<MaterialButton> {
        materialButton1, materialButton2, materialButton3 };
    }
    switch (lvl)
    {
        case 0:
            materialLabelTitle.Text = "Наибольшее
удовлетворение я получаю от:";
            materialButton1.Text = "Одобрения моей
работы";
            materialButton2.Text = "Сознания того, что
работа сделана хорошо";
            materialButton3.Text = "Сознание того, что
меня окружают друзья";
            break;
        case 1:
            materialLabelTitle.Text = "Если бы я играл в
футбол (волейбол, баскетбол), то я хотел бы быть:";
            materialButton1.Text = "Тренером, который
разрабатывает тактику игры";
            materialButton2.Text = "Известным игроком";
            materialButton3.Text = "Выбранным капитаном
команды";
            break;
        case 2:
            materialLabelTitle.Text = "По-моему, лучшим
педагогом является тот, кто:";
            materialButton1.Text = "Проявляет интерес к
учащимся и к каждому имеет индивидуальный подход";
            materialButton2.Text = "Увлечен своим
предметом и вызывает интерес к нему своих учеников";
            materialButton3.Text = "Создает в коллективе
такую атмосферу, при которой никто не" + (char)160 + "боится
высказать свое мнение";
            break;
        case 3:
            materialLabelTitle.Text = "Мне нравится, когда
люди:";
            materialButton1.Text = "Радуются выполненной
работе";

```

```

        materialButton2.Text = "С удовольствием
работают в коллективе";
        materialButton3.Text = "Стремятся выполнить
свою работу лучше других";
        break;
    case 4:
        materialLabelTitle.Text = "Я хотел бы, чтобы
мои друзья:";
        materialButton1.Text = "Были отзывчивы и
помогали людям, когда для этого представляются возможности";
        materialButton2.Text = "Были верны и преданны
мне";
        materialButton3.Text = "Были умными и
интересными людьми";
        break;
    case 5:
        materialLabelTitle.Text = "Лучшими друзьями я
считаю тех:";
        materialButton1.Text = "С кем складываются
хорошие взаимоотношения ";
        materialButton2.Text = "На кого всегда можно
положиться";
        materialButton3.Text = "Кто может многого
достичь в жизни";
        break;
    case 6:
        materialLabelTitle.Text = "Больше всего я не
люблю:";
        materialButton1.Text = "Когда у меня что-то не
получается";
        materialButton2.Text = "Когда портятся
отношения с товарищами";
        materialButton3.Text = "Когда меня критикуют";
        break;
    case 7:
        materialLabelTitle.Text = "По-моему, хуже
всего, когда педагог:";
        materialButton1.Text = "Не скрывает, что
некоторые учащиеся ему не симпатичны, насмехается и подшучивает
над ними";
        materialButton2.Text = "Вызывает дух
соперничества в коллективе";
        materialButton3.Text = "Недостаточно хорошо
знает предмет, который преподает";
        break;
    case 8:
        materialLabelTitle.Text = "В детстве мне
больше всего нравилось:";
        materialButton1.Text = "Проводить время с
друзьями";
        materialButton2.Text = "Ощущение выполненных
дел";

```

```

        materialButton3.Text = "Когда меня за что-
нибудь хвалили";
        break;
    case 9:
        materialLabelTitle.Text = "Я хотел бы быть
похожим на тех, кто:";
        materialButton1.Text = "Добился успеха в
жизни";
        materialButton2.Text = "По-настоящему увлечен
своим делом";
        materialButton3.Text = "Отличается дружелюбием
и доброжелательностью";
        break;
    case 10:
        materialLabelTitle.Text = "В первую очередь
школа должна:";
        materialButton1.Text = "Научить учиться и
работать, решать задачи, которые ставит жизнь";
        materialButton2.Text = "Развивать
индивидуальные способности и самостоятельность ученика";
        materialButton3.Text = "Воспитывать качества,
помогающие взаимодействовать с людьми";
        break;
    case 11:
        materialLabelTitle.Text = "Если бы у меня было
больше свободного времени, охотнее всего я использовал бы его:";
        materialButton1.Text = "Для общения с
друзьями";
        materialButton2.Text = "Для отдыха и
развлечений";
        materialButton3.Text = "Для своих любимых дел
и самообразования";
        break;
    case 12:
        materialLabelTitle.Text = "Наибольших успехов
я добиваюсь, когда:";
        materialButton1.Text = "Работаю с людьми,
которые мне симпатичны";
        materialButton2.Text = "У меня интересная
работа";
        materialButton3.Text = "Мои усилия хорошо
вознаграждаются";
        break;
    case 13:
        materialLabelTitle.Text = "Я люблю, когда:";
        materialButton1.Text = "Другие люди меня
ценят";
        materialButton2.Text = "Испытываю
удовлетворение от хорошо выполненной работы";
        materialButton3.Text = "Приятно провожу время
с друзьями";
        break;
    case 14:

```

