Lomonosov Moscow State University
Faculty of Computational Mathematics and Cybernetics

Report for Assignment No. 6

**"Building multi-module programs.
Computing equation roots and definite integrals."**

Variant 5 / 4 / 3

Prepared by:
Student of Group 101
Antipov G. O.

Instructor:
Kuzmenkova E. A.

Moscow
2025

Содержание

1. Problem Statement **2**

2. Mathematical Justification **3**

2..1 Combined method (secants and tangents) . . . 3

2..2 Simpson's Method . . . 5

2..3 Choice of ε_1 and ε_2 . . . 6

2..4 Justification for the internal tolerances ε_1 and ε_2 . . . 6

2..4.1 Justification for ε_1 . . . 7

2..4.2 Justification for ε_2 . . . 7

3. Experimental Results **8**

4. Program Structure and Function Specification **9**

4..1 Overall architecture . . . 9

4..2 Detailed description of project files . . . 9

4..2.1 compute.c . . . 9

4..2.2 functions.h . . . 10

4..2.3 test_mode.c . . . 10

4..2.4 main.c . . . 10

4..2.5 func.asm . . . 10

4..2.6 Graphical representation of the structure . . . 10

5. Program Build (Makefile) **12**

5..1 Compilers and flags . . . 12

5..1.1 C compilation flags . . . 12

5..1.2 NASM flags . . . 12

5..2 Build targets . . . 12

5..3 Build rules . . . 13

5..4 Features . . . 13

5..5 Module dependencies . . . 13

5..6 Build specifics . . . 14

6. Debugging and Testing the Program **15**

6..1 Function $f_1(x) = x^2 + 5x + 6$ . . . 15

6..2 Function $f_2(x) = \sin x$ . . . 15

6..3 Function $f_3(x) = x^3 - 1$ . . . 15

6..4 Choice of intervals for root search . . . 16

6..5 Conclusions . . . 16

7. Analysis of Mistakes Made **17**

References **18**

1. Problem Statement

In this work, we compute the area of a plane figure bounded by three curves:

$$f_1(x) = 0.35x^2 - 0.95x + 2.7 \qquad f_2(x) = 3x + 1 \qquad f_3(x) = \frac{1}{x+2}$$

To solve the problem, we perform the following steps:

- Determine the integration limits analytically by finding the intersection intervals for each pair of functions;
- Implement the combined method (secant and tangent method) to find the x-coordinates of the intersection points of the curves;
- Implement Simpson's method for numerical computation of the area of the given figure.
- Test the implemented functions to validate their correctness.

2. Mathematical Justification

This section analyzes the methods used: the combined method (secants and tangents) for root finding and Simpson's method for numerical integration.

2..1 Combined method (secants and tangents)

Suppose we need to find a root of the equation $F(x) = f_1(x) - f_2(x) = 0$ on the interval $[a, b]$. According to [1], for the combined method to converge, the function on the chosen interval must satisfy:

- $F(x)$ is continuous and continuously differentiable on $[a, b]$;
- $F(a)F(b) < 0$ (the root lies within the interval);
- $F'(x)$ is monotonic and does not change sign on $[a, b]$.

Algorithm of the method:

At each step k we have the current interval $[a_k, b_k]$ with $F(a_k)F(b_k) < 0$. To construct the next interval:

1. Build the secant-method approximation:

$$x_k^{(\text{chor})} = \frac{a_k\, F(b_k) - b_k\, F(a_k)}{F(b_k) - F(a_k)} \tag{1}$$

2. Determine the point d_k for the tangent method:

$$d_k = \begin{cases} b_k, F'(x)F''(x) > 0 \text{ on } [a_k, b_k] \\ a_k, \text{otherwise} \end{cases} \tag{2}$$

   Compute the approximation

$$x_k^{(\text{sec})} = d_k - \frac{F(d_k)}{F'(d_k)} \tag{3}$$

3. Choose the next interval $[a_{k+1}, b_{k+1}]$ so that $F(a_{k+1})\, F(b_{k+1}) < 0$, for example

$$[a_{k+1}, b_{k+1}] = \left[\min(x_k^{(\text{chor})}, x_k^{(\text{sec})}), \max(x_k^{(\text{chor})}, x_k^{(\text{sec})})\right]$$

The process stops when $|b_{k+1} - a_{k+1}| < \varepsilon_1$.

