Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчет по заданию №6

«Сборка многомодульных программ. Вычисление корней уравнений и определенных интегралов.»

Вариант 3 / 3 / 1

Выполнил:
студент 103 группы
Кудисов А. А.

Преподаватель:
Кузьменкова Е. В.
Дудина. И. А

Москва
2020

Содержание

Постановка задачи

В данном задании требовалось реализовать программу, считающую с заданной точностью ε площадь фигуры, ограниченной 3 графиками функций.

Для нахождения точек пересечения графиков функций (вершин многоугольника) использовался метод касательных. Для нахождения определенных интегралов применялся метод прямоугольников.

Для метода касательных необходимо постояннство знака как 1-ой производной, так и 2-ой (кроме того, они не должны обращаться в нуль), а также значения функции на концах отрезка должны иметь разные знаки. Отрезок, на котором происходит поиск корня, определяется аналитически.

Программа реализованна на языке C. Функции, считающие значения функций и их первых производных в заданной точке, написаны на языке ассемблера.

Математическое обоснование

Поиск точек пересечения графиков функций:

Рассмотрим все функции вида $f_{i,j} = f_i$ - f_j, где i, j = 1,2,3, $i < j$, а также их 1-ые и 2-ые производные. Для этого построим следующую таблицу:

i	$f_i(x)$	$f_i^{'}(x)$	$f_i^{''}(x)$
1	$e^{-x}+3$	$-e^{-x}$	e^{-x}
2	$2x-2$	2	0
3	$\frac{1}{x}$	$-\frac{1}{x^2}$	$\frac{2}{x^3}$

Таблица 1: Функции и их производные

Для применения метода касательных требуется постояннство знака как 1-ой производной, так и 2-ой на заданном отрезке (кроме того, они не должны обращаться в нуль), а также значения функции на концах отрезка должны иметь разные знаки [1].

