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 № 9/2/2**

Выполнил:

Студент 101 группы

Казаринов А. В.

Преподаватели:

Дудина И. А.

Кузьменкова Е. А.

Москва

2020

**Содержание**

**Постановка задачи 3**

**Математическое обоснование 4-5**

**Результаты экспериментов 6**

**Структура программы и спецификация функций 7-8**

**Сборка программы (Make-файл) 9-10**

**Отладка программы, тестирование функций 11**

**Программа на Си и на Ассемблере 12**

**Анализ допущенных ошибок 13**

**Список цитируемой литературы 14**

**Постановка задачи**

Была поставлена задача реализовать метод вычисления площади фигуры, ограниченной тремя заданными кривыми, с точностью . Уравнения кривых:

Задача состоит из двух шагов.

Первый шаг: найти вершины фигуры, являющиеся точками пересечения кривых, *методом хорд (секущих)*. Точки находятся решением приближённо уравнений , где , c точностью .

Второй шаг: представить площадь фигуры в виде алгебраической суммы определённых интегралов. Вычислить данные интегралы нужно *по формуле трапеций* с точностью . В качестве пределов интегрирования берутся абсциссы точек пересечения кривых, вычисленные на первом шаге.

Для решения задачи требуется создать функции root и integral для нахождения корней уравнения и вычисления определенных интегралов соответственно. Данные функции необходимо протестировать. Также отрезок, на котором функция root будет искать корень, должен быть вычислен аналитически.

Сборка программы должна осуществляться при помощи утилиты make.

**Математическое обоснование**

**I. Обоснование выбора отрезков для поиска точек пересечения кривых[1]**

1) Пересечение кривых и ищется на отрезке [1, 2], – , ) < 0 непрерывна на отрезке [1, 2] как разность непрерывных функций.

2) Пересечение кривых и ищется на отрезке [0, 1], – , ) < 0 непрерывна на отрезке [0, 1] как разность непрерывных функций.

3) Пересечение кривых и ищется на отрезке [-1, 0], – , ) < 0 непрерывна на отрезке [-1, 0] как разность непрерывных функций.

Для выбранных отрезков выполнена непрерывность функций ,, и различие знаков этих функции на концах отрезков.

**II. Обоснование выбора значений и [2]**

Возьмём = 0.0001 и = 0.00005

Пределы интегрирования , функций (*i* = 1, 2, 3) были найдены с точностью . Площади под графиком функций в - окрестности точек , не превышают в сумме Данные площади являются погрешностью возникающей из-за точности . Так как значения интегралов считались с точностью , то погрешность одного *i*-ого интеграла будет .

Площадь фигуры представляет собой алгебраическую сумму (со знаками) определенных интегралов от заданных функций на некоторых отрезках: . Значит суммарная погрешность считает по следующей формуле:

( ) + 3 .

Обозначим это выражение как .

Сделаем верхнюю оценку данного выражения, оценив сверху значения функций в точках пересечения:

; ; .

В итоге, получаем следующую оценку:

.

Суммарная погрешность ниже = 0.001, то есть условие выполнено.

**III. Графики заданных кривых**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

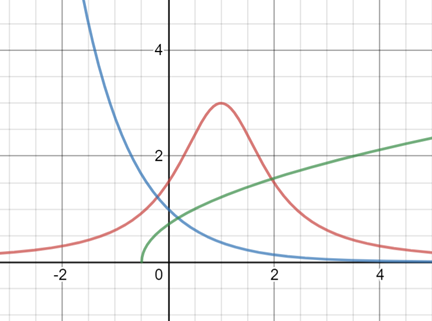


Рис. 1: Плоская фигура, ограