

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

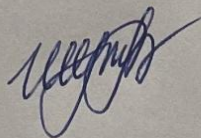
к курсовой работе
по курсу «Программирование»
на тему «Игра Рас-Ман»

Выполнил:

студент группы 24ВВВЗ

Давыдов А.Р.

15.05.2025
хор.



Принял:

к.т.н., доцент Токарев А.Н.

Пенза 2025

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники
Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

М.А. Митрохин

«___» _____ 20__ г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Давыдов Арсений Султанович Группа 24ВВВ3

Тема проекта: *Разработка программы сложной структуры методом нисходящего программирования.*

Исходные данные (технические требования) на проектирование

Игра "Рас-Ман"

- Создание препятствий
- Генерация пола
- Система движений противников
- Система движения персонажа Рас-ман
- Создание системы победы и поражения
- Создание системы счета побед

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания “20” Февраля 2025

Дата защиты проекта “15” Май 2025

Руководитель _____

Задание получил “20” Февраля 2025г.

Студент Давыдов Арсений Русланович _____

Содержание

Введение	5
1.Постановка задачи	6
2. Выбор решения	7
3.Описание разработки программы	8
4. Отладка и тестирование	12
5.Описание программы	14
6. Руководство пользователя	32
Заключение	33
Список используемых источников	34

Введение

Игра Рас-Ман - классическая аркадная игра, цель которой — управление персонажем, сбор точек и избегание призраков. Актуальность разработки заключается в изучении алгоритмов искусственного интеллекта для призраков, обработки коллизий, работы с графическим интерфейсом (GUI) и интеграции внешних модулей.

Цель работы: создание игры Рас-Ман с поддержкой управления, системы очков, жизней, различных режимов поведения призраков и сохранением статистики.

Задачи:

1. Реализация игровой логики движение Рас-Ман, генерация точек, поведение призраков.
2. Разработка GUI с использованием Windows Forms.
3. Интеграция DLL-модуля для работы с системой очков.
4. Реализация системы статистики (сохранение побед и поражений).
5. Добавление анимации персонажей и визуальных эффектов.
6. Запись и хранение статистики в отдельном документе.

1. Постановка задачи

Требовалось разработать программу, соответствующую следующим требованиям.

1. Функционал игры:

- Управление Рас-Ман с помощью клавиш (вверх, вниз, влево, вправо).
- Генерация точек и энерджайзеров на игровом поле.
- Реализация поведения призраков в режимах преследование, патрулирование, испуг.

- Система жизней и очков.

- Условия победы (сбор всех точек) и поражения (потеря всех жизней).

2. Интерфейс:

- Окно авторизации - поля для ввода логина и пароля. Кнопки “Вход” и “Выход”.

- Окно статистики - показана статистика побед/поражений. Кнопка “Начать игру”.

- Игровое окно - стандартный интерфейс Рас-Ман с отображением стен, точек, призраков и статистики.

3. Дополнительно:

- Использование DLL-файла для работы с системой очков.
- Сохранение статистики в файл game_stats.txt.

2. Выбор решения

Во-первых, C# в связке с Windows Forms предоставляет идеальные инструменты для реализации многооконного интерфейса, необходимого для игры Рас-Ман. Создание форм авторизации, статистики и игрового поля занимает минимум времени благодаря интуитивному дизайнеру интерфейса и готовым компонентам (кнопки, текстовые поля, таблицы). Например, для окна авторизации достаточно добавить два поля TextBox для логина и пароля, кнопки «Вход» и «Выход», а также подключить их к логике проверки данных через события Click. Встроенная событийная модель Windows Forms упрощает обработку действий пользователя - нажатия клавиш для управления Рас-Ман, переключения между окнами, валидации вводимых данных.

Во-вторых, C# устраняет необходимость ручного управления ресурсами, что критично для стабильной работы игры, сборщик мусора автоматически освобождает память, предотвращая утечки даже при длительных игровых сессиях. Работа с файлами упрощена благодаря классам System.IO, статистика пользователей сохраняется в персонализированные файлы (например, user123_stats.txt), а пароли шифруются через интеграцию с DLL-модулем Security.dll, что обеспечивает безопасное хранение данных. Графическая подсистема GDI+ позволяет легко реализовать анимацию Рас-Ман (например, эффект «открывания рта» через метод FillPie) и отрисовку призраков с изменяющимися цветами в зависимости от их режима.

В итоге C# оказался оптимальным выбором, он позволяет быстро разрабатывать функциональные приложения, не жертвуя при этом качеством.

3.Описание разработки программы

Проект представляет собой многооконную игру Pac-Man, состоящую из трёх ключевых форм - окно авторизации (LoginForm), окно статистики (MainMenuForm), игровое окно (Class1).

1. Меню авторизации (LoginForm) представлено на рисунке 1.

- Поля TextBox для ввода логина (txtUsername) и пароля (txtPassword). Кнопки «Войти» (btnLogin) и «Выход» (btnExit).
- Проверка логина и пароля через хардкод (логин - arsen, пароль - 12345). При успешной проверке открывается MainMenuForm, текущая форма закрывается. При ошибке выводится сообщение MessageBox.

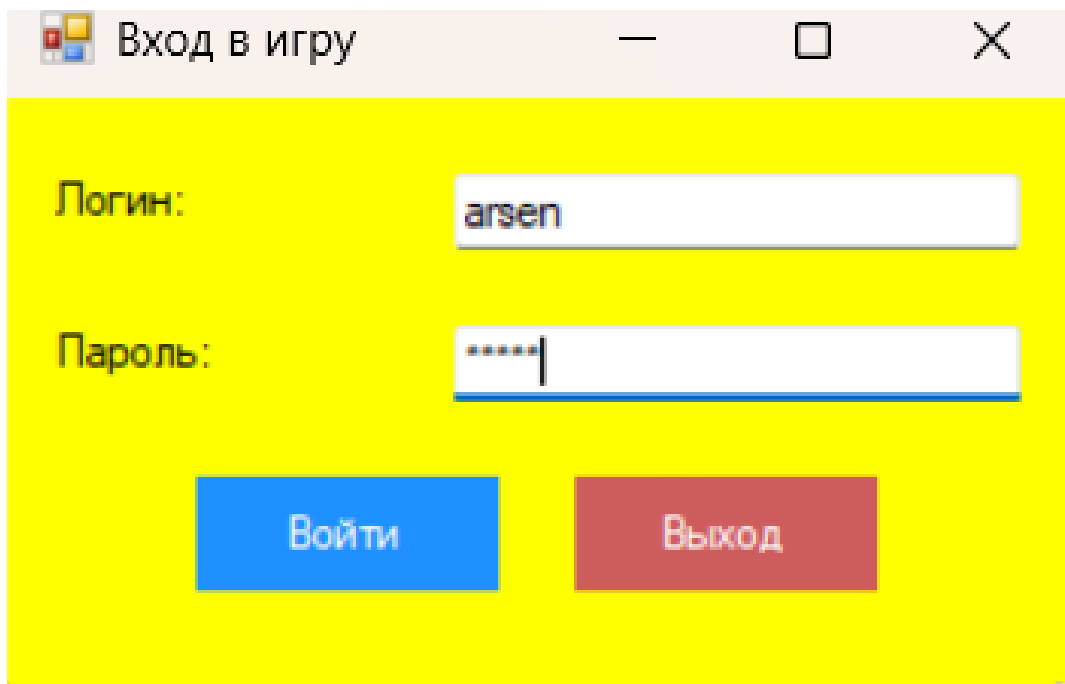


Рисунок 1 – меню авторизации.

2. Меню входа в игру (MainMenuForm) предоставлено на рисунке 2.

– Отображение статистики игрока (победы/поражения). Запуск игры через кнопку «Начать игру».

Визуализация:

– Приветствие пользователя (lblWelcome).

– Статистика в виде текста (lblStats).

– Кнопка «Начать игру» (btnStartGame).

Логика игры:

– Загрузка статистики из файла game_stats.txt при инициализации.

– Нажатие кнопки скрывает меню и запускает игровое окно Form1.

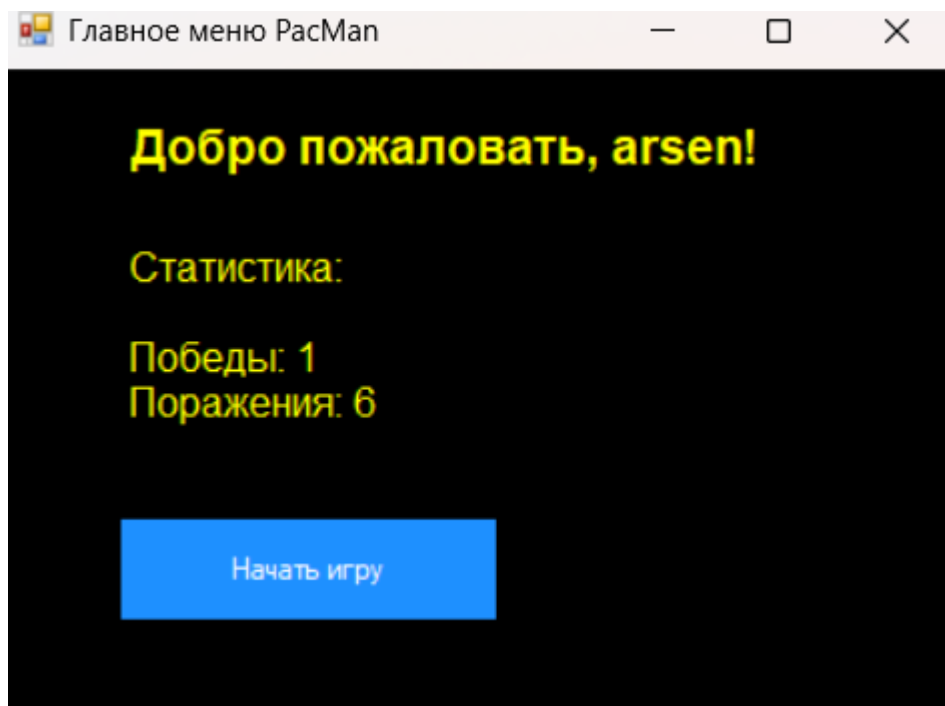


Рисунок 2 - игровое меню.