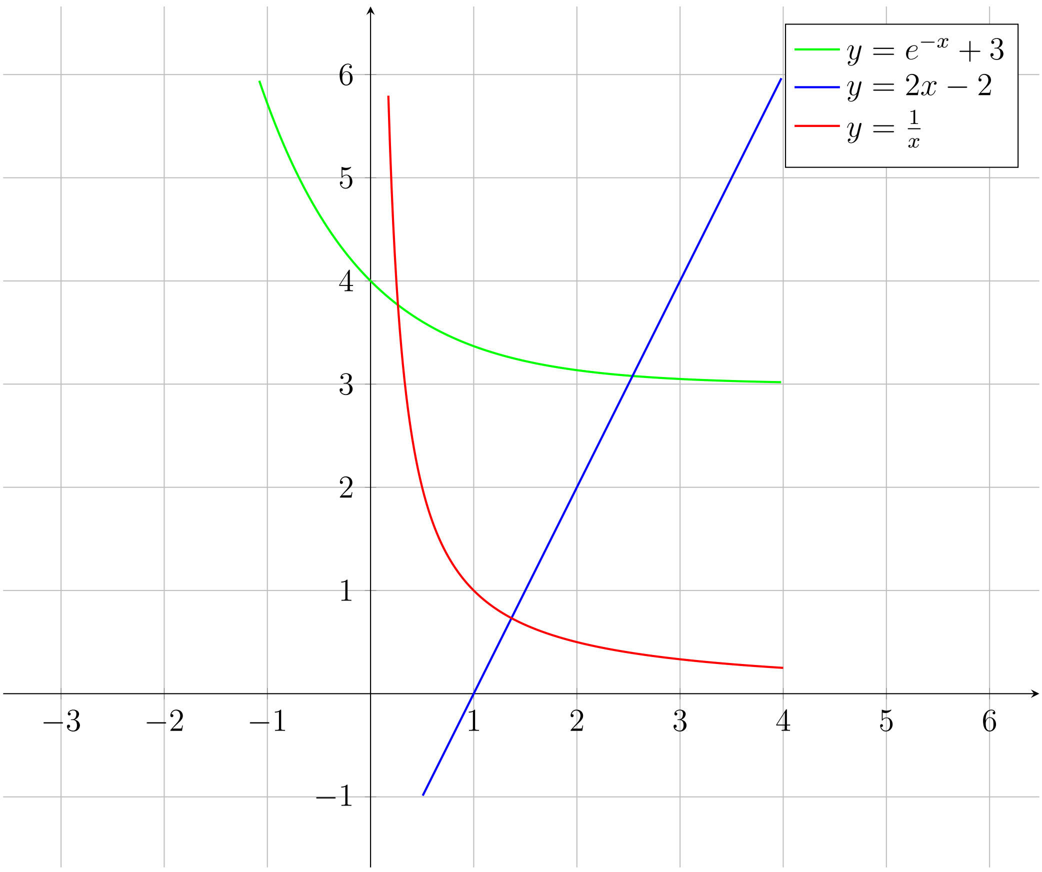


Рис. 1: Плоская фигура, ограниченная графиками заданных уравнений

- Корень функции $f_{1,2}(x) = e^{-x} - 2x + 5$ лежит на отрезке $[0; 4]$, так как:

    1. $f_{1,2}(0) = e^0 - 2 * 0 + 5 = 1 + 5 = 6 > 0$
       $f_{1,2}(4) = e^{-4} - 2 * 4 + 5 < 1 - 8 + 5 = -2 < 0$

2. Первая производная $f'_{1,2}(x) = -e^{-x} - 2 < 0 \ \forall x \in [0;4]$
3. Вторая производная $f''_{1,2}(x) = e^{-x} > 0 \ \forall x \in [0;4]$

- Корень функции $f_{1,3}(x) = e^{-x} - \frac{1}{x} + 3$ лежит на отрезке $[0.1;1]$, так как:
  1. $f_{1,3}(0.1) = e^{-0.1} - \frac{1}{0.1} + 3 < 1 - 10 + 3 = -6 < 0$
     $f_{1,3}(1) = e^{-1} - \frac{1}{1} + 3 > -1 + 3 = 2 > 0$
  2. Первая производная $f'_{1,3}(x) = -e^{-x} + \frac{1}{x^2} > 0 \ \forall x \in [0.1;1]$
  3. Вторая производная $f''_{1,3}(x) = e^{-x} - \frac{2}{x^3} < 0 \ \forall x \in [0.1;1]$

- Корень функции $f_{2,3}(x) = 2x - \frac{1}{x} - 2$ лежит на отрезке $[1;2]$, так как:
  1. $f_{2,3}(1) = 2*1 - \frac{1}{1} - 2 = 2 - 1 - 2 = -1 < 0$
     $f_{2,3}(2) = 2*2 - \frac{1}{2} - 2 = 4 - 0.5 - 2 = 1.5 > 0$
  2. Первая производная $f'_{2,3}(x) = 2 + \frac{1}{x^2} > 0 \ \forall x \in [1;2]$
  3. Вторая производная $f''_{2,3}(x) = -\frac{2}{x^3} < 0 \ \forall x \in [1;2]$

Метод прямоугольников:

Условием сходимости метода прямоугольников является непрерывность второй производной функции. Погрешность данного метода равна $R = f''(\varphi)\frac{b-a}{24}h^2$, где $a < \varphi < b$, $h = \frac{b-a}{n}$ — шаг разбиения отрезка $[a;b]$, n — количество сегментов разбиения [1].

Обоснование выбора ε_1 и ε_2:

Итоговая погрешность ε должна быть не более 10^{-3}. Докажем, что $\varepsilon_1 = \varepsilon_2 = 10^{-5}$ дадут итоговую погрешность, удовлетворяющую всем требованиям, где ε_1 - погрешность при вычислении абсцисс точек пересечения, ε_2 - погрешность при подсчете определенного интеграла.