Choice of initial intervals:

The intervals for root search are determined by analyzing the function plots (Fig. 1) and verifying the conditions:

1. $F(a)F(b) < 0$
2. $F'(x) \neq 0$
3. $F'(x)$ is monotonic on $[a, b]$

Where $F(x) = f_i(x) - f_j(x), i \neq j$. After analyzing the graphs, we obtained the following intervals for each pair of functions:

$$\begin{aligned} f_1(x) = f_2(x) &\to [0, 1] \\ f_1(x) = f_3(x) &\to [-1.96, 0] \\ f_2(x) = f_3(x) &\to [-1, 0] \end{aligned}$$

Let us check that they satisfy all necessary conditions of the method.

Mathematical justification of the chosen root-search intervals

To apply the combined method (secants and tangents), one must choose intervals $[a, b]$ such that:

1. The function $F(x) = f_i(x) - f_j(x)$ is continuous on $[a, b]$.
2. $F(a)F(b) < 0$, i.e., the function values at the endpoints have different signs — by the Bolzano–Cauchy theorem, there is at least one root on this interval.
3. $F'(x)$ does not vanish and does not change sign on $[a, b]$ (monotonic derivative).

We verify these conditions for each function pair:

1. $f_1(x)$ and $f_2(x)$

Consider $F_{12}(x) = f_1(x) - f_2(x) = 0.35x^2 - 3.95x + 1.7$.

- $F_{12}(x)$ is continuous on all of $\mathbb{R}$.
- Check the signs on $[0, 1]$:
  $F_{12}(0) = 0.35 \cdot 0^2 - 3.95 \cdot 0 + 1.7 = 1.7$
  $F_{12}(1) = 0.35 \cdot 1^2 - 3.95 \cdot 1 + 1.7 = 0.35 - 3.95 + 1.7 = -1.9$
- $F_{12}(0) \cdot F_{12}(1) = 1.7 \cdot (-1.9) < 0$, hence there is a root on $[0, 1]$.
- Derivative $F'_{12}(x) = 0.7x - 3.95$ on $[0, 1]$:
  $F'_{12}(0) = -3.95$,
  $F'_{12}(1) = 0.7 - 3.95 = -3.25$
  The derivative is negative and does not change sign; the function is strictly decreasing.

2. $f_1(x)$ **and** $f_3(x)$

Consider $F_{13}(x) = f_1(x) - f_3(x) = 0.35x^2 - 0.95x + 2.7 - \frac{1}{x+2}$.

- $F_{13}(x)$ is continuous for $x > -2$.
- Check on $[-1.96, 0]$:
  $F_{13}(-1.96) \approx 0.35 \cdot (-1.96)^2 - 0.95 \cdot (-1.96) + 2.7 - \frac{1}{-1.96+2} \approx 0.35 \cdot 3.8416 + 1.862 + 2.7 - \frac{1}{0.04} \approx 1.3446 + 1.862 + 2.7 - 25 \approx 5.9066 - 25 \approx -19.0934$
  $F_{13}(0) = 0.35 \cdot 0^2 - 0.95 \cdot 0 + 2.7 - \frac{1}{2} = 2.7 - 0.5 = 2.2$
- $F_{13}(-1.96) \cdot F_{13}(0) < 0$.
- Derivative $F'_{13}(x) = 0.7x - 0.95 + \frac{1}{(x+2)^2}$ on $[-1.96, 0]$: the denominator is positive, x ranges from -1.96 to 0, therefore $F'_{13}(x)$ does not vanish and does not change sign.

3. $f_2(x)$ **and** $f_3(x)$

$F_{23}(x) = f_2(x) - f_3(x) = 3x + 1 - \frac{1}{x+2}$.