3. Игровое поле (Class1) предоставлено на рисунке 3.

- Реализация игрового процесса Pac-Man.

Игровая механика:

- Движение Pac-Man - обработка нажатий клавиш (←, →, ↑, ↓) через метод Form1_KeyDown.

- Генерация карты - стены задаются списком Rectangle в методе InitializeWalls. Точки и энерджайзеры создаются методом InitializeDots.

- Режимы призраков - Chase (преследование), Scatter (патрулирование), Frightened (испуг).

- Сбор точек - удаление точек с поля, активация энерджайзеров.

Визуализация:

- Отрисовка стен, точек, Pac-Man и призраков через GDI+ (метод OnPaint).

- Анимация «рта» Pac-Man с использованием таймера.

Логика конца игры:

- Поражение - столкновение с призраком в обычном режиме.

- Победа - сбор всех точек.

Вспомогательные классы:

- GameStats - управление очками (Score), жизнями (Lives), победами (Wins), поражениями (Losses). Интеграция с DLL PacManAsm.dll через методы AddScore и GetScore. Сохранение статистики в файл game_stats.txt.

- Ghost - режимы движения - Chase, Scatter, Frightened. Алгоритм преследования Pac-Man (CalculateChaseTarget). Возрождение призраков методом Respawn.

- Dot - Хранение позиции точки (Position). Определение типа точки - бычья или энерджайзер (IsEnergizer).

Поток данных и взаимодействие:

- Запуск игры - пользователь вводит логин/пароль в LoginForm. После успешной авторизации открывается MainMenuForm. Нажатие «Начать игру» запускает Form1.

- Сохранение статистики - при завершении игры (победа/поражение) статистика записывается в файл game_stats.txt. Данные загружаются при запуске MainMenuForm.

- Обработка ошибок - критические операции (работа с файлами) обёрнуты в try-catch. Ошибки выводятся через MessageBox.

Стиль и UI

- Анимации - плавное «открывание рта» Рас-Ман через изменение угла заливки FillPie. Призраки меняют цвет в режиме Frightened (синий).
- Адаптивность - игровое поле фиксированного размера (1000x800). Точки и стены генерируются адаптивно под размер окна.
- Локализация - все тексты на русском языке. Поддержка изменения языка через ресурсные файлы (архитектурно заложена, но не реализована).

4. Отладка и тестирование

1. Основные этапы отладки:

- Проверка генерации игрового поля - убедиться, что стены создаются корректно согласно заданным координатам (`InitializeWalls`).
- Проверить генерацию точек и энерджайзеров - точки не должны находиться внутри стен или слишком близко к ним (`IsPositionInWall`, `IsTooCloseToWall`). Энерджайзеры должны составлять 5% от общего числа точек.
- Тестирование логики движения Рас-Мана - проверить, что Рас-Ман перемещается в заданном направлении ($\leftarrow$, $\rightarrow$, $\uparrow$, $\downarrow$) без прохождения сквозь стены (`CheckWallCollision`).
- Поведение призраков - проверить режимы движения Chase, призраки должны двигаться к Рас-Ману с опережением на 4 клетки (`CalculateChaseTarget`). Scatter - призраки должны перемещаться в углы карты (`GetCornerPosition`). Frightened призраки должны менять цвет и двигаться случайно (`MoveRandomly`). Убедиться, что призраки возрождаются в начальной позиции после “съедания” (`Respawn`).
- Сбор точек и энерджайзеров - проверить, что точки удаляются при пересечении с Рас-Маном (`CheckCollisions`). Убедиться, что энерджайзеры активируют режим Frightened на 250 игровых тиков.
- Логика конца игры - поражение при столкновении с призраком в обычном режиме количество жизней уменьшается. Если жизни = 0 - игра завершается (`GameOver`). Победа - при сборе всех точек игра завершается (`WinGame`).
- Работа с файлом статистики - проверить, что файл `game_stats.txt` создается при первом запуске. Убедиться, что записи добавляются в формате plaintext. Проверить, что статистика корректно загружается при открытии `MainMenuForm`.
- Тестирование UI - проверить анимацию рта Рас-Мана (`AnimateMouth`). Убедиться, что диалоговые окна (победа/поражение) открываются в центре экрана и не вызывают зависаний. Проверить отображение статистики (счёт, жизни, победы/поражения) в реальном времени.

2. Инструменты отладки в Visual Studio:

- Установить точки останова в ключевых методах
- InitializeDots - проверка генерации точек.
- MoveGhosts - отслеживание алгоритмов движения призраков.
- CheckCollisions - проверка обработки столкновений.
- SaveStats - мониторинг записи данных в файл.

3. Пошаговое выполнение (Step Into/Over):

– Проверить последовательность действий при сборе энерджайзера, активация режима Frightened.

- Контроль переменных (Watch, Locals).
- Отслеживать - `_dots.Count` - количество оставшихся точек. `_gameStats.Lives` - текущее количество жизней. `_ghostModeTimer` - таймер смены режимов призраков.

4. Имитация редких ситуаций:

- Попробовать собрать все точки за минимальное время.
- Специально столкнуться с призраком в режиме Frightened, чтобы проверить начисление очков (200 за каждого).
- Обработка исключений (Exception Breakpoints):
- Включить остановку при необработанных исключениях (Debug → Windows → Exception Settings).
- Загрузку изображений (например, отсутствие файла текстур).
- Чтение/запись в файл `game_stats.txt` при отсутствии прав доступа.

5.Описание программы

Программа представляет собой реализацию классической аркадной игры Рас-Ман, разработанную на платформе .NET с использованием языка C#. Архитектура приложения включает три ключевые формы - окно авторизации (LoginForm), главное меню (MainMenuForm) и игровое поле (Class1). Дополнительно задействована внешняя библиотека РасManAsm.dll, отвечающая за низкоуровневые операции с игровым счетом.

Окно авторизация содержит:

Форма авторизации представленная на рисунке 1 служит точкой входа в приложение. Пользователю предлагается ввести логин и пароль в соответствующие текстовые поля. Для доступа к игре предусмотрена проверка учетных данных - по умолчанию используется комбинация логин “arsen” / пароль “12345”. При успешной аутентификации и нажатия кнопки “Войти” пользователя перенаправляет в главное меню. В случае ошибки выводится диалоговое окно с уведомлением. Кнопка “Выход” позволяет завершить работу приложения без авторизации.

Интерфейс формы выполнен в стилистике игры - желтый фон, контрастные элементы управления. Ввод пароля маскируется символами “*” для обеспечения базовой безопасности. После закрытия формы данные не сохраняются.

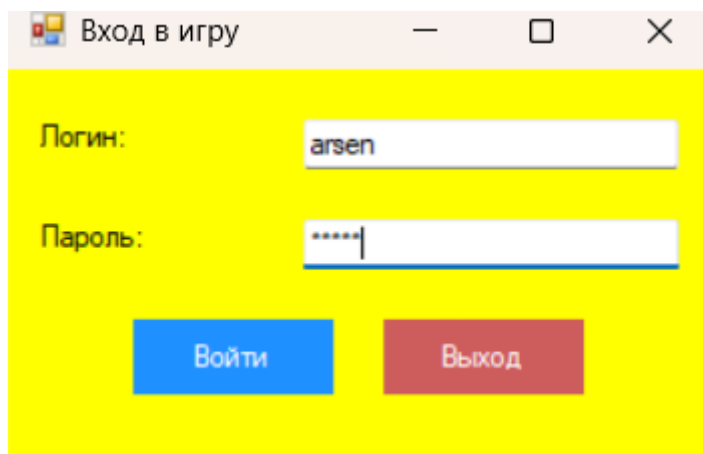


Рисунок 1 – Меню авторизации

Окно главного меню содержит:

Главное меню представленное на рисунке 2 обеспечивает навигацию между игровыми режимами и предоставляет доступ к статистике. Центральным элементом интерфейса является приветствие пользователя с отображением его логина. В верхней части экрана выводится сводка по результатам предыдущих игр, количество побед и поражений, обновляемая в реальном времени. Функционал меню реализован через одну основную кнопку “Начать игру”.

Система статистики хранит данные в текстовом файле `game_stats.txt` в корневой папке программы. Формат записи — пара ключ-значение (например, Победы=5). При каждом завершении игры (победе или поражении) файл перезаписывается, что обеспечивает актуальность данных. В случае отсутствия файла при первом запуске он создается автоматически.