Прежде всего, заметим, что функции f_1, f_2, f_3 монотонны и все абсциссы точек пересечений лежат на отрезке $[0.1;4]$. Обозначим за f_{max} максимальное из абсолютных значений функций f_1, f_2, f_3 на отрезке $[0.1;4]$. Нетрудно убедиться, что $f_{max} = |\frac{1}{0.1}| = 10$.

При поиске абсцисс точек пересечения погрешность равна ε_1, таким образом при вычислении определенного интеграла мы получаем потерю точности не более $\varepsilon_2 + 2 * \varepsilon_1 * f_{max}$ (погрешность интегрирования + "лишняя" площадь прямоугольников с двух сторон). Следовательно, итоговая погрешность $\varepsilon \le 3(\varepsilon_2 + 2 * \varepsilon_1 * f_{max})$, так как площадь искомой фигуры считается как алгебраическая сумма 3-ех определенных интегралов.

$\varepsilon \le 3(0.00001 + 2 * 0.00001 * 10) = 3 * 0.00021 = 0.00063 < 0.001 \rightarrow \varepsilon_1 = 10^{-5}$ и $\varepsilon_2 = 10^{-5}$ подходят.

Результаты экспериментов

В результате проведенных вычислений были получены следующие значения: координаты точек пересечения (табл. 2, рис. 2) и площадь полученной фигуры (рис. 2).

Кривые	x	y
1 и 2	2.5394	3.0788
2 и 3	1.3660	0.7320
1 и 3	0.2655	3.7665

Таблица 2: Координаты точек пересечения

Площадь плоской фигуры, заключенной между кривыми f_1, f_2 и f_3, равна 3.635786 (рис. 2).