- $F_{23}(x)$ is continuous for $x > -2$.
- Check on $[-1, 0]$: $F_{23}(-1) = 3 \cdot (-1) + 1 - \frac{1}{-1+2} = -3 + 1 - 1 = -3$
  $F_{23}(0) = 0 + 1 - \frac{1}{2} = 1 - 0.5 = 0.5$
- $F_{23}(-1) \cdot F_{23}(0) = -3 \cdot 0.5 = -1.5 < 0$.
- Derivative: $F'_{23}(x) = 3 + \frac{1}{(x+2)^2}$ is always positive on $[-1, 0]$, hence strictly increasing.

Thus, for all chosen intervals the necessary conditions are satisfied: the function is continuous, values at the endpoints have opposite signs, and the derivative does not vanish or change sign. This guarantees existence and uniqueness of the root on each interval.

2..2 Simpson's Method

Let us approximate the integral $I = \int_a^b f(x)dx$ using Simpson's method. According to [1], we require $f \in C^4[a, b]$. To compute the integral, split $[a, b]$ into an even number n of equal subintervals of length

$$h = \frac{b-a}{n}, \quad x_i = a + i\,h,\ i = 0, 1, \ldots, n$$

On every other subinterval $[x_{i-1}, x_{i+1}]$ we approximate $f(x)$ by a parabola passing through (x_{i-1}, f_{i-1}), (x_i, f_i), (x_{i+1}, f_{i+1}) and obtain Simpson's formula:

$$\int_a^b f(x)\,dx = \frac{h}{3}\Big[f_0 + 4\sum_{\substack{i=1 \\ i \text{ odd}}}^{n-1} f_i + 2\sum_{\substack{i=2 \\ i \text{ even}}}^{n-2} f_i + f_n\Big] + R \tag{4}$$