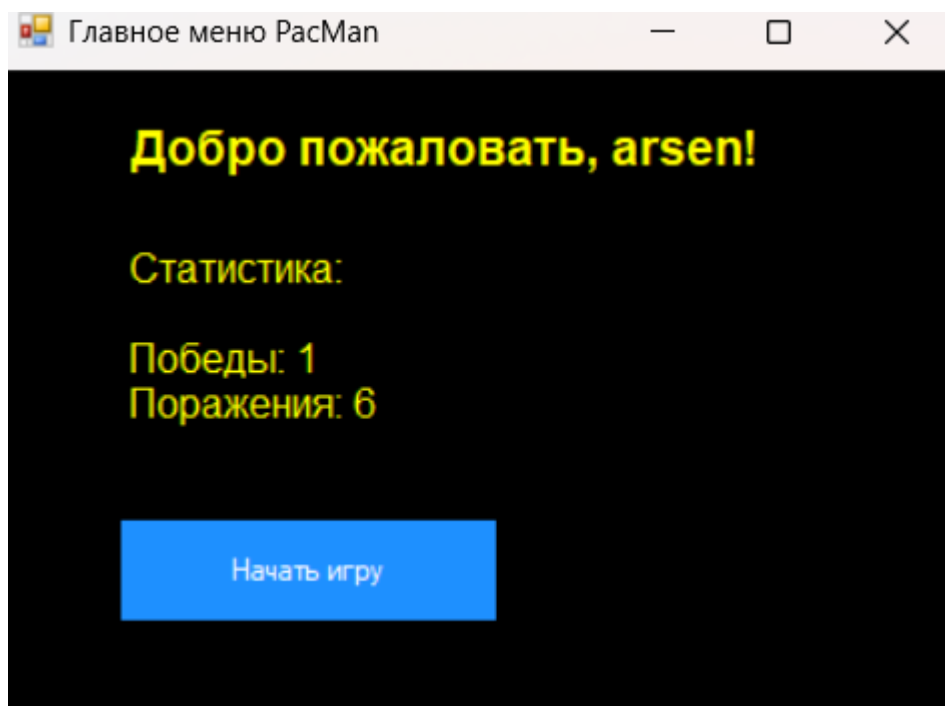


Рисунок 2 – Главное меню

Окно игрового поля:

Игровое поле представленное на рисунке 3 содержит замкнутый лабиринт, сгенерированный по строго заданным координатам, что обеспечивает идентичную структуру при каждом запуске игры. Основой лабиринта являются вертикальные и горизонтальные стены, образующие коридоры и тупики. Карта имеет симметричную структуру с несколькими ключевыми зонами, центральным перекрестком, боковыми ответвлениями и «тоннелями» по краям, позволяющими Рас-Ман телепортироваться на противоположную сторону.

Структура лабиринта содержит стены. Стены выполнены в виде прямоугольников темно-синего цвета. Их расположение прописано в коде (например, `new Rectangle(100, 100, 20, 200)` создает стену высотой 200 пикселей в левой части экрана). Стены полностью блокируют движение. Проверка столкновений реализована через пересечение прямоугольных областей.

Объекты распределяются по полю такие как точки и энеджерджайзеры. Точки (белые круги диаметром 6 пикселей) размещаются на свободных участках лабиринта с шагом 30 пикселей. Алгоритм генерации исключает их появление внутри стен или слишком близко к ним (менее 15 пикселей). Энеджерджайзеры (оранжевые круги диаметром 8 пикселей) составляют 5% от общего числа точек. Они генерируются случайным образом, но всегда располагаются в стратегически важных точках — например, возле тупиков или перекрестков. При съедании энеджерджайзера активируется режим испуга длительностью 250 кадров. В этом режиме призраки меняют алгоритм движения на случайный, а столкновение с ними приносит игроку 200 очков.

Визуальное оформление выглядит таким образом. Фон игрового поля — черный, что контрастирует с яркими элементами (Рас-Ман, призраки, точки). Рас-Ман отрисовывается как желтый сектор круга с анимированным ртом. Угол раскрытия рта варьируется от 15 до 30 градусов, создавая иллюзию движения. Анимация синхронизирована и изменяется каждые 5 кадров. Призраки имеют уникальные цвета, Blinky — красный, Pinky — розовый. В режиме испуга их окраска меняется на синий.



Рисунок 3 – Игровое поле

Завершение игровой сессии:

При завершении игры, в зависимости от результата, пользователь сталкивается с одним из двух диалоговых окон.

В случае победы — когда все точки и энерджайзеры собраны — игра приостанавливается, а движение персонажей блокируется. На экране появляется окно изображенное на рисунке 4. Здесь доступны две кнопки «Да» для перезапуска игры с сохранением статистики (счетчик побед увеличивается) и «Нет» для возврата в главное меню. При выборе «Да» позиции объектов сбрасываются, генерируется новое поле. Если игрок нажимает «Нет», статистика автоматически сохраняется в файл `game_stats.txt`, а игровое окно закрывается.

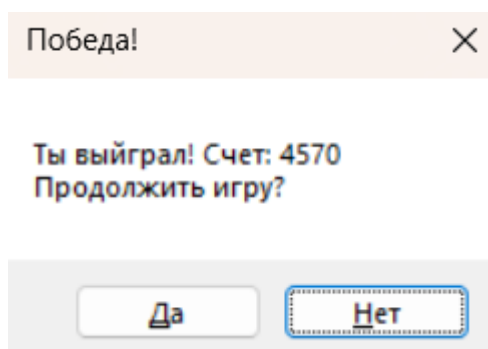


Рисунок 4 – Победа

Поражение наступает при столкновении Рас-Ман с призраком вне режима испуга или потере всех жизней. Игровой процесс прерывается, и на экране возникает диалоговое окно изображенное на рисунке 5. Кнопки «Да» и «Нет» работают аналогично, рестарт восстанавливает жизни, но не пересоздает точки, а выход сохраняет поражение в статистику.

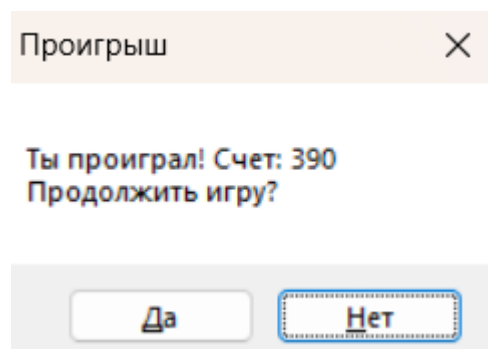


Рисунок 5 - Поражение

Функция InitializeGame сбрасывает позицию Рас-Мана, жизни, очки, генерирует стены, точки и призраков (Рисунок.6).

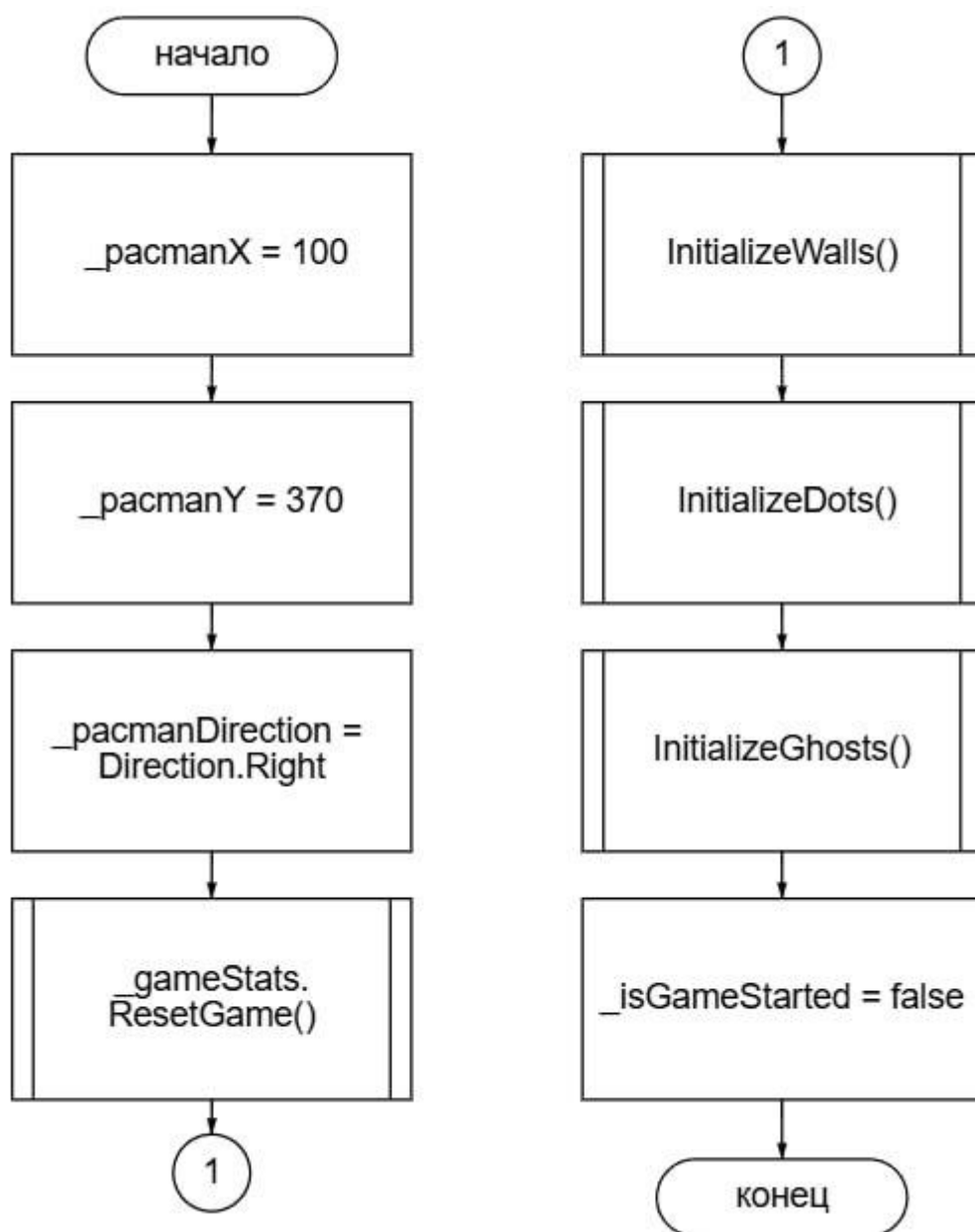


Рисунок 6 - схема функции InitializeGame

Функция AnimateMouth отвечает за анимацию рта Рас-Мана (Рисунок.7).