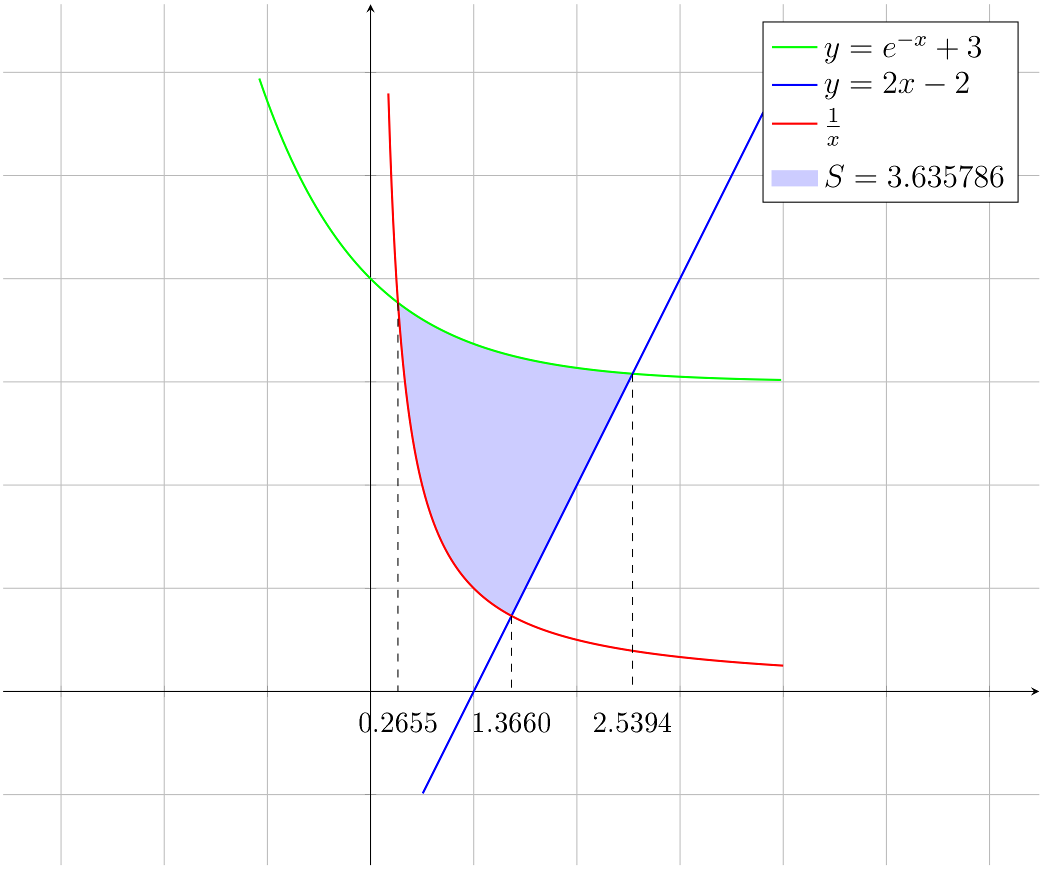


Рис. 2: Плоская фигура, ограниченная графиками заданных уравнений

Структура программы и спецификация функций

Программа состоит из 7 модулей, написанных на 2-ух языках: C и язык ассемблера:

1. Язык ассемблера:
   - funct.asm
2. C:
   - main.c
   - root.c
   - integral.c
   - funct_test.c
3. Библиотеки:
   - funct.h
   - math_module.h

funct.asm

Данный модуль содержит в себе реализацию функций f_1, f_2, f_3 и их первых производных. Каждая из функций имеет возвращаемый тип double (8 бит), в качестве входного параметра функции получают одно вещественное число типа double.

funct_test.c

Данный модуль содержит в себе реализацию 3 тестовых функций и их первых производных, также есть "нулевая" функция, всегда возвращающая 0 ($y = 0$). Каждая из функций имеет возвращаемый тип double (8 бит), в качестве входного параметра функции получают одно вещественное число типа double.

integral.c

Данный модуль содержит в себе реализацию всего одной функций:

- $double\ integral(double\ (^*f)(double),\ double\ a,\ double\ b,\ double\ eps2)$

  Эта функция считает определенный интеграл заданной функции f на промежутке от a до b с заданной точностью $eps2$ при помощи метода прямоугольников.

root.c

Данный модуль содержит в себе реализацию 2-ух функций:

- $double\ root(double\ (^*f)(double),\ double\ (^*g)(double),\ double\ (^*f\_der)(double),\ double\ (^*g\_der)(double),\ double\ a,\ double\ b,\ double\ eps1)$

  Эта функция находит абсциссу точки пересечения функций f, g на промежутке от a до b с заданной точностью $eps1$ при помощи метода касательных.

- *double val(double (*f)(double), double (*g)(double), double x)*

  Это вспомогательная функция, которая находит разность функций f, g в точке x.

funct.h
Данная библиотека содержит в себе объявления всех тестовых функций, их первых производных и функций f_1, f_2, f_3 с их первыми производными.