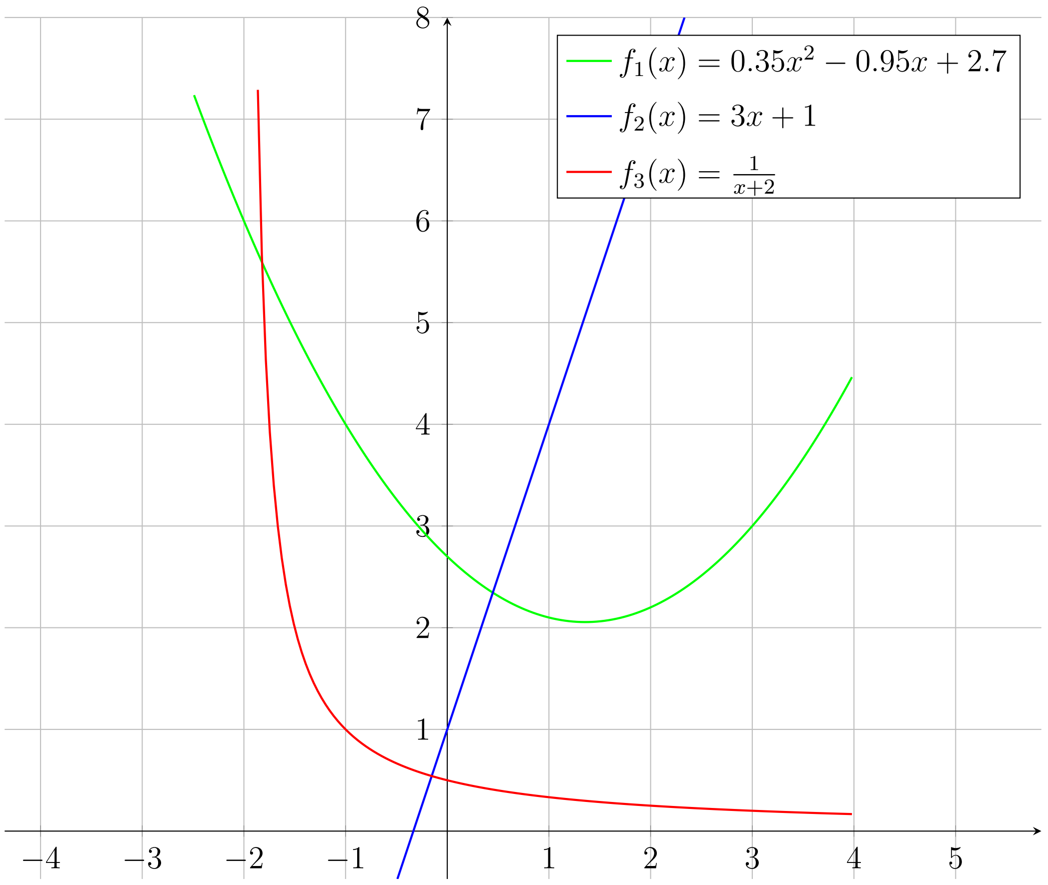


Рис. 1: Plane figure bounded by the graphs of the given equations

where R is the remainder term:

$$R = -\frac{(b-a)}{180}\, h^4\, f^{(4)}(\xi), \qquad \xi \in [a,b] \tag{5}$$

or equivalently:

$$R = -\frac{(b-a)^5}{2880\, n^4}\, f^{(4)}(\xi) \tag{6}$$

2..3 Choice of ε_1 and ε_2

In the program, the required final accuracy is 10^{-4}. To ensure that the total errors from root finding and numerical integration stay within this bound, we introduce stricter internal tolerances:

$$\varepsilon_1 = \varepsilon_2 = 10^{-6}.$$

2..4 Justification for the internal tolerances ε_1 and ε_2

To ensure overall accuracy not worse than 10^{-4}, we must account for error accumulation at each stage. Therefore, stricter internal tolerances ε_1 and ε_2 are chosen to guarantee the final accuracy requirement.

2..4.1 Justification for ε_1

Let x^* be the exact solution of $F(x) = 0$, and x_{k+1} be the approximation after the (k+1)-th iteration. Then, by properties of the method, the root error is related to the interval length:

$$|x_{k+1} - x^*| \le \frac{b_{k+1} - a_{k+1}}{2}.$$

To ensure error no more than ε_x, we need:

$$\frac{b_{k+1} - a_{k+1}}{2} < \varepsilon_x,$$

where ε_x is the allowable error in the root. Given the final accuracy 10^{-4} and the internal tolerance $\varepsilon_1 = 10^{-6}$, we have:

$$|x_{k+1} - x^*| < \frac{\varepsilon_1}{2} = 5 \times 10^{-7}.$$

This is much smaller than the required final accuracy, providing a buffer for floating-point arithmetic and iteration error accumulation.

2..4.2 Justification for ε_2

For numerical integration by Simpson's method, the remainder term is:

$$R = -\frac{(b-a)^5}{2880\, n^4} f^{(4)}(\xi),$$

where $M = \max_{x \in [a,b]} |f^{(4)}(x)|$. To guarantee the integration error does not exceed ε_2, we need:

$$|R| \le \frac{(b-a)^5 M}{2880\, n^4} < \varepsilon_2.$$

Hence the minimum number of subintervals:

$$n \ge \left(\frac{(b-a)^5 M}{2880\, \varepsilon_2} \right)^{1/4}.$$

When M is unknown, the Runge rule is used:

$$|I_n - I_{2n}| \approx \frac{1}{15} |I_{2n} - I_{4n}| < \varepsilon_2,$$

which provides error control and ensures the final error does not exceed 10^{-4}, accounting for possible accumulation.

3. Experimental Results

Using the implemented program, we compute the coordinates of the intersection points of the curves and compile a table, also showing them on a plot (Fig. 2)