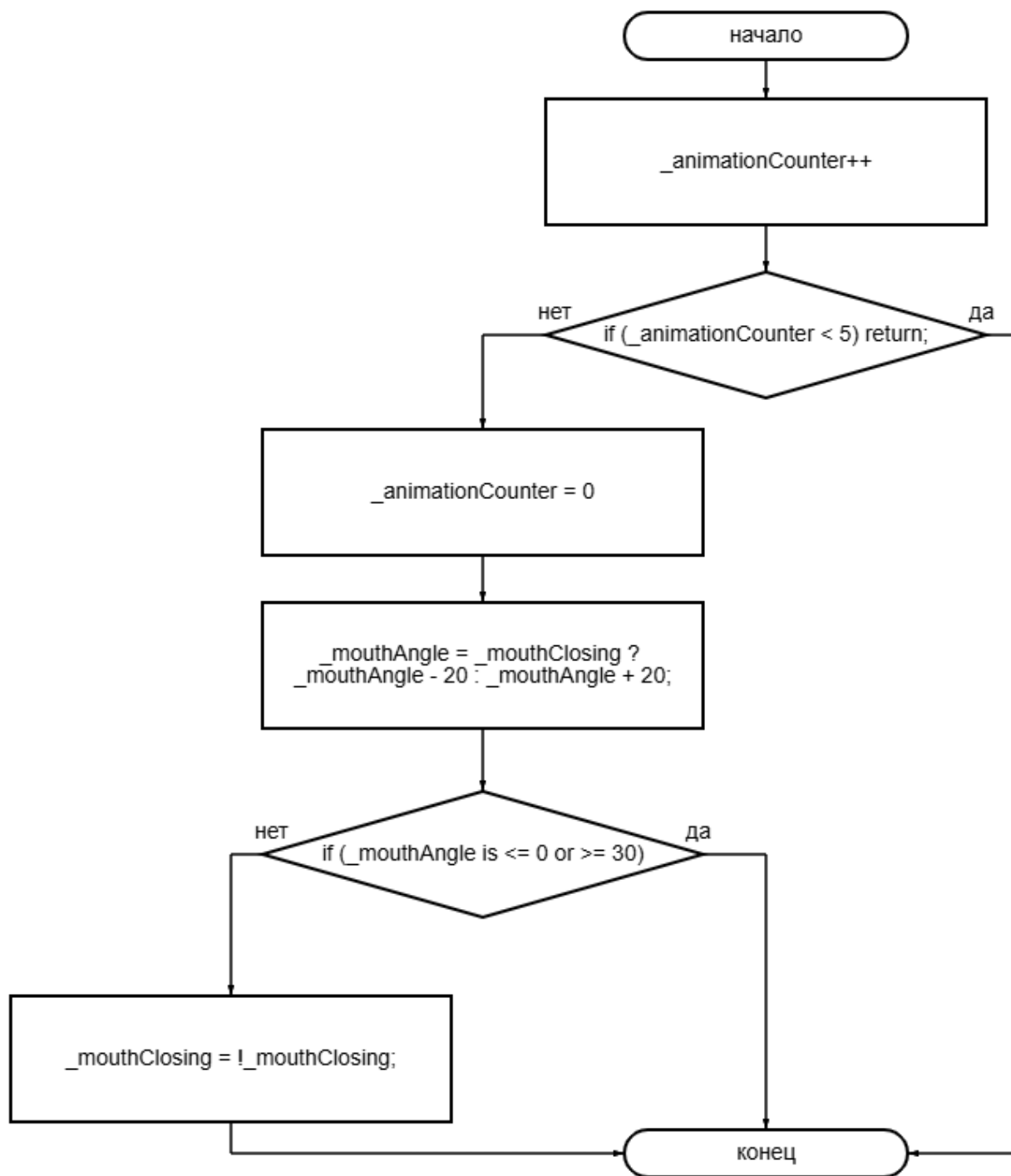


Рисунок 7 - схема функции AnimateMouth

Функция `GameTimer_Tick` обновляет позиции объектов, проверяет коллизии, анимирует элементы (Рисунок.8.).

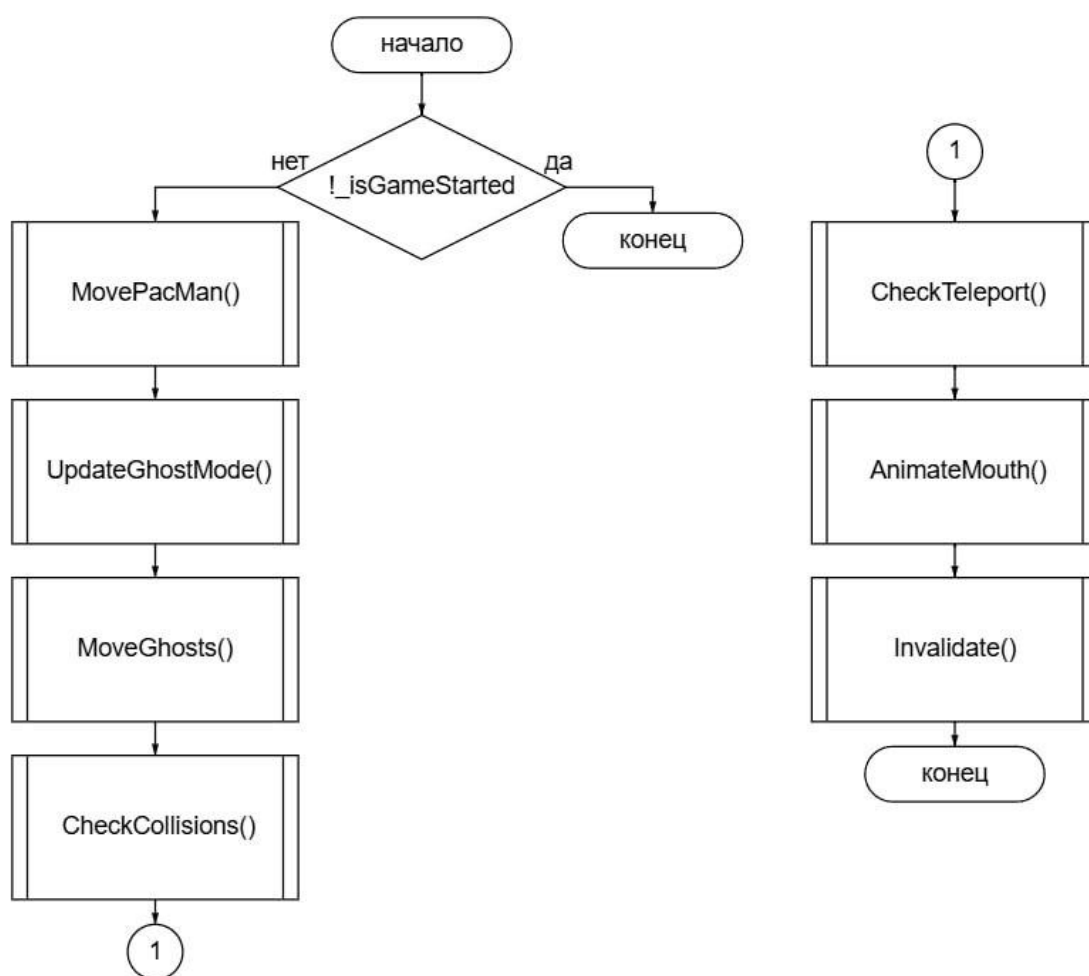


Рисунок 8 - схема функции `GameTimer Tick`

Функция UpdateGhostMode отвечает за переключение режимов поведения призраков (Рисунок.9).