```

        materialLabelText.Text = "Если бы обо мне
решили написать в газете, мне бы хотелось, чтобы:";
        materialButton1.Text = "Рассказали о каком-
либо интересном деле, связанном с учебой, работой, спортом и т.п.,
в котором мне довелось участвовать";
        materialButton2.Text = "Написали о моей
деятельности";
        materialButton3.Text = "Обязательно рассказали
о коллективе, в котором я работаю";
        break;
    case 15:
        materialLabelText.Text = "Лучше всего я
учусь, если преподаватель:";
        materialButton1.Text = "Имеет ко мне
индивидуальный подход";
        materialButton2.Text = "Сумеет вызвать у меня
интерес к предмету";
        materialButton3.Text = "Устраивает
коллективные обсуждения изучаемых проблем";
        break;
    case 16:
        materialLabelText.Text = "Для меня нет ничего
хуже, чем:";
        materialButton1.Text = "Оскорбление личного
достоинства";
        materialButton2.Text = "Неудача при выполнении
важного дела";
        materialButton3.Text = "Потеря друзей";
        break;
    case 17:
        materialLabelText.Text = "Больше всего я
ценю:";
        materialButton1.Text = "Успех";
        materialButton2.Text = "Возможности хорошей
совместной работы";
        materialButton3.Text = "Здравый практичный ум
и смекалку";
        break;
    case 18:
        materialLabelText.Text = "Я не люблю людей,
которые:";
        materialButton1.Text = "Считают себя хуже
других";
        materialButton2.Text = "Часто ссорятся и
конфликтуют";
        materialButton3.Text = "Возражают против всего
нового";
        break;
    case 19:
        materialLabelText.Text = "Приятно, когда:";
        materialButton1.Text = "Работаешь над важным
делом";
        materialButton2.Text = "Имеешь много друзей";

```

```

        materialButton3.Text = "Вызываешь восхищение и
всем нравишься";
        break;
    case 20:
        materialLabelTitle.Text = "По-моему, в первую
очередь руководитель должен быть:";
        materialButton1.Text = "Доступным";
        materialButton2.Text = "Авторитетным";
        materialButton3.Text = "Требовательным";
        break;
    case 21:
        materialLabelTitle.Text = "В свободное время я
охотно прочитал бы книги:";
        materialButton1.Text = "О том, как заводить
друзей и поддерживать хорошие отношения с людьми";
        materialButton2.Text = "О жизни знаменитых и
интересных людей";
        materialButton3.Text = "О последних
достижениях науки и техники";
        break;
    case 22:
        materialLabelTitle.Text = "Если бы у меня были
способности к музыке, я предпочел бы быть:";
        materialButton1.Text = "Дирижером";
        materialButton2.Text = "Композитором";
        materialButton3.Text = "Солистом";
        break;
    case 23:
        materialLabelTitle.Text = "Мне бы хотелось:";
        materialButton1.Text = "Придумать интересный
конкурс";
        materialButton2.Text = "Победить в конкурсе";
        materialButton3.Text = "Организовать конкурс и
руководить им";
        break;
    case 24:
        materialLabelTitle.Text = "Для меня важнее
всего знать:";
        materialButton1.Text = "Что я хочу сделать";
        materialButton2.Text = "Как достичь цели";
        materialButton3.Text = "Как организовывать
людей для достижения цели";
        break;
    case 25:
        materialLabelTitle.Text = "Человек должен
стремиться к тому, чтобы:";
        materialButton1.Text = "Другие были им
довольны";
        materialButton2.Text = "Прежде всего выполнить
свою задачу";
        materialButton3.Text = "Его не нужно было
упрекать за выполненную работу";
        break;

```