Curves	x	y
1 and 2	0.4482	2.3445
2 and 3	-0.1529	0.5414
1 and 3	-1.8211	5.5909

Таблица 1: Coordinates of the intersection points

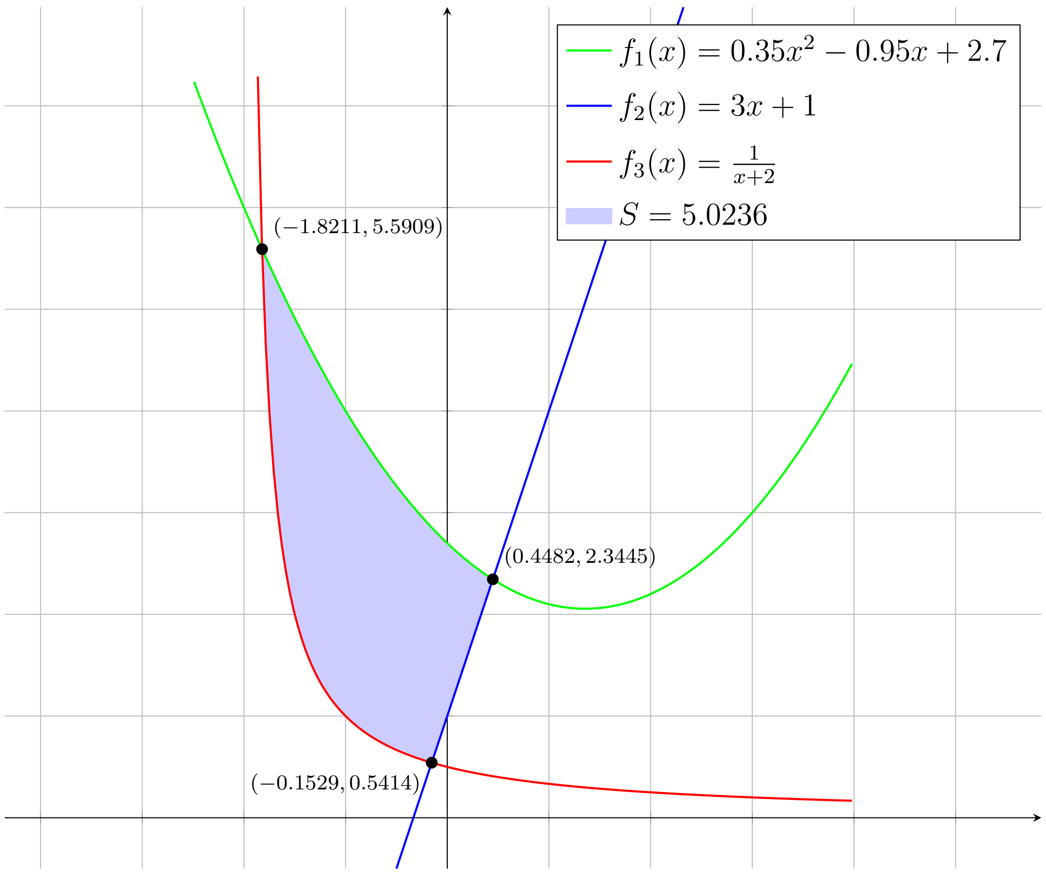


Рис. 2: Plane figure bounded by the graphs of the given equations

4. Program Structure and Function Specification

This section describes the program architecture, its components, and their interaction. The program is divided into several modules, each responsible for a specific aspect of functionality.

4..1 Overall architecture

The program consists of the following modules:

- Header file with declarations (`functions.h`)
- Main build module (`main.c`)
- Module with implemented numerical methods (`compute.c`)
- Test build module (`test_mode.c`)
- Assembly module implementing evaluation of the assignment functions and their derivatives at a point (`func.asm`)

4..2 Detailed description of project files

4..2.1 compute.c

Purpose: Implementation of numerical methods and testing.

- `test_root()`: for testing the `root` function
  - Automatically determines the offset to access derivatives via `funcs[0](0)`
  - Output format: absolute and relative error
- `test_integral()`: for testing the `integral` function
  - Compares the result with a reference up to 1e-4
- `root()` implements root finding using the combined method
- `integral()` implements adaptive Simpson for computing the integral. Function specifics:
  - Initial partition: $n = 2$
  - Runge factor: $p = 1/15$
  - Maximum partition: 5×10^6 (to avoid an infinite loop)

4..2.2 functions.h

- global variables:
  - array `funcs`:
    - In main mode: 7 elements $[idx, f1, f2, f3, df1, df2, df3]$
    - In test mode: 9 elements $[idx, test\_f1 - 4, test\_df1 - 4]$
  - `EPSILON`
  - `root_iteration` — counting the number of iterations in `root`
- function prototypes