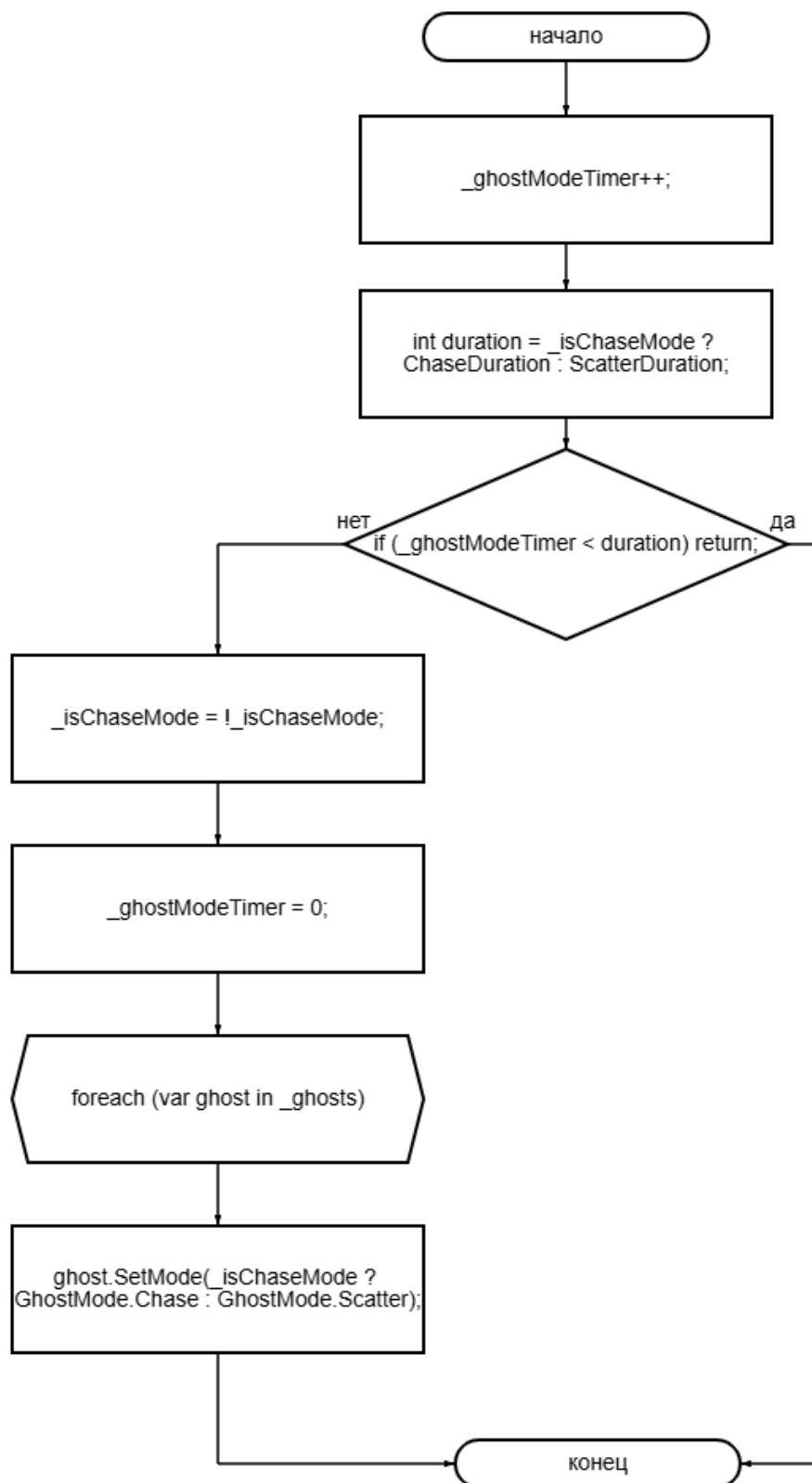


Рисунок 9 - схема функции UpdateGhostMode

Функция CheckCollisions проверяет сбор точек, столкновения с призраками, активацию энерджайзеров (Рисунок.10(a),10(b),10(c)).

