

**Penza State University**  
**Faculty of Computer Engineering**  
**Department of Computer Engineering**

**COURSE PROJECT**

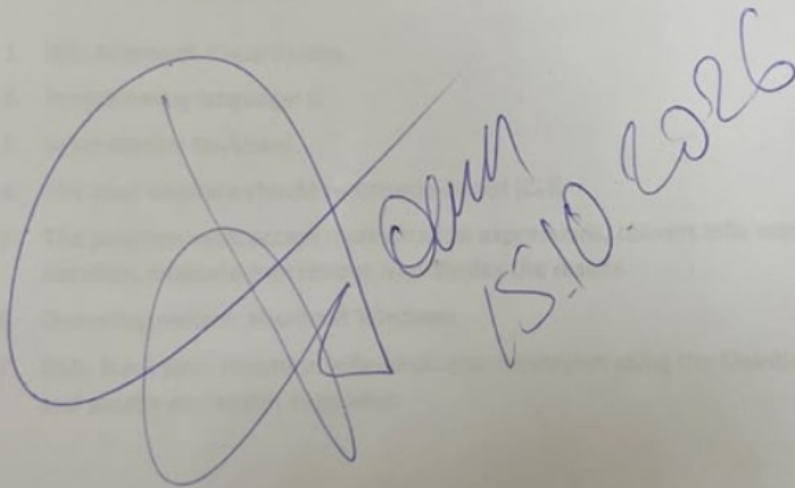
**EXPLANATORY NOTE**

for course design for the course

*«Logic and fundamentals of algorithmization in engineering problems»*

on the topic

**Complex program structure «Calculator Using Stacks»**

A handwritten signature in blue ink, which appears to be 'M. Ma' (likely Mohammed Ma), is written over the printed text. To the right of the signature, the date '15.10.2026' is written in blue ink.

Completed: Mohammed Ma

Student of group: 24VVIA

Accepted: Fedjunin R.N

Penza 2026

Head Department of VT

M. S. Karavysheva

- 2024 year

for the course task

Student Mohammed Maai, Group 24VVIA1

**Initial data (technical requirements) for design** Develop a console-based calculator program for evaluating mathematical expressions using the stack data structure.

1. IDE: Microsoft Visual Studio.
2. Programming language: C.
3. Input device: Keyboard.
4. The user interface should be console-based (CLI).
5. The program must accept mathematical expressions, convert infix expressions into postfix notation, evaluate expressions, and display the results.
6. Operating system: Microsoft Windows.
7. DSA: Stack data structure, infix-to-postfix conversion using the Shunting Yard algorithm, and postfix expression evaluation.

### Scope of work for the course

#### Algorithmic part

- Study the stack data structure and its operations.
- Develop an algorithm for converting infix expressions into postfix notation.
- Develop an algorithm for evaluating postfix expressions.
- Implement the calculator program in C.

#### ☐ The grafical part

- Program flowchart.
- System operation diagram.
- Hierarchical structure of the program.
- Program interaction diagram.
- Diagram illustrating stack operations and expression evaluation.

#### Experimental part

- Compile and run the program.
- Test arithmetic operations and operator precedence.
- Test expressions containing parentheses and decimal numbers.
- Debug the program and verify the correctness of the results.

#### Project completion time by section

According to schedule.

The date of issue of the task is "October 14", 2026.

Project defense date " \_\_\_\_\_"

Teacher Fediunin R.N. \_\_\_\_\_

Received the task on Oct 14, 2026.