4..2.3 test_mode.c

- test functions
- `main()` — prints testing results

4..2.4 main.c

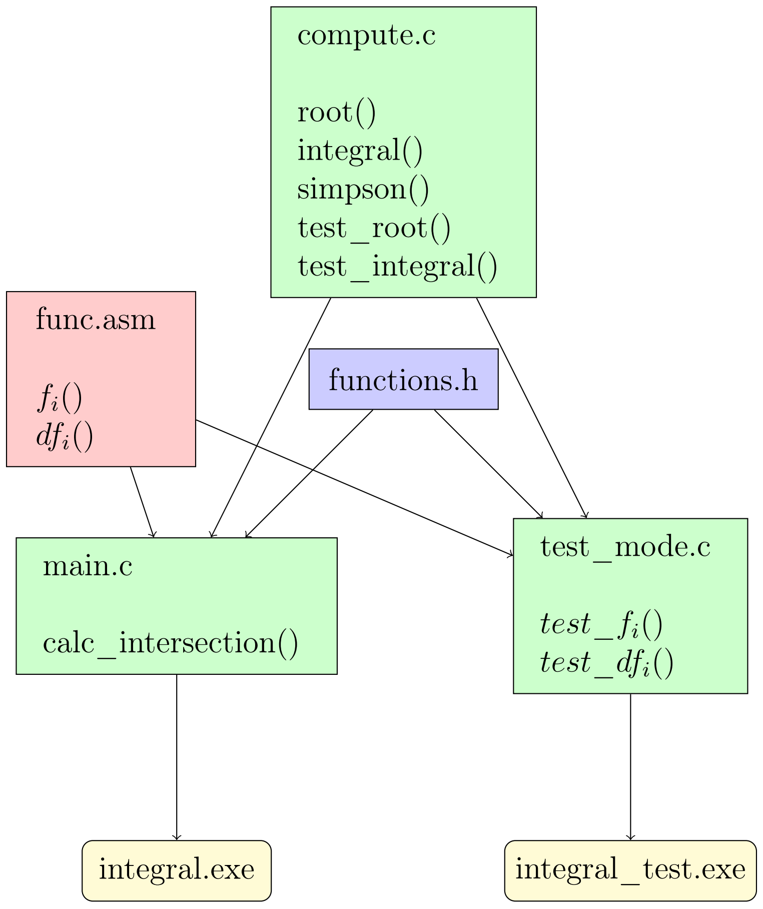
- `calc_intersection()` — computes intersections in the main mode
- `main()` — main function with command-line support

4..2.5 func.asm

- functions to compute values of $f1$, $f2$, $f3$ at a point
- functions to compute values of the derivatives $df1$, $df2$, $df3$ at a point

4..2.6 Graphical representation of the structure

The structure of the program and interaction of its components can be represented by the following diagram:


compute.c
root()
integral()
simpson()
test_root()
test_integral()
func.asm
$f_i()$
$df_i()$
functions.h
main.c
calc_intersection()
test_mode.c
$test\_f_i()$
$test\_df_i()$
integral.exe
integral_test.exe

5. Program Build (Makefile)

The project Makefile controls the compilation process, defining compilers, flags, and module dependencies. Below is a description of its key components consistent with your Makefile structure.

5..1 Compilers and flags

At the beginning of the Makefile, the compilers and their parameters are set:

- `CC = gcc -m32 -no-pie -fno-pie` — C compiler with options:
  - `-m32` — build 32-bit code for compatibility with the assembly module;
  - `-no-pie`, `-fno-pie` — disable PIE, important for correct linking with assembly.
- `NASM = nasm` — NASM assembler for building assembly files.

5..1.1 C compilation flags

Several groups of flags are used:

- `CFLAGS_COMMON` — enables a wide range of warnings (`-Wall`, `-Werror`, `-Wformat-security`, etc.) to improve code quality and safety.
- `CFLAGS_SANITIZE` — enables sanitizers for detecting undefined behavior (`-fsanitize=undefine` `-fsanitize-undefined-trap-on-error`).
- `CFLAGS` — combines common flags and sanitizers.
- `CFLAGS_TEST` — similar to `CFLAGS_COMMON`, but adds `TEST_MODE` definition for compiling tests.
- `-g` — add debug information.
- `-std=gnu99` — C language standard.
- `-O2` — level 2 optimization.