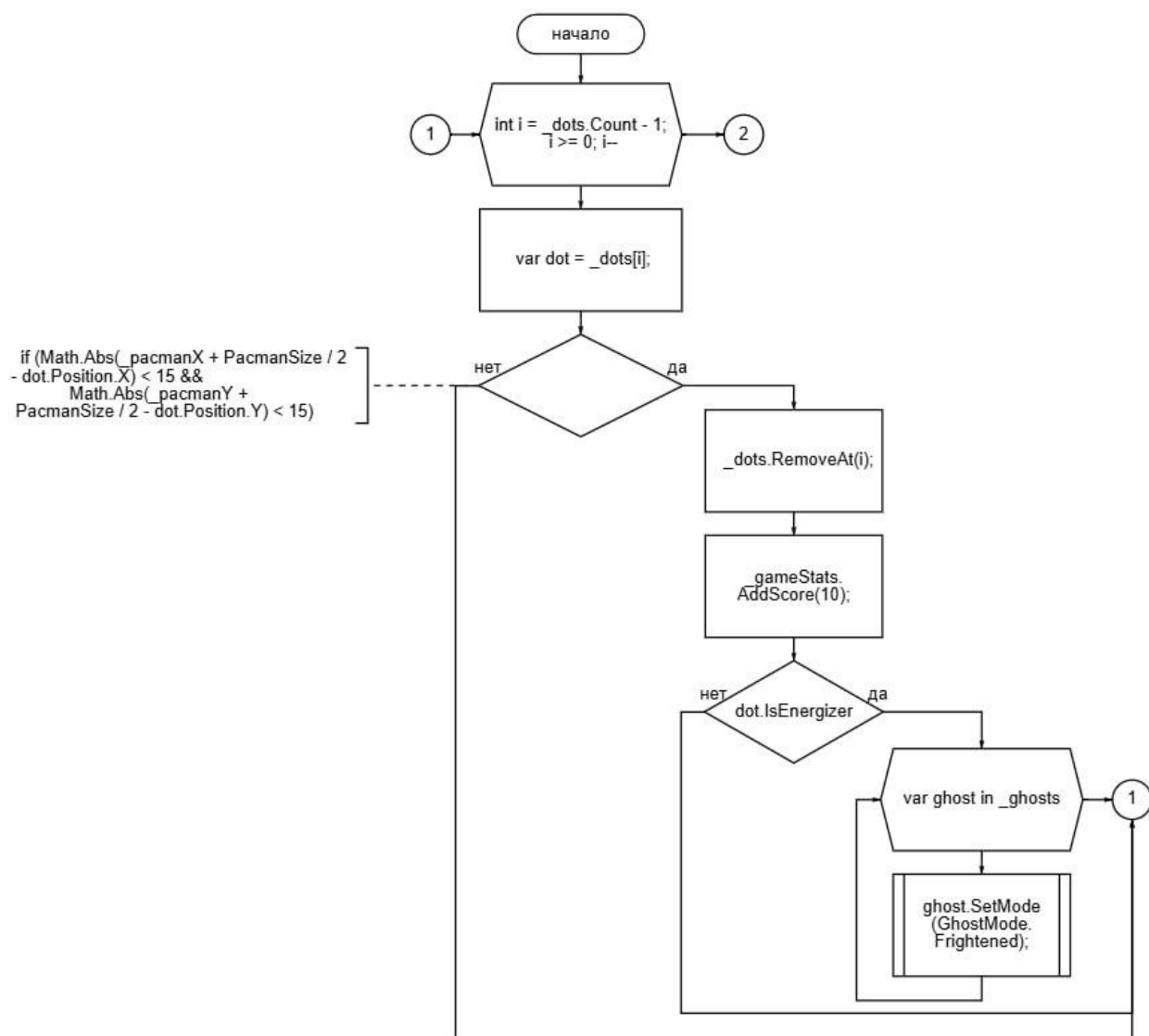


Рисунок 10 - а - схема функции CheckCollision

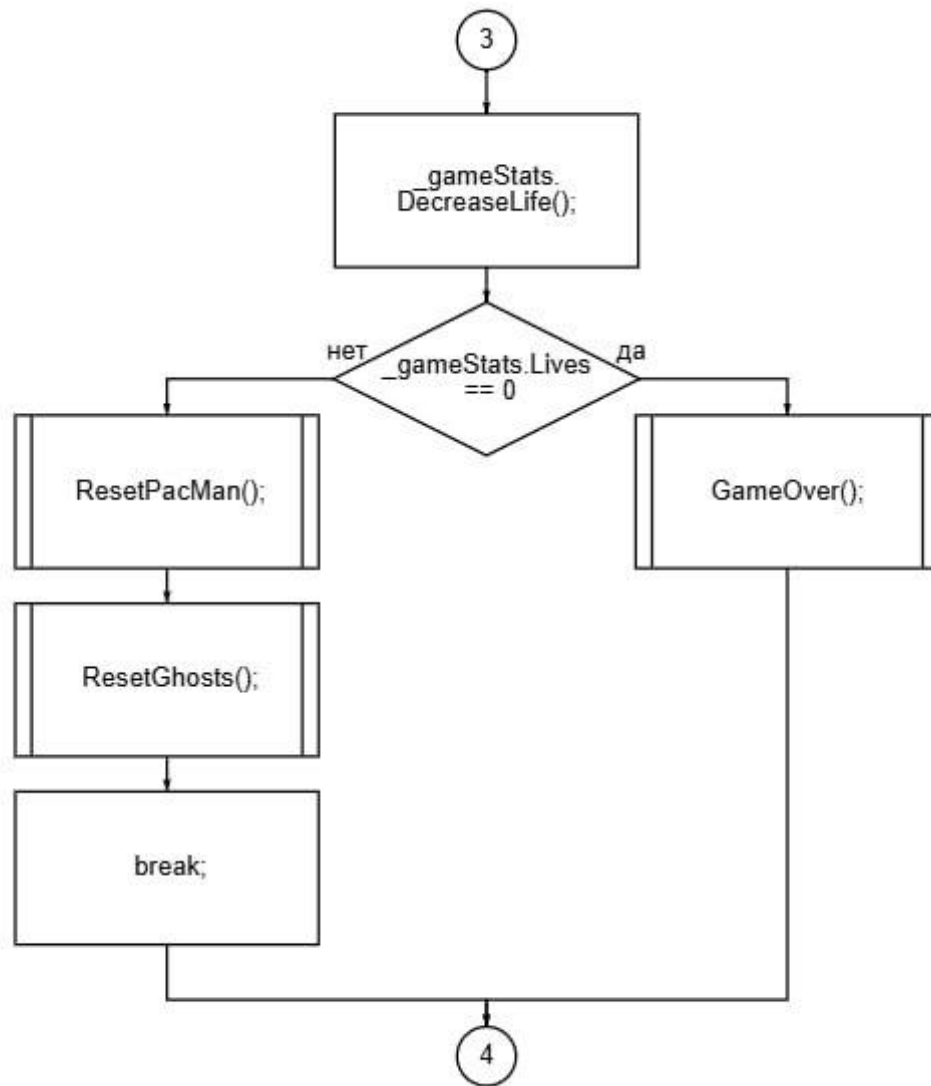


Рисунок 10 - б - схема функции `CheckCollision`

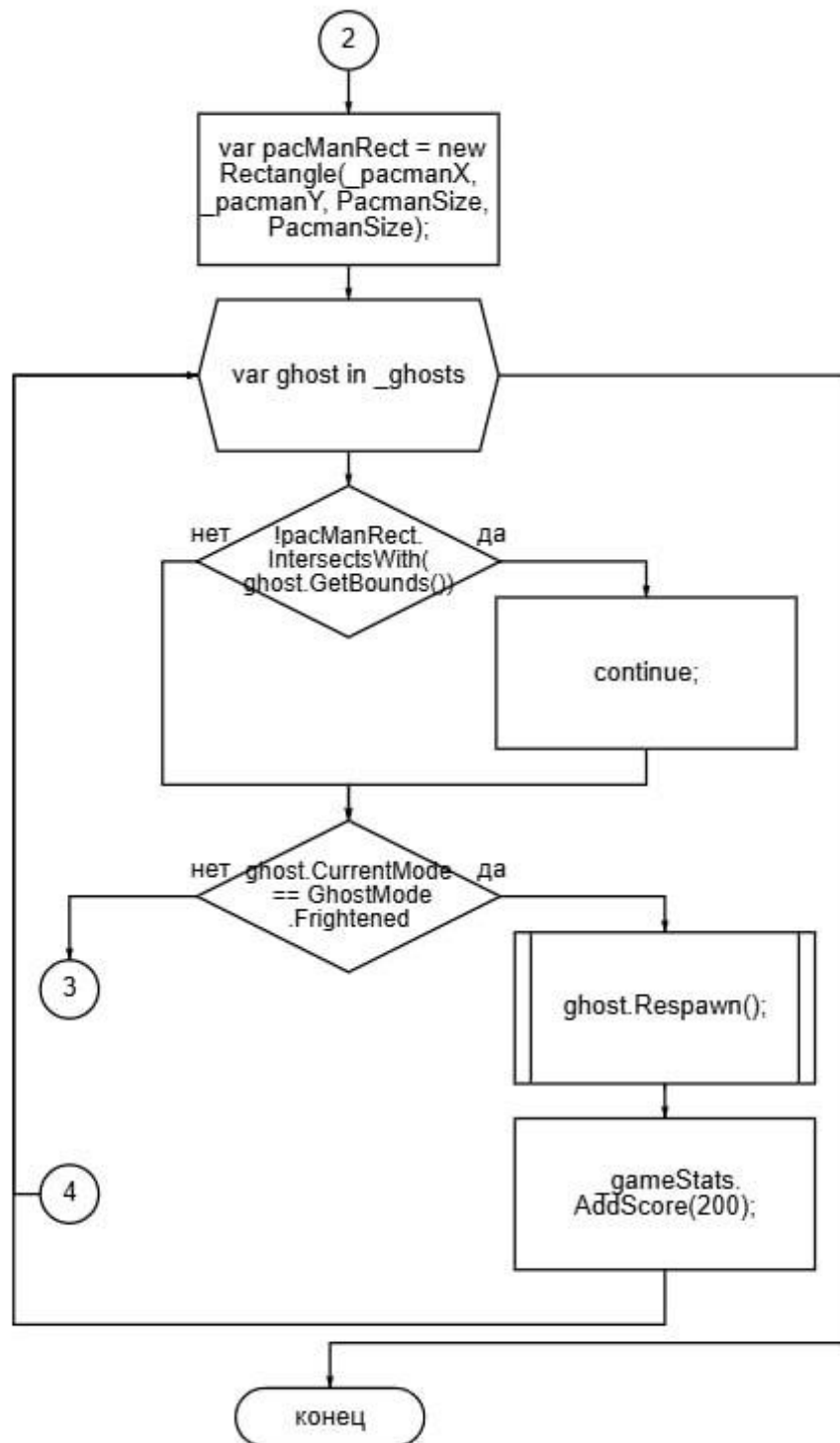


Рисунок 10 - с - схема функции CheckCollision

Функция MovePacMan для движения Рас-Мап (Рисунок.11).

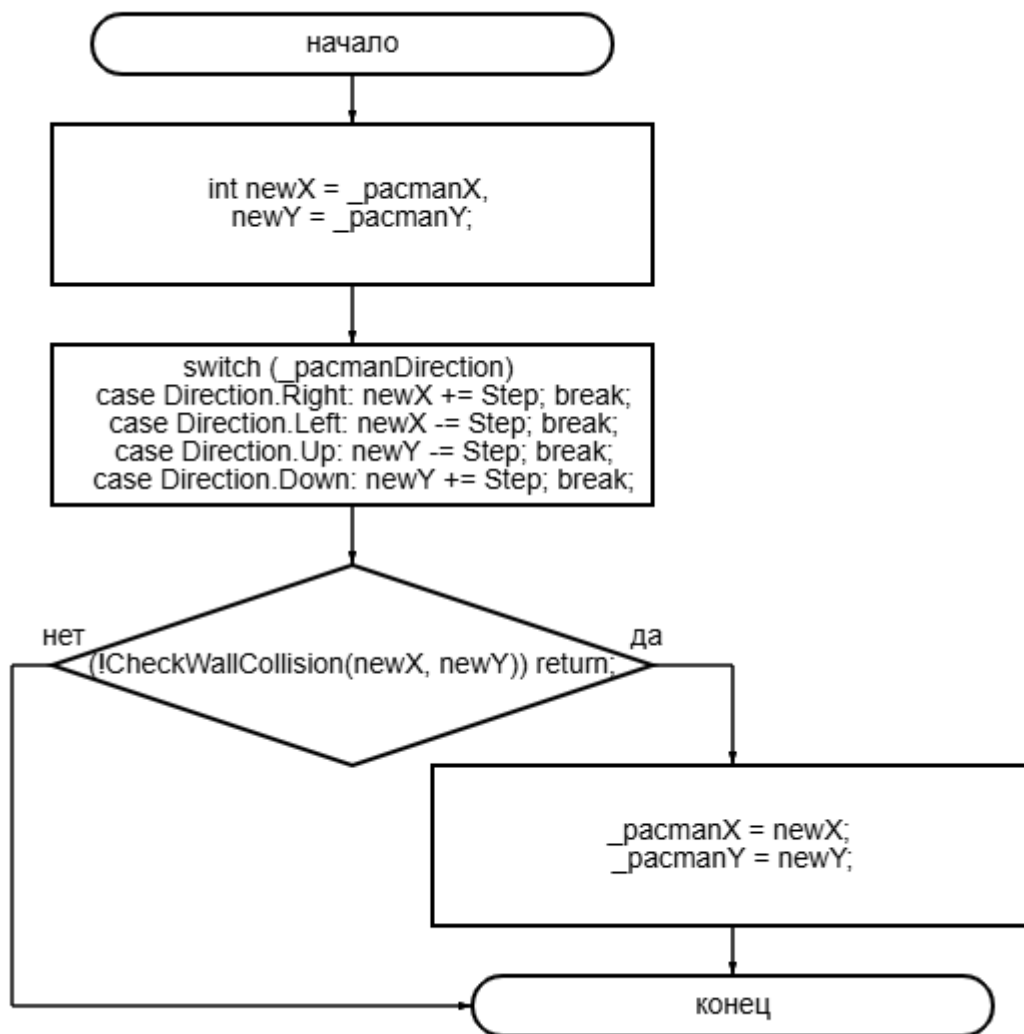


Рисунок 11 - схема функции MovePacMan

Функция Move отвечает логику движения всех призраков в зависимости от режима (Рисунок.12).