```

        case 26:
            materialLabelTitle.Text = "Лучше всего я
отдыхаю в свободное время:";
            materialButton1.Text = "В общении с друзьями";
            materialButton2.Text = "Просматривая
развлекательные фильмы";
            materialButton3.Text = "Занимаясь своим
любимым делом";

            break;
        case 27:
            materialLabelTitle.Text = "Для меня важнее
всего определить:";
            materialButton1.Text = "Что я хочу сделать";
            materialButton2.Text = "Как достичь цели";
            materialButton3.Text = "Как привлечь других к
достижению моей цели";
            break;
        case 28:
            CalculateAndProcessResults();
            break;
    }

    return 0;
}

private void CalculateAndProcessResults()
{
    int Y = 0, O = 0, D = 0;
    Y = choices[0, 0] + choices[1, 1] + choices[2, 0] +
choices[3, 2] + choices[4, 1]
    + choices[5, 2] + choices[6, 2] + choices[7, 0] +
choices[8, 2]
    + choices[9, 0] + choices[10, 1] + choices[11, 1]
+ choices[12, 2]
    + choices[13, 0] + choices[14, 1] + choices[15, 0]
+ choices[16, 0]
    + choices[17, 0] + choices[18, 0] + choices[19, 2]
+ choices[20, 1]
    + choices[21, 1] + choices[22, 2] + choices[23, 1]
+ choices[24, 0]
    + choices[25, 2] + choices[26, 1] + choices[27,
0];

    O = choices[0, 2] + choices[1, 2] + choices[2, 2] +
choices[3, 1] + choices[4, 0]
    + choices[5, 0] + choices[6, 1] + choices[7, 1] +
choices[8, 0]
    + choices[9, 2] + choices[10, 2] + choices[11, 0]
+ choices[12, 0]
    + choices[13, 2] + choices[14, 2] + choices[15, 2]
+ choices[16, 2]
    + choices[17, 1] + choices[18, 1] + choices[19, 1]
+ choices[20, 0]

```

```

        + chois[21, 0] + chois[22, 0] + chois[23, 2]
+ chois[24, 2]
        + chois[25, 0] + chois[26, 0] + chois[27,
2];

```

```

        D = chois[0, 1] + chois[1, 0] + chois[2, 1] +
chois[3, 0] + chois[4, 2]
        + chois[5, 1] + chois[6, 0] + chois[7, 2] +
chois[8, 1]
        + chois[9, 1] + chois[10, 0] + chois[11, 2]
+ chois[12, 1]
        + chois[13, 1] + chois[14, 0] + chois[15, 1]
+ chois[16, 1]
        + chois[17, 2] + chois[18, 2] + chois[19, 0]
+ chois[20, 2]
        + chois[21, 2] + chois[22, 1] + chois[23, 0]
+ chois[24, 1]
        + chois[25, 1] + chois[26, 2] + chois[27,
1];

```

```

        string YY, OO, DD;
        if (Y < 10) YY = "0" + Y.ToString();
        else YY = Y.ToString();
        if (O < 10) OO = "0" + O.ToString();
        else OO = O.ToString();
        if (D < 10) DD = "0" + D.ToString();
        else DD = D.ToString();
        string result = (char)938 +
DateTime.Today.ToString().Remove(10, 8) + YY.ToString() +
OO.ToString() + DD.ToString();

```

```

        string s = result.Remove(11, result.Length - 11);
        string bd = Crypto.Type(File.ReadAllText(Data.location
+ Data.group + @"\" + Data.name + @"\" + "result.era"),
"Decrypt");

```

```

        int j;
        int i;
        for (i = 0; i < bd.Length; i++)
        {
            if (s[0] == bd[i])
            {
                for (j = 1; j < 11; j++)
                {
                    if (s[j] != bd[i + j]) break;
                }
                if (j == 11)
                {
                    string left, right;
                    if (i == 0) left = null;
                    else left = bd.Remove(i, bd.Length - i);
                    if (i + 17 == bd.Length) right = null;
                    else right = bd.Remove(0, i + 17);
                    result = left + result + right;
                }
            }
        }

```