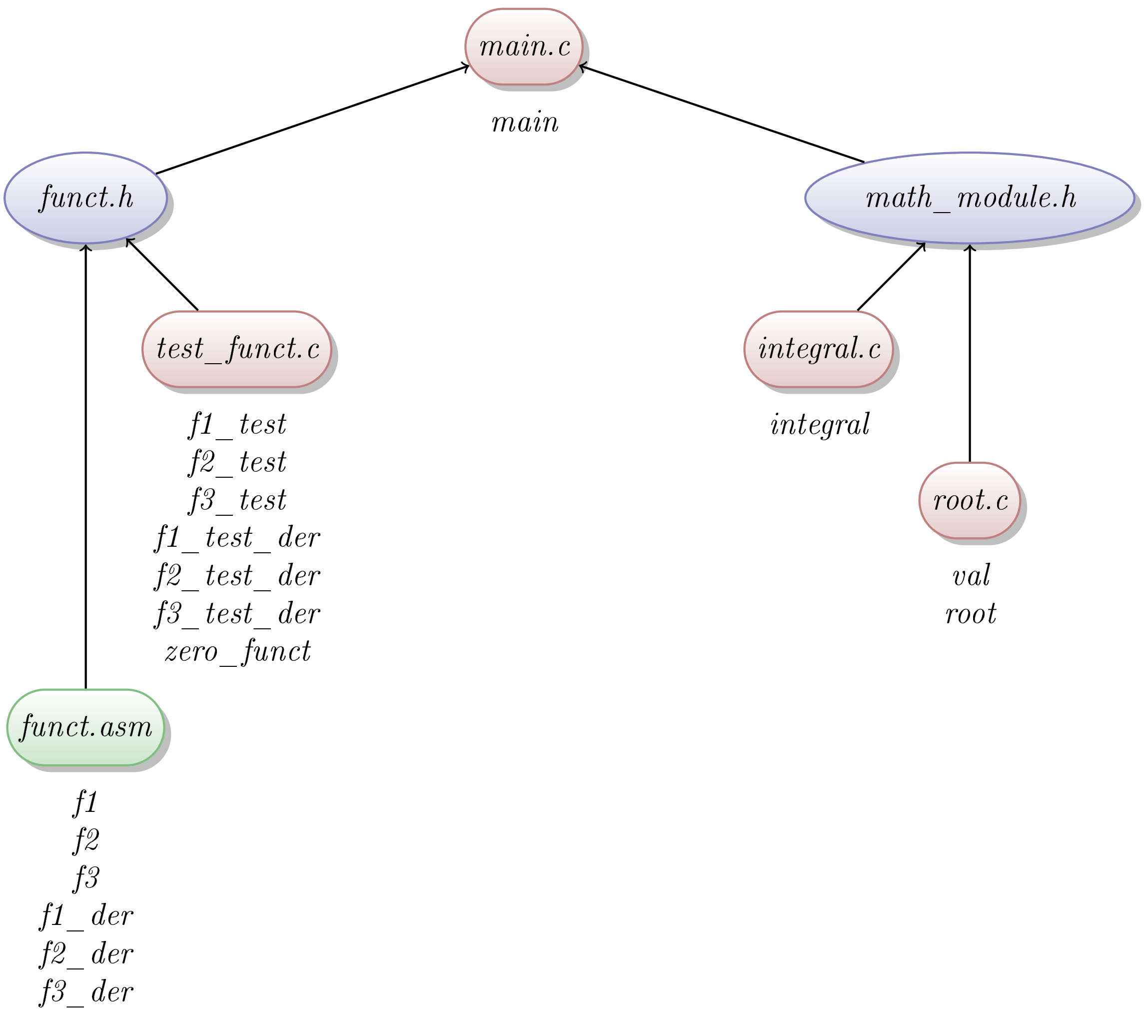
math_module.h
Данная библиотека содержит в себе объявления функций *root* и *integral*.

main.c
Главный модуль программы, содержащий в себе реализацию лишь одной функции.

- *int main(int args, char *argv[])*

  Функция main получает на вход данные из командной строки. Дальнейшее её поведение зависит от полученных ключей:

  1. Нет ключей: программа выведет значение площади плоской фигуры, ограниченной кривыми f_1, f_2, f_3, и завершит работу.
  2. -iter: помимо значения посчитанной площади программа выведет количество итераций, необходимых для нахождения абсцисс точек пересечения графиков функций.
  3. -absc: помимо значения посчитанной площади программа выведет абсциссы точек пересечения графиков функций.
  4. -test [параметры]: данная команда позволяет протестировать как функцию *root*, так и *integral*. Пользователь сам определяет, какую функцию он тестирует, на каком отрезке и с какой точностью.
  5. -help: программа выводит список доступных ключей, их описание, пример использования ключа -test, список тестовых функций, после чего завершает работу.
  6. Некорректный ключ или неверный формат: программа выдаст сообщение об ошибке и завершит работу.


main.c
main
funct.h
math_module.h
test_funct.c
f1_test
f2_test
f3_test
f1_test_der
f2_test_der
f3_test_der
zero_funct
integral.c
integral
root.c
val
root
funct.asm
f1
f2
f3
f1_der
f2_der
f3_der

Сборка программы (Make-файл)