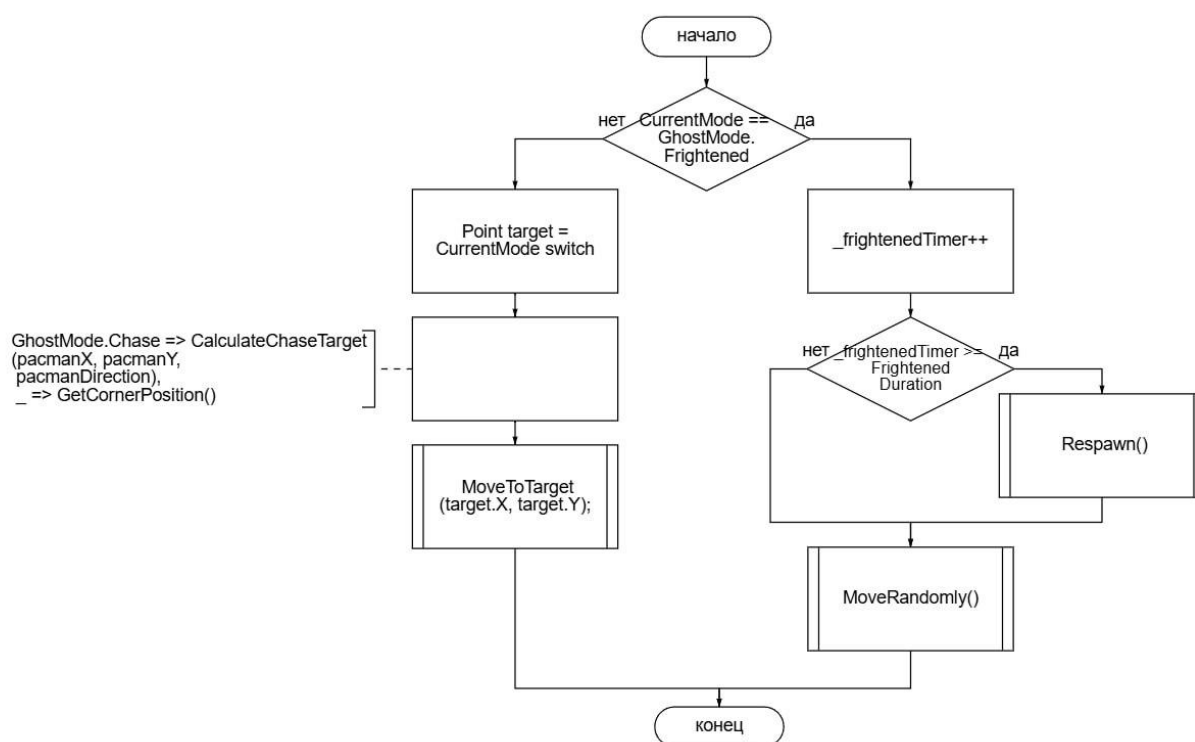


Рисунок 12 - схема функции Move

Функция LoadStats для загрузки сохранения статистику побед\поражений (Рисунок.13).

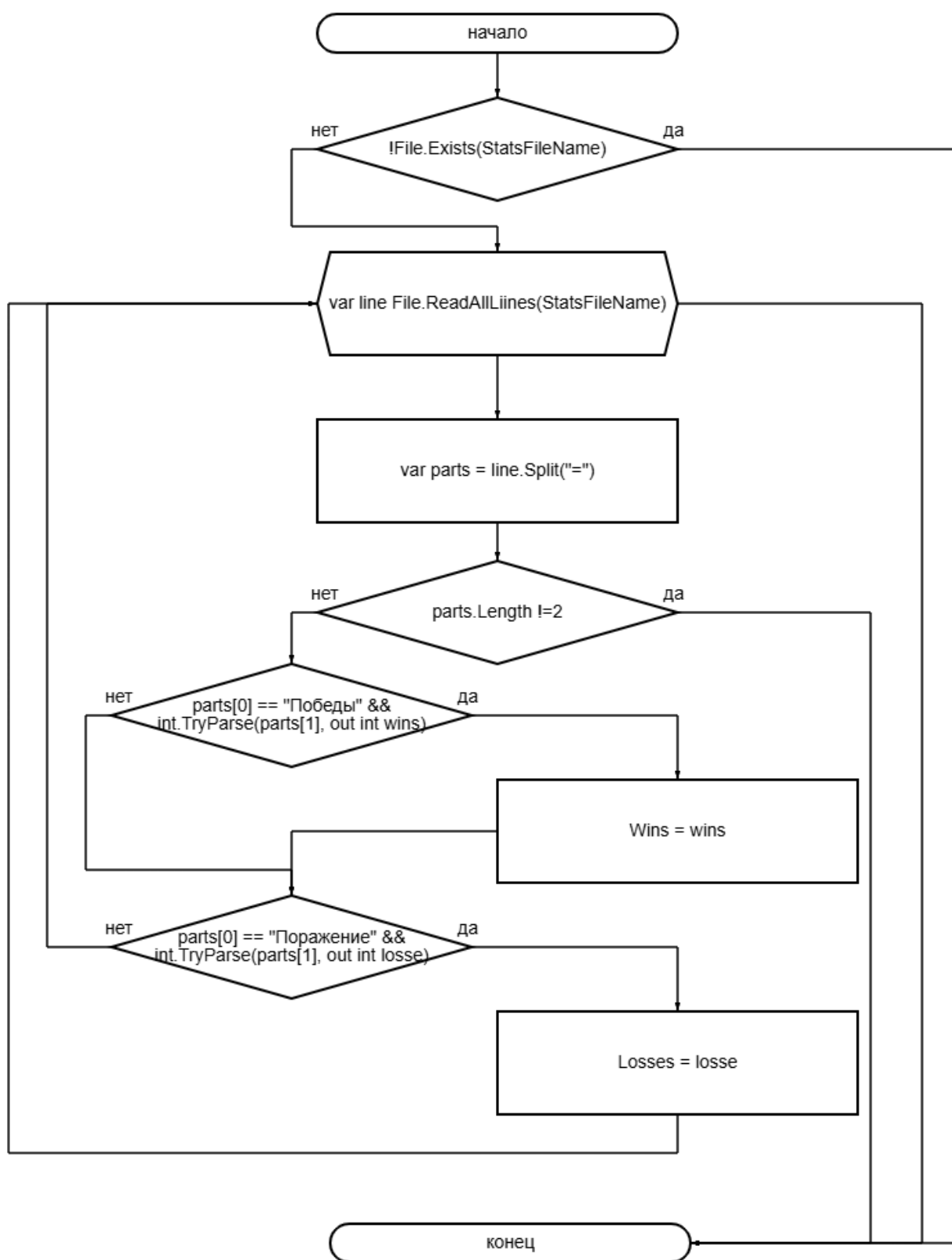


Рисунок 13 - LoadStats

Функция SaveStats для сохранения статистики и перезапуска игры (Рисунок.14).

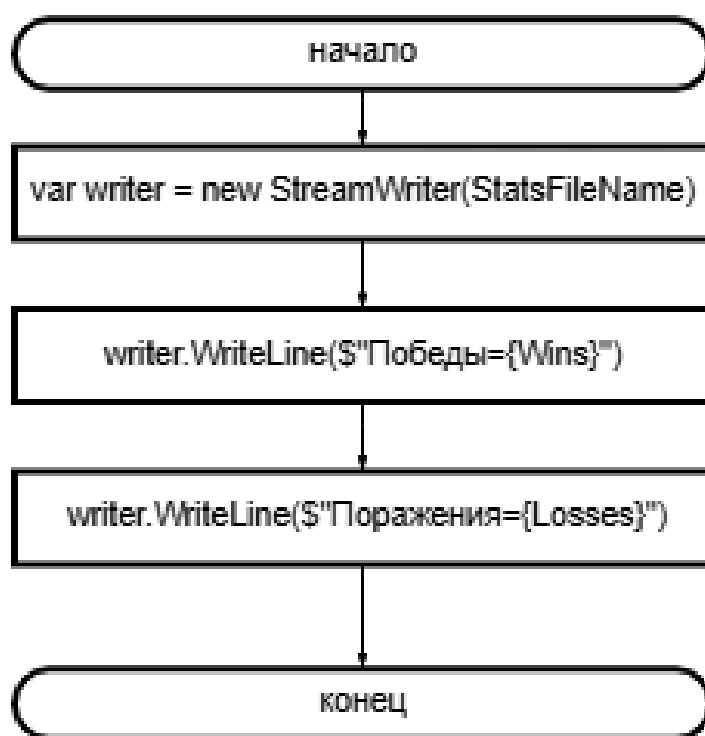


Рисунок 14 - SaveStats

Функция GameOver для завершения игры при проигрыше (Рисунок.15)