```

        break;
    }
    i += j;
}
}
if (i == bd.Length) result = bd + result;
HAE.HOS(Data.location + Data.group + @"\" + Data.name
+ @"\" + "result.era", "Show");
File.WriteAllText(Data.location + Data.group + @"\" +
Data.name + @"\" + "result.era", Crypto.Type(result, "Encrypt"));
HAE.HOS(Data.location + Data.group + @"\" + Data.name
+ @"\" + "result.era", "Hide");
Data.instruction = 3;
Data.locform = this.Location;
Form f = new Form4();
this.Hide();
f.Show();
}
private void Form_Load(object sender, EventArgs e)
{
    this.Location = Data.locform;
    materialButtonBack.Visible = false;

    originalFormSize = this.Size;
    foreach (Control control in GetAllControls(this))
    {
        controlsData[control] = new Rectangle(
            control.Left, control.Top,
            control.Width, control.Height
        );
        originalFonts[control] = control.Font;
    }

    if (Data.isMaximized)
    {
        this.WindowState = FormWindowState.Maximized;
    }
    else
    {
        this.WindowState = FormWindowState.Normal;
        this.Size = originalFormSize;
        if (this.StartPosition == FormStartPosition.Manual
|| this.StartPosition == FormStartPosition.CenterParent)
        {
            int screenWidth =
Screen.PrimaryScreen.WorkingArea.Width;
            int screenHeight =
Screen.PrimaryScreen.WorkingArea.Height;
            this.Location = new Point(
                (screenWidth - this.Width) / 2,
                (screenHeight - this.Height) / 2
            );
        }
    }
}

```

```

    }
    Form_Resize(sender, e);
}

private void Form_Resize(object sender, EventArgs e)
{
    if (controlsData.Count == 0 || originalFormSize.Width
== 0 || originalFormSize.Height == 0) return;

    float widthRatio = (float)this.ClientSize.Width /
originalFormSize.Width;
    float heightRatio = (float)this.ClientSize.Height /
originalFormSize.Height;
    float fontScaleFactor = Math.Min(widthRatio,
heightRatio);
    fontScaleFactor = Math.Min(fontScaleFactor, 1.1f);

    foreach (Control control in
controlsData.Keys.ToList())
    {
        if (control == this.backgroundPanel) continue;
        if (!controlsData.ContainsKey(control) ||
!originalFonts.ContainsKey(control)) continue;

        var rect = controlsData[control];

        control.Left = (int)(rect.X * widthRatio);
        control.Top = (int)(rect.Y * heightRatio);
        control.Width = (int)(rect.Width * widthRatio);

        if (control is MaterialCard || control is
MaterialButton)
        {
            control.Height = (int)(rect.Height *
heightRatio * 0.98);
        }
        else
        {
            control.Height = (int)(rect.Height *
heightRatio);
        }

        if (originalFonts.ContainsKey(control))
        {
            try
            {
                float newSize =
originalFonts[control].Size * fontScaleFactor;
                if (newSize > 0.1f)
                {
                    control.Font = new
Font(originalFonts[control].FontFamily, newSize,
originalFonts[control].Style);

```

```

        }
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Font scaling error for
{control.Name}: {ex.Message}");
    }
}
}
this.backgroundPanel.Invalidate();
}

```

Листинг Б.10 – Файл Results.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MaterialSkin;
using MaterialSkin.Controls;
using System.Windows.Forms.DataVisualization.Charting;

namespace Programm3
{
    public partial class Results : MaterialForm
    {
        bool code = false;
        private Size originalFormSize;

        private readonly Dictionary<Control, Rectangle>
controlsData = new Dictionary<Control, Rectangle>();
        private Dictionary<Control, Font> originalFonts = new
Dictionary<Control, Font>();
        private Dictionary<PictureBox, Size> originalPicSizes =
new Dictionary<PictureBox, Size>();

        public Results()
        {
            InitializeComponent();

            materialSkinManager = MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme =
MaterialSkinManager.Themes.LIGHT;
            materialSkinManager.ColorScheme = new ColorScheme(
                Primary.Indigo500, Primary.Indigo700,
                Primary.Indigo100, Accent.Pink200,