5..1.2 NASM flags

- `NASMFLAGS = -fwin32` — build assembly code as 32-bit Windows object files.

5..2 Build targets

The Makefile defines the following main targets:

- `all` — build the main program and the test program, then remove object files.
- `main` — build the main program (`integral`).
- `test` — build and run the test program (`integral_test`).
- `clean` — remove all object files (`*.o`).

5..3 Build rules

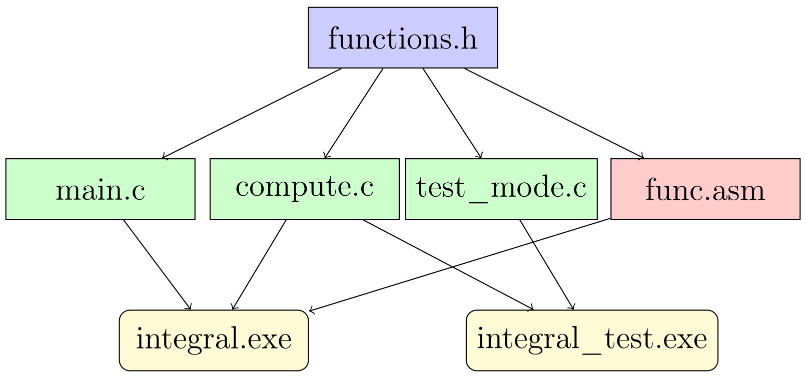
- `integral`: built from `main.o`, `compute.o`, `func.o` using `CC` and `CFLAGS`, linked with the math library (`-lm`).
- `integral_test`: built from `test_mode.o`, `compute.o` using `CFLAGS_TEST`.
- `func.o`: built from `func.asm` using NASM.

5..4 Features

- Use of strict compilation flags and sanitizers improves reliability and safety.
- Automated test run via the `test` target.
- Cleanup of intermediate files after building both main and test programs.

5..5 Module dependencies

The main dependencies between modules can be represented by the following diagram:



Where:

- `functions.h` — header file with function declarations:
  - Declarations of main functions: `root`, `integral`
  - Declarations of `f1`, `f2`, `f3` and their derivatives
  - Definition of `func` as a function pointer type
  - Constant `EPSILON` for computation accuracy
- `main.c` — main program module:
  - `main` — entry point, command-line argument handling
  - `calc_intersection` — computing curve intersections
  - `test_root` — root-finding tests
  - `test_integral` — integral computation tests

- `compute.c` — numerical methods:
  - `root` — combined method for root finding
  - `integral` — definite integral computation
  - `simpson` — Simpson's method for numerical integration
  - Global variable `root_iterations` for counting iterations
- `test_mode.c` — testing module:
  - Test functions: `test_f1`, `test_f2`, `test_f3`, `test_f4`
  - Their derivatives: `test_df1`, `test_df2`, `test_df3`, `test_df4`
- `func.asm` — assembly module:
  - Main functions: `f1`, `f2`, `f3`
  - First derivatives: `df1`, `df2`, `df3`
- `integral.exe` — main executable
- `integral_test.exe` — test executable

5..6 Build specifics

When building the test mode, a special `-DTEST_MODE` flag is added to include test functions in the program. This allows maintaining a single code base for both the main program and tests.

6. Debugging and Testing the Program

To verify program correctness, we consider three test functions for which roots and definite integrals on given intervals are computed analytically:

$$f_1(x) = x^2 + 5x + 6 \tag{7}$$
$$f_2(x) = \sin x \tag{8}$$
$$f_3(x) = x^3 - 1 \tag{9}$$

For each function, we find the intersections with the Ox axis (i.e., zeros of the function), and analytically compute the definite integral on the corresponding interval.