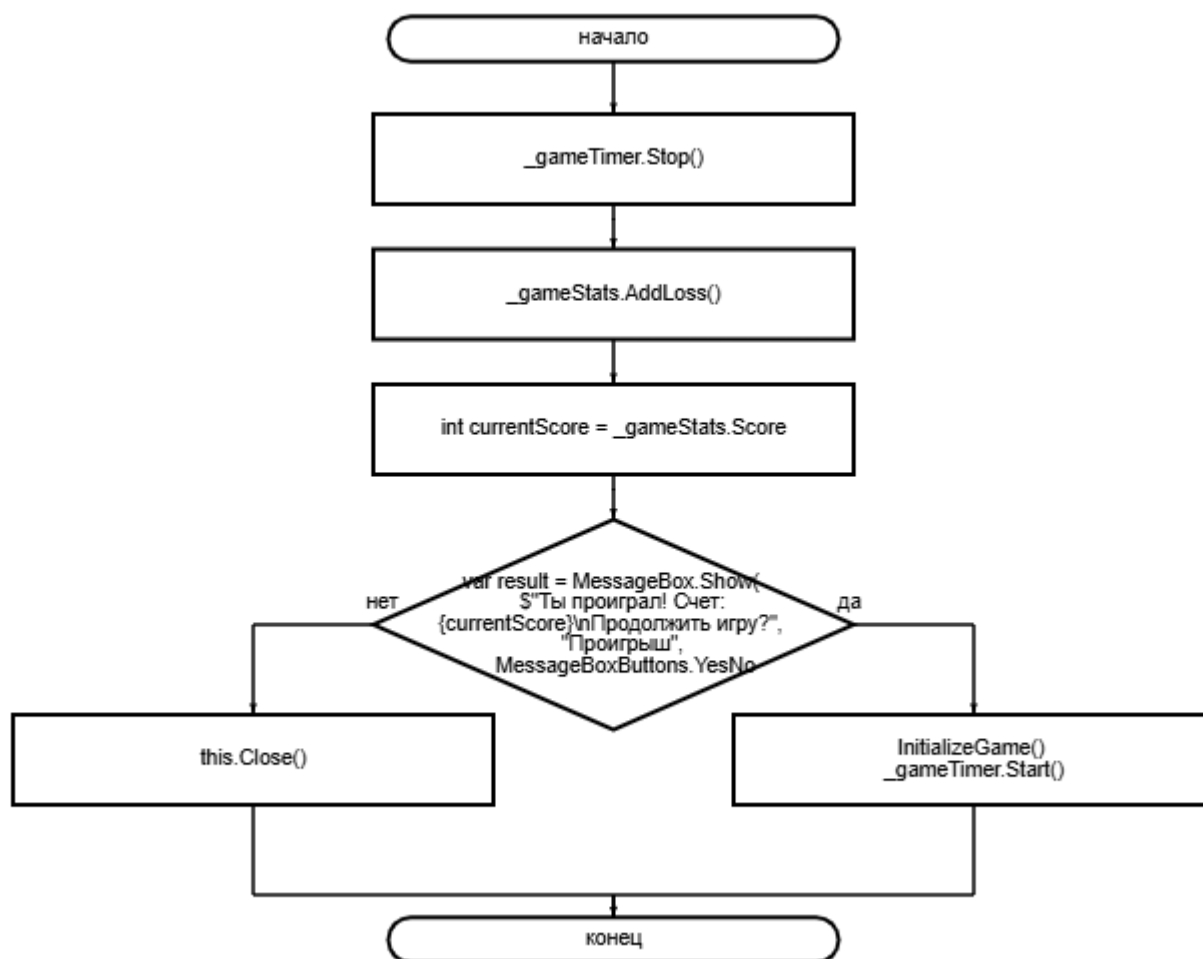


Рисунок 15 - схема функции GameOver

Функция WinGame для завершения игры при проигрыше (Рисунок.16).

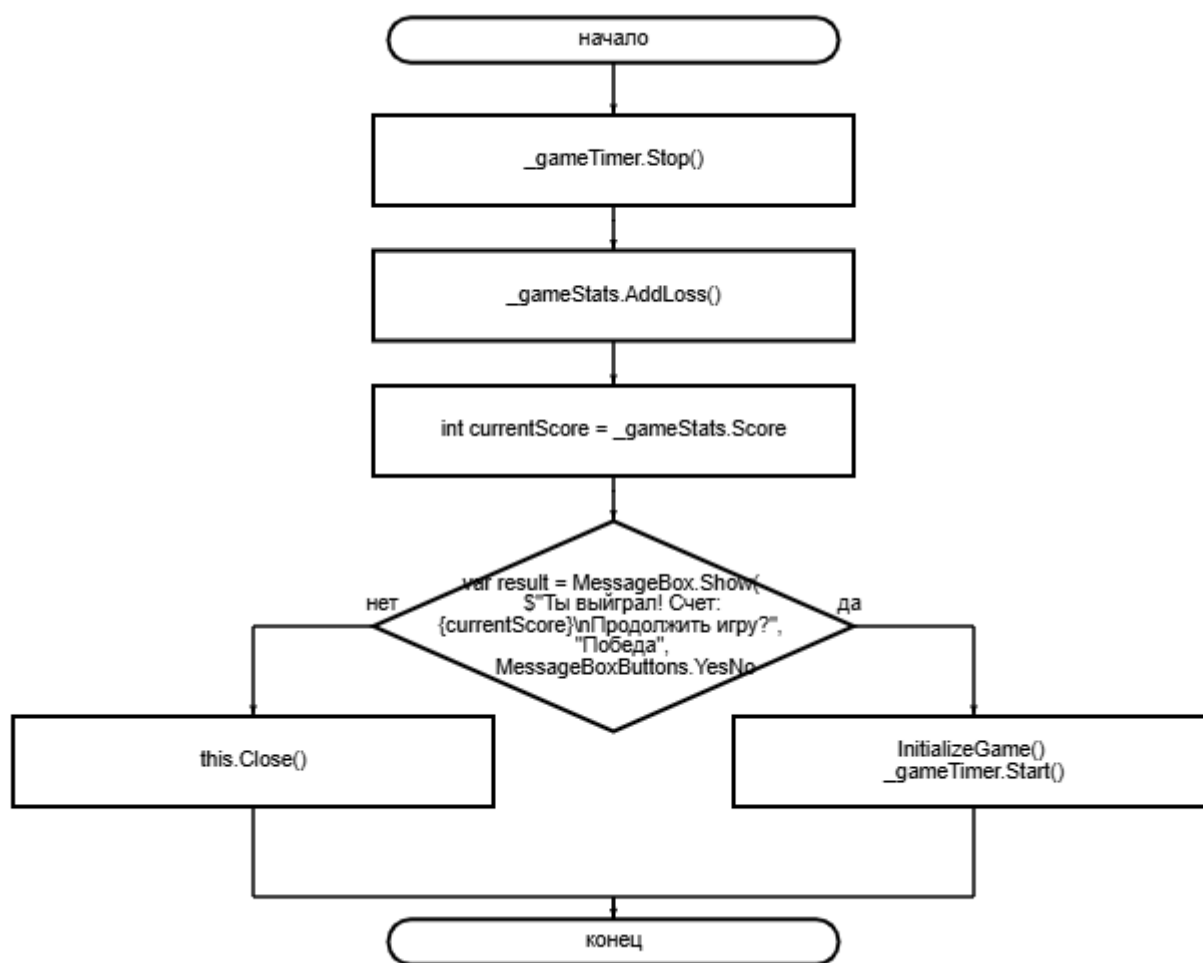


Рисунок 16 - схема функции WinGame

6. Руководство пользователя

Запуск и авторизация запустите файл RasManGame2.exe. Откроется окно авторизации с полями для ввода логина и пароля рисунок 1. Введите логин “arsen” Пароль “12345” Нажмите «Войти». При успешной авторизации откроется главное меню рисунок 2. В главном меню приветствие с именем пользователя. Статистика, количество побед и поражений. Кнопка «Начать игру».

Игровой процесс нажмите «Начать игру». Откроется игровое поле Рас-Ман управляется стрелками. Цель собрать все точки, избегая призраков. Энерджайзеры (крупные точки) - временно делают призраков уязвимыми. Элементы интерфейса - счет отображается в верхнем левом углу. Жизни 3 попытки (отображаются справа). Механика столкновение с призраком в обычном режиме — потеря жизни. В режиме испуга — призрак исчезает, +200 очков.

Завершение игры победа рисунок 4 собраны все точки. Появится окно с предложением, «Новая игра» — рестарт. «В меню» — возврат к настройкам. Поражение рисунок 5 закончились жизни. Отобразятся кнопки, «Да» — рестарт с 3 жизнями. «Нет» — закрытие игры. Статистика (победы/поражения) автоматически сохраняется в game_stats.txt.

Заключение

Разработка игры Pac-Man на C# и Windows Forms стала для меня важным этапом освоения не только языка программирования, но и принципов создания игровых приложений.

Приобретённые навыки - работа с графикой и анимацией, реализация анимации рта Pac-Man через FillPie с динамическим изменением угла. Отрисовка стен, точек и энерджайзеров с использованием GDI+. Визуализация призраков с учетом их режимов (Chase, Scatter, Frightened). Сохранение статистики в файл game_stats.txt.

Реализация ИИ призраков с алгоритмами преследования, патрулирования и убегания.

Механика сбора точек и энерджайзеров с подсчётом очков через DLL (PacManAsm.dll).

Подключение ассемблерной DLL для работы с игровыми очками (AddScore, GetScore).

Большой урок я извлёк из отладки - научился использовать точки останова, следить за состоянием переменных и методично искать причины неправильного поведения программы. Это оказалось не менее важно, чем написание самого кода.

Проект также научил меня – модульности, классы GameStats, Ghost, Dot позволили организовать код.

В обработке исключений, защита операций с файлами (статистика) через try-catch. Оптимизации - использование DoubleBuffered = true для устранения мерцания графики.

Создание Pac-Man стало для меня возможностью глубоко погрузиться в основы игровой разработки. Я освоил работу с графикой, алгоритмами ИИ и внешними библиотеками, а также научился проектировать логику, ориентированную на пользовательский опыт. Этот проект подтвердил, что даже классические игры требуют тщательной проработки деталей — от коллизий до визуальной обратной связи. Полученные навыки станут основой для более сложных проектов в будущем.

Список используемых источников

1. CLR via C# / Джеффри Рихтер. – Москва : Питер, 2022. – 896 с. – ISBN 978-5-4461-2095-6.(дата обращения 20.04.2025).
2. Microsoft Learn : Документация по C# : [сайт]. – URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp> (дата обращения: 15.05.2025).
3. C# 10 in a Nutshell / Джозеф Албахари, Бен Албахари. – O'Reilly Media, 2022. – URL: <https://www.oreilly.com> (дата обращения: 05.05.2025). – подписка.