```

```

        TextShade.WHITE
    );

    InitializeAdditionalProperties();
}

private void InitializeAdditionalProperties()
{
    if (this.chartResults != null)
    {
        this.chartResults.Visible = false;
        this.chartResults.BackColor = Color.Transparent;
        this.chartResults.Palette =
ChartColorPalette.Pastel;
        this.chartResults.ChartAreas.Clear();
        this.chartResults.Series.Clear();
        this.chartResults.Legends.Clear();
        this.chartResults.Titles.Clear();
    }
    if (this.materialTextBoxInterpretation != null)
    {
        this.materialTextBoxInterpretation.Text = "";
        this.materialTextBoxInterpretation.Visible =
false;
        this.materialTextBoxInterpretation.BackColor =
this.BackColor; // Для лучшей интеграции
    }
    if (this.panelScrollableTextResults != null)
    {
        this.panelScrollableTextResults.BackColor =
Color.Transparent; // или this.tabPageTextResults.BackColor
    }

    if (chartResults != null)
    {
        chartResults.Visible = false;
        chartResults.Series.Clear();
        chartResults.ChartAreas.Clear();
        chartResults.Titles.Clear();
        chartResults.Legends.Clear();
    }
    if (materialTextBoxInterpretation != null)
    {
        materialTextBoxInterpretation.Text = "";
        materialTextBoxInterpretation.Visible = false;
    }
    if (tabControlResults.TabPages.Count > 0)
        tabControlResults.SelectedIndex = 0; // По
умолчанию первая вкладка
    return false;
}

private void PositionShapesEvenly(string result)

```

```

        {
            PictureBox[] shapePictureBoxes = { pic3, pic4, pic5,
pic2, pic1 };
            char[] shapeOrderChars = { '1', '2', '3', '4', '5' };

            private void InterpretGarbuzovResults(string resultData)
            {
                if (materialTextBoxInterpretation == null) return;
                StringBuilder sb = new StringBuilder();
                sb.AppendLine("АНАЛИЗ РЕЗУЛЬТАТОВ (Опросник
Гарбузова):");

                sb.AppendLine("=====");

                int[] verbal = new int[7];
                int[] nonVerbal = new int[7];
                int[] total = new int[7];
                string[] instinctFullNames = {
                    "Эгофильный (инстинкт самосохранения, фокус на
себе, безопасности, комфорте)",
                    "Генофильный (инстинкт продолжения рода, забота о
детях, семье, \"своих\")",
                    "Альтруистический (сопереживание, помощь другим,
часто в ущерб себе)",
                    "Исследовательский (любопытность, тяга к
новому, творчеству, открытиям)",
                    "Доминантный (стремление к лидерству, контролю,
влиянию, организации)",
                    "Либертофильный (инстинкт свободы, протест против
ограничений, независимость)",
                    "Дигниతోфильный (сохранение самоуважения, чести,
нетерпимость к унижению)"
                };
                string[] instinctAdvice = {
                    "Высокий 'Эгофильный' говорит о важности для Вас
личной безопасности и благополучия. Это помогает избегать рисков,
но чрезмерность может вести к тревожности.",
                    "Высокий 'Генофильный' указывает на сильную
привязанность к близким и заботу о них. Это основа крепких
семейных уз, но может приводить к гиперопеке.",
                    "Высокий 'Альтруистический' означает готовность
помогать другим. Это ценное качество, но важно не забывать и о
собственных потребностях.",
                    "Высокий 'Исследовательский' – это двигатель
прогресса и личного развития. Поощряйте свою любопытность, но
не забывайте о практическом применении знаний.",
                    "Высокий 'Доминантный' дает Вам способность вести
за собой и организовывать. Используйте это для достижения общих
целей, уважая мнение других.",
                    "Высокий 'Либертофильный' – это стремление к
независимости. Цените свою свободу, но помните об ответственности
и обязательствах.",