6..1 Function $f_1(x) = x^2 + 5x + 6$

Solve:

$$x^2 + 5x + 6 = (x + 2)(x + 3) = 0 \Rightarrow x = -2, -3$$

Compute the definite integral on $[-4, -2]$:

$$\int_{-4}^{-2} (x^2 + 5x + 6)\, dx = \left(\frac{x^3}{3} + \frac{5x^2}{2} + 6x \right) \Big|_{-4}^{-2}$$

Substitute:

$$= \left(-\frac{8}{3} + 10 - 12 \right) - \left(-\frac{64}{3} + 40 - 24 \right) = \frac{2}{3} \approx 0.6667$$

6..2 Function $f_2(x) = \sin x$

Zeros are given by:

$$\sin x = 0 \Rightarrow x = \pi n, \quad n \in \mathbb{Z}$$

Compute the integral on $[1, 2]$:

$$\int_{0}^{\pi} \sin x\, dx = -\cos x \Big|_{0}^{\pi} = -(-1) - (-1) = 2$$

6..3 Function $f_3(x) = x^3 - 1$

Zero:

$$x^3 - 1 = 0 \Rightarrow x = 1$$

Compute the integral on $[0, 2]$:

$$\int_{0}^{2} (x^3 - 1)\, dx = \left(\frac{x^4}{4} - x \right) \Big|_{0}^{2} = \left(\frac{16}{4} - 2 \right) - \left(\frac{0}{4} - 0 \right) = (4 - 2) - 0 = 2$$

6..4 Choice of intervals for root search

For each test function we provide the derivative, determine its sign on the corresponding interval, and indicate the interval used for root search.

- $f_1(x) = x^2 + 5x + 6$
  Derivative: $f_1'(x) = 2x + 5$
  On $[-4, -3]$:
  $f_1'(-4) = 2 \cdot (-4) + 5 = -3$
  $f_1'(-3) = 2 \cdot (-3) + 5 = -1$
  Derivative is negative. On $[-2.4, -1.5]$:
  $f_1'(-2.4) = 2 \cdot (-2.4) + 5 = 0.2$
  $f_1'(-1.5) = 2 \cdot (-1.5) + 5 = 2.0$
  Derivative is positive. On each subinterval the sign is constant.

- $f_2(x) = \sin x$
  Derivative: $f_2'(x) = \cos x$
  On $[3, 4]$:
  $f_2'(3) = \cos 3 \approx -0.990$
  $f_2'(4) = \cos 4 \approx -0.654$
  Derivative is negative on the entire interval.

- $f_3(x) = x^3 - 1$
  Derivative: $f_3'(x) = 3x^2$
  On $[0.5, 1.5]$:
  $f_3'(0.5) = 3 \cdot (0.5)^2 = 0.75$
  $f_3'(1.5) = 3 \cdot (1.5)^2 = 6.75$
  Derivative is strictly positive.

Intervals used for root search:

- For $f_1(x)$: $[-4, -3]$ (root $x = -3$), $[-2.5, -1.5]$ (root $x = -2$)
- For $f_2(x)$: $[3, 4]$ (root $x = \pi \approx 3.1416$)
- For $f_3(x)$: $[0.5, 1.5]$ (root $x = 1$)

Thus, for each test the derivative does not change sign on the chosen interval, which matches the conditions for applying the combined method.

6..5 Conclusions

We compare the program results with the theoretical ones and present them in a table.

Таблица 2: Results of theoretical calculations for testing

Function	**Roots (theor.)**	**Roots (prog.)**	**Integral (theor.)**	**Integral (prog.)**
$x^2 + 5x + 6$	$-3, -2$	$-3, -2$	$\frac{2}{3}$	0.6667
$\sin x$	$\pi n, n \in \mathbb{Z}$	3.1416	2	2
$x^3 - 1$	1	1	2	2

The functions `root` and `integral` work correctly and achieve the required accuracy.

7. Analysis of Mistakes Made

- missing `finit` instruction at the start of the assembly functions that compute values and derivatives of the given functions
- refinement of the `integral` function to reuse previously computed values
- missing function prototypes
- using second derivatives in the `root` function

Список литературы

[1] Ilyin V. A., Sadovnichy V. A., Sendov Bl. Kh. Mathematical Analysis. Vol. 1 — Moscow: Nauka, 1985.