Make-файл:

```
NASM = nasm -g -f elf32
GCC = gcc -m32 -Wall -g
ALL_NAMES = main.o root.o integral.o funct.o funct_test.o

.PHONY: all clean

all: square

square: $(ALL_NAMES)
	$(GCC) $^ -o square -lm

main.o: main.c funct.h math_module.h
	$(GCC) -c $<

root.o: root.c
	$(GCC) -c $^

integral.o: integral.c
	$(GCC) -c $^

funct.o: funct.asm
	$(NASM) $^

funct_test.o: funct_test.c
	$(GCC) -c $^

clean:
	rm -rf *.o
```

Сборка программы осуществляется при помощи Makefile. Ключ "all" позволяет полностью собрать весь проект. Ключ "clean" удаляет все объектные файлы, образовавшиеся в результате сборки программы. Метка ".PHONY" необходима для избежания конфликта имен, если в проекте будут присутствовать файлы с именем *all* или *clean*. Итоговый файл - *square* - зависит от всех объектных файлов, образующихся в результате сборки. Объектные файлы зависят от соответствующих с-/асм-файлов и библиотек.

Отладка программы, тестирование функций

Отладка программы производилась при помощи 3-ех вспомогательных функций, описанный в файле $funct\_test.c$. Пользователь вправе самостоятельно выбрать функцию для тестирования, отрезок, на котором будет искаться корень или же считаться определенный интеграл, и точность, с которой будут работать численные методы. В таблице 3 представлены значения производных и первообразных тестовых функций.

i	$f_i(x)$	$f_i'(x)$	$f_i''(x)$	$F_i(x)$
1	x^2-4	$2x$	2	$\frac{x^3}{3}-4x+C$
2	$\sqrt{x}-1$	$\frac{1}{2\sqrt{x}}$	$-\frac{1}{4\sqrt{x^3}}$	$\frac{2}{3}\sqrt{x^3}-x+C$
3	$e^{x+1}-1$	e^{x+1}	e^{x+1}	$e^{x+1}-x+C$