```

"Высокий 'Дигниతోфильный' подчеркивает Ваше чувство собственного достоинства. Это важно для самоуважения, но избегайте излишней обидчивости."

```
};

try
{
    for (int i = 0; i < 7; i++)
    {
        verbal[i] =
int.Parse(resultData[i].ToString());
        nonVerbal[i] = int.Parse(resultData[i +
7].ToString());
        total[i] = verbal[i] + nonVerbal[i];
    }
}
catch { materialTextBoxInterpretation.Text = "Ошибка
данных для интерпретации."; return; }

sb.AppendLine("Оценка выраженности инстинктов (сумма
вербального и невербального компонента, макс. 8 баллов:");
List<Tuple<int, int>> sortedInstincts = new
List<Tuple<int, int>>();
for (int i = 0; i < 7; i++) sortedInstincts.Add(new
Tuple<int, int>(total[i], i));
sortedInstincts = sortedInstincts.OrderByDescending(t
=> t.Item1).ToList();

foreach (var instTuple in sortedInstincts)
{
    int score = instTuple.Item1;
    int index = instTuple.Item2;
    string level;
    if (score >= 7) level = "Очень высокий уровень";
    else if (score >= 5) level = "Высокий уровень";
    else if (score >= 3) level = "Средний уровень";
    else level = "Низкий уровень";
    sb.AppendLine($"{instinctFullNames[index]}:
{score} баллов ({level}).");
}
sb.AppendLine();

sb.AppendLine("Ключевые аспекты вашего профиля:");
if (sortedInstincts[0].Item1 >= 6) // Если самый
сильный инстинкт достаточно выражен
{
    sb.AppendLine($"Наиболее выражен у Вас
{instinctFullNames[sortedInstincts[0].Item2].Split(' ')[0].Trim()}
.");
}

sb.AppendLine(instinctAdvice[sortedInstincts[0].Item2]);
```

```

        if (sortedInstincts[1].Item1 >= 5 &&
sortedInstincts[0].Item1 - sortedInstincts[1].Item1 <= 2) // Если
второй тоже сильный и близок к первому
        {
            sb.AppendLine($"Также значим
{instinctFullNames[sortedInstincts[1].Item2].Split('(')[0].Trim()}
.");

sb.AppendLine(instinctAdvice[sortedInstincts[1].Item2]);
            sb.AppendLine("Такое сочетание может указывать
на сложную, но потенциально очень продуктивную мотивационную
структуру.");
        }
    }
    else
    {
        sb.AppendLine("У Вас достаточно сбалансированный
профиль инстинктов без резко доминирующих тенденций. " +
            "Это может говорить о гибкости в
поведении и способности адаптироваться к разным ситуациям, " +
            "однако иногда может не хватать
четкой направленности для достижения долгосрочных целей.");
    }
    sb.AppendLine();
    sb.AppendLine("Для детального понимания каждого
инстинкта нажмите на его название в списке слева (если
доступно).");
    materialTextBoxInterpretation.Text = sb.ToString();
}

private void InterpretPsychoGeometryResults(string
resultData)
{
    if (materialTextBoxInterpretation == null) return;
    StringBuilder sb = new StringBuilder();
    sb.AppendLine("АНАЛИЗ РЕЗУЛЬТАТОВ (Психогеометрия):");

sb.AppendLine("=====");

    if (string.IsNullOrEmpty(resultData) ||
resultData.Length < 1)
    {
        materialTextBoxInterpretation.Text = "Недостаточно
данных для интерпретации."; return;
    }

    char firstChoiceChar = resultData[0];
    string firstChoiceName = "", dominantTrait = "",
secondaryTrait = "", communicationStyle = "", stressBehavior = "";

    switch (firstChoiceChar)
    {
        case '1': // Квадрат