Student Mohammed Maai. \_\_\_\_\_

## Content

|                                                            |   |
|------------------------------------------------------------|---|
| Introduction.....                                          |   |
| Abstract.....                                              |   |
| Keywords.....                                              |   |
| Subject studied, goal, methods and result.....             |   |
| 1. Statement of the problem.....                           |   |
| 1.1 Input.....                                             |   |
| 1.2 Output.....                                            |   |
| 1.3 Functional requirements.....                           |   |
| 1.4 Non-functional requirements.....                       |   |
| 2. Choosing a solution.....                                |   |
| 2.1 Theoretical background.....                            |   |
| 2.2 Stack data structures.....                             |   |
| 2.3 Infix and postfix notation.....                        |   |
| 2.4 Operator precedence.....                               |   |
| 2.5 Shunting Yard algorithm.....                           |   |
| 3. Description of program development.....                 |   |
| 3.1 General program workflow.....                          |   |
| 3.2 Conversion algorithm: Infix $\rightarrow$ Postfix..... |   |
| 3.3 Postfix evaluation algorithm.....                      |   |
| 3.4 Program modules and functions.....                     |   |
| 3.5 Data structures and implementation details.....        | 1 |
| 3.6 Complexity analysis.....                               | 1 |
| 4. Debugging and testing.....                              | 1 |
| 4.1 Testing approach.....                                  | 1 |
| 4.2 Test cases.....                                        | 1 |
| 4.3 Test summary.....                                      | 1 |
| 4.4 Manual verification.....                               | 1 |
| 4.5 Cross-check with program output.....                   | 1 |
| 5. Program Description.....                                | 1 |
| 5.1 Program startup and user interaction.....              | 1 |
| 5.2 Expression input.....                                  | 1 |

|                                    |   |
|------------------------------------|---|
| 5.3 Postfix conversion.....        | 1 |
| 5.4 Postfix evaluation.....        | 1 |
| 5.5 Result and error handling..... | 1 |
| 6. User manual.....                | 1 |
| 6.1 Supported input.....           | 1 |
| 6.2 Example.....                   | 1 |
| 6.3 User recommendations.....      | 1 |
| Conclusion.....                    | 1 |
| List of sources used.....          | 1 |
| Appendix A. Program Listings.....  | 1 |

## Abstract

This course project presents the development of a console calculator program in C. The calculator evaluates arithmetic expressions in infix notation by first converting them into postfix notation using the Shunting Yard algorithm, and then evaluating the postfix expression with a separate operand stack. The calculator also uses a stack structure to store operators and parentheses and a switch statement to manage operators. The developed program supports multi-digit numbers, floating-point values, arithmetic operators, parentheses, and operator precedence. The project demonstrates the practical application of stack data structures, expression parsing, modular programming, debugging, and testing.

## Keywords

C programming, calculator, stack, Shunting Yard algorithm, infix notation, postfix notation, expression evaluation, operator precedence, parsing.

## Subject studied, goal, methods and results

Subject studied: algorithms and data structures for parsing arithmetic expressions.

Goal: to implement a functional calculator that uses the Shunting Yard algorithm to parse and evaluate arithmetic expressions.

## Conclusion

During this coursework, the following skills were improved:

- Implementation of stack data structures
- Understanding of the Shunting Yard algorithm
- Expression parsing techniques
- Modular program design in C
- Debugging and testing console applications

As a result, a functional calculator program was developed that evaluates mathematical expressions by converting infix notation to postfix notation and then processing the postfix expression with a stack.

The project also demonstrates how a relatively simple set of data structures and algorithms can be combined into a complete console application. The use of two different stacks separates operator processing from numerical evaluation and makes the program structure easier to understand.

The program can be improved by:

- Adding a graphical interface
- Supporting more operators
- Improving input validation and error handling
- Implementing prefix conversion
- Adding more extensive automated tests

### List of sources used

34. Wikipedia – Shunting Yard Algorithm. URL:  
[https://en.wikipedia.org/wiki/Shunting\\_yard\\_algorithm](https://en.wikipedia.org/wiki/Shunting_yard_algorithm) (accessed date: 12.05.2026).
35. GeeksforGeeks – Expression Evaluation Using Stack. URL:  
<https://www.geeksforgeeks.org/expression-evaluation-using-stack/> (accessed date: 12.05.2026).
36. Kernighan, B. W.; Ritchie, D. M. The C Programming Language. 2nd ed. Englewood Cliffs, N. Prentice Hall, 1988. 272 p.
37. Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. Introduction to Algorithms. 3rd ed. MIT Press, 2009.
38. Sedgewick, R.; Wayne, K. Algorithms. 4th ed. Addison-Wesley, 2011.
39. Knuth, D. E. The Art of Computer Programming, Volume 1: Fundamental Algorithms. 3rd ed. Addison-Wesley, 1997.