Таблица 3: Функции, их производные и первообразные

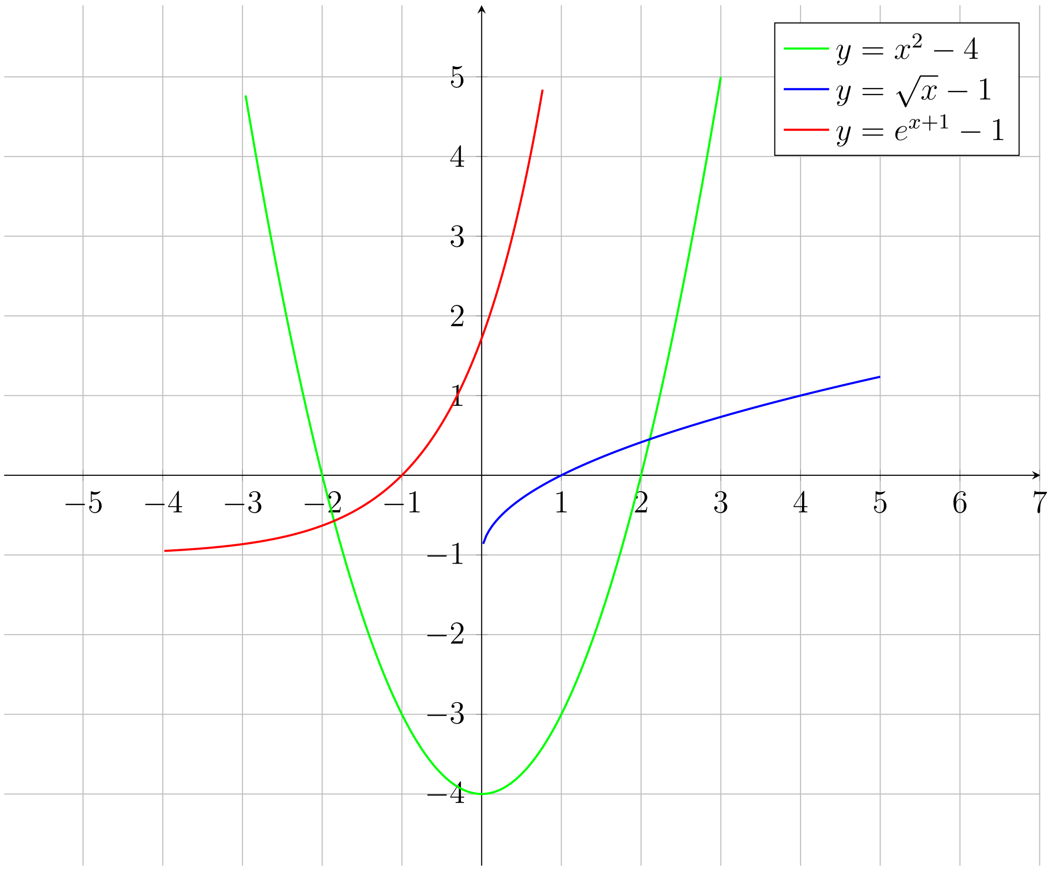


Рис. 3: Графики тестируемых функций

В таблице 4 представлены результаты работы функции root для каждой тестовой функции на заданном отрезке с точностью $\varepsilon = 0.001$. В справедливости полученных значений нетрудно убедиться непосредственной подстановкой. Однако, для каждой тестовой функции легко найти корни и аналитическим методом:

1. $x^2 - 4 = (x-2)(x+2) = 0 \to x = \pm 2$

2. $\sqrt{x} - 1 = 0 \to \sqrt{x} = 1 \to x = 1$

3. $e^{x+1} - 1 = 0 \to e^{x+1} = 1 \to x + 1 = 0 \to x = -1$

i	$f_i(x)$	отрезок	корень	точность
1	$x^2 - 4$	$[1;3]$	2.000	0.001
2	$\sqrt{x} - 1$	$[0.1;2]$	0.999	0.001
3	$e^{x+1} - 1$	$[-2;0]$	-0.999	0.001

Таблица 4: Корни функций на заданных отрезках

В таблице 5 представлены результаты работы функции integral для каждой тестовой функции на заданном отрезке с точностью $\varepsilon = 0.001$. В справедливости полученных значений нетрудно убедиться, воспользовавшись формулой Ньютона-Лейбница:

1. $(\frac{x^3}{3} - 4x)|_1^3 = \frac{3^3}{3} - 4*3 - \frac{1^3}{3} + 4*1 = 9 - 12 - \frac{1}{3} + 4 = 0.(6)$

2. $(\frac{2}{3}\sqrt{x^3} - x)|_1^4 = \frac{2}{3}\sqrt{4^3} - 4 - \frac{2}{3}\sqrt{1^3} + 1 = \frac{2}{3}*8 - 3 - \frac{2}{3} = \frac{16-9-2}{3} = \frac{5}{3} = 1.(6)$

3. $(e^{x+1} - x)|_{-1}^1 = e^2 - 1 - e^0 - 1 = e^2 - 3 \approx 4.389$

i	$f_i(x)$	отрезок	определенный интеграл	точность
1	$x^2 - 4$	$[1;3]$	0.666	0.001
2	$\sqrt{x} - 1$	$[1;4]$	1.667	0.001
3	$e^{x+1} - 1$	$[-1;1]$	4.389	0.001

Таблица 5: Определенные интегралы функций на заданных отрезках

Программа на Си и на Ассемблере

Исходный код программы лежит в архиве и приложен к данному отчету.

Анализ допущенных ошибок

В ходе написания программы возникло несколько ошибок в реализации функций *integral* и *root*, которые впоследствии были исправлены. С созданием архитектуры программы сложностей не возникло.

Список литературы

[1] Ильин В. А., Садовничий В. А., Сендов Бл. Х. Математический анализ. Ч. 1 — Москва: Издательство Проспект, 2004.