```

```

firstChoiceName = "КВАДРАТ";
dominantTrait = "Организованность и
трудолюбие. Вы - человек структуры, порядка и деталей. Цените
стабильность, надежность и последовательность.";
secondaryTrait = "Аналитический склад ума. Вы
предпочитаете собирать информацию, анализировать факты и принимать
взвешенные решения.";
communicationStyle = "Общение четкое, по делу,
без лишних эмоций. Иногда можете казаться суховатым.";
stressBehavior = "В стрессе можете становиться
излишне педантичным, критичным или замыкаться в себе, прорабатывая
детали.";
break;
case '2': // Прямоугольник
firstChoiceName = "ПРЯМОУГОЛЬНИК";
dominantTrait = "Любознательность и открытость
новому. Вы сейчас в состоянии перехода, поиска, активного усвоения
информации и опыта.";
secondaryTrait = "Некоторая
непоследовательность. Из-за поиска лучших решений можете менять
мнения или планы.";
communicationStyle = "Можете быть восприимчивы
к мнению других, легко вступать в контакт, но иногда
поверхностно.";
stressBehavior = "В стрессе можете испытывать
растерянность, суетливость, становиться более внушаемым или
раздражительным.";
break;
case '3': // Треугольник
firstChoiceName = "ТРЕУГОЛЬНИК";
dominantTrait = "Лидерство и
целеустремленность. Вы - энергичный, решительный человек,
сфокусированный на достижении целей и победе.";
secondaryTrait = "Уверенность в себе и
амбициозность. Вы стремитесь к успеху, признанию и карьерному
росту.";
communicationStyle = "Речь быстрая, четкая,
убедительная. Умеете концентрироваться на главном. Можете быть
нетерпимы к возражениям.";
stressBehavior = "В стрессе можете становиться
нетерпеливым, авторитарным, категоричным или идти напролом.";
break;
case '4': // Круг
firstChoiceName = "КРУГ";
dominantTrait = "Ориентация на отношения и
гармонию. Вы - доброжелательный, чуткий человек, для которого
важны люди и хорошие отношения с ними.";
secondaryTrait = "Эмпатия и
коммуникабельность. Вы хорошо чувствуете других, умеете слушать и
сглаживать конфликты.";
communicationStyle = "Общение мягкое,
доброжелательное, стремитесь к согласию. Можете быть нерешительны
в принятии жестких мер.";

```

```

        stressBehavior = "В стрессе можете становиться
излишне уступчивым, эмоциональным, принимать все близко к сердцу
или избегать конфликтов.";
        break;
        case '5': // Зигзаг
            firstChoiceName = "ЗИГЗАГ";
            dominantTrait = "Креативность и
оригинальность. Вы - генератор идей, человек с нестандартным
мышлением, устремленный в будущее.";
            secondaryTrait = "Интуиция и любовь к новизне.
Вы легко комбинируете разные идеи, видите мир в постоянном
изменении.";
            communicationStyle = "Речь яркая, образная,
увлекательная. Можете быть остроумны, но иногда непоследовательны
или непрактичны.";
            stressBehavior = "В стрессе можете проявлять
недисциплинированность, разбрасываться, становиться эксцентричным
или бунтовать против правил.";
            break;
        default:
            materialTextBoxInterpretation.Text =
"Выбранная фигура не распознана."; return;
    }

    sb.AppendLine($"Ваш психологический тип по первому
выбору: {firstChoiceName}.");
    sb.AppendLine("Это означает, что Ваши основные
личностные качества, стиль мышления и поведения соответствуют
характеристикам данной фигуры.");
    sb.AppendLine();
    sb.AppendLine($"Ключевые черты: {dominantTrait}");
    sb.AppendLine($"Дополнительные аспекты:
{secondaryTrait}");
    sb.AppendLine($"Стиль общения: {communicationStyle}");
    sb.AppendLine($"Поведение в стрессовых ситуациях:
{stressBehavior}");
    sb.AppendLine();
    sb.AppendLine("Фигуры, выбранные Вами далее (2-я, 3-я
и т.д.), описывают Ваши развивающиеся качества, скрытые
потребности или то, как Вы взаимодействуете с другими типами
личности.");
    sb.AppendLine("Для получения полного описания всех
фигур и их значения в последовательности, нажмите на изображение
фигуры на вкладке 'Детальные результаты'.");

    materialTextBoxInterpretation.Text = sb.ToString();
}

private void InterpretPersonalityOrientationResults(string
resultData)
{
    if (materialTextBoxInterpretation == null) return;
    StringBuilder sb = new StringBuilder();

```

```

        sb.AppendLine("АНАЛИЗ РЕЗУЛЬТАТОВ (Направленность
личности):");

sb.AppendLine("=====");

        int self = 0, communication = 0, task = 0;
        try
        {
            self = int.Parse(resultData.Substring(0, 2));
            communication = int.Parse(resultData.Substring(2,
2));

            task = int.Parse(resultData.Substring(4, 2));
        }
        catch { materialTextBoxInterpretation.Text = "Ошибка
данных для интерпретации."; return; }

        sb.AppendLine($"- Направленность на себя (личные цели,
интересы): {self} баллов.");
        sb.AppendLine($"- Направленность на общение
(взаимодействие, отношения): {communication} баллов.");
        sb.AppendLine($"- Направленность на дело (задачи,
результаты): {task} баллов.");
        sb.AppendLine();

        string interpretation = "";
        // Анализ доминирующей направленности (упрощенный)
        if (self > communication && self > task && self >= 25)
        // Порог для выраженности
        {
            interpretation = "У Вас преобладает НАПРАВЛЕННОСТЬ
НА СЕБЯ. Это означает, что Вы, вероятно, в первую очередь
ориентированы на достижение личных целей, саморазвитие и
удовлетворение собственных потребностей. Вы цените независимость и
личное пространство. Важно следить, чтобы это не переросло в
эгоцентризм и не мешало командной работе.";
        }
        else if (communication > self && communication > task
&& communication >= 25)
        {
            interpretation = "У Вас преобладает НАПРАВЛЕННОСТЬ
НА ОБЩЕНИЕ. Для Вас крайне важны гармоничные отношения с
окружающими, командный дух и социальное одобрение. Вы хороший
командный игрок и стремитесь к сотрудничеству. Следите, чтобы
стремление к хорошим отношениям не шло вразрез с необходимостью
достижения результата.";
        }
        else if (task > self && task > communication && task
>= 25)
        {
            interpretation = "У Вас преобладает НАПРАВЛЕННОСТЬ
НА ДЕЛО (ЗАДАЧУ). Вы сфокусированы на достижении результатов,
выполнении поставленных задач и эффективности. Вы цените

```

```

компетентность и профессионализм. Важно не забывать о человеческом
факторе и о потребностях коллег.";
    }
    else if (self > 20 && communication > 20 && task > 20)
// Примерный сбалансированный профиль
    {
        interpretation = "У Вас СБАЛАНСИРОВАННЫЙ ТИП
НАПРАВЛЕННОСТИ. Вы умеете сочетать личные интересы, потребности в
общении и ориентацию на задачу. Это говорит о хорошей адаптивности
и гибкости. В разных ситуациях Вы можете проявлять ту или иную
доминанту.";
    }
    else // Другие комбинации или низкие значения по всем
шкалам
    {
        interpretation = "Ваш профиль направленности
требуется более детального рассмотрения. Возможно, нет ярко
выраженной доминанты, или все показатели находятся на
среднем/низком уровне. Это может указывать на ситуативную гибкость
или, в некоторых случаях, на неопределенность в приоритетах.
Рекомендуется проанализировать, как эти показатели проявляются в
Вашей повседневной деятельности.";
    }
    sb.AppendLine(interpretation);
    sb.AppendLine();
    sb.AppendLine("Это общая интерпретация. Соотношение
баллов дает более тонкую картину Ваших мотивационных
предпочтений.");

    materialTextBoxInterpretation.Text = sb.ToString();
}

// Сохраняем данные о кнопках для масштабирования
foreach (Control control in this.Controls)
{
    if (control is Button &&
(control.Name.StartsWith("btn") && control.Name != "button2"))
    {
        controlsData[control] = new Rectangle(
            control.Left, control.Top,
            control.Width, control.Height
        );
        originalFonts[control] = control.Font;
    }
}
else
{
    // Удаляем кнопки, если мы не отображаем
психогеометрию
    List<Control> controlsToRemove = new
List<Control>();

```

```

        foreach (Control control in this.Controls)
        {
            if (control is Button &&
                (control.Name.StartsWith("btn") && control.Name != "button2"))
            {
                controlsToRemove.Add(control);
            }
        }
        string[] instinctTypes = new string[] {
            "Эгофильный тип инстинкта",
            "Генофильный тип инстинкта",
            "Альтруистический тип инстинкта",
            "Исследовательский тип инстинкта",
            "Доминантный тип инстинкта",
            "Либертофильный тип инстинкта",
            "Дигниతోфильный тип инстинкта"
        };
    }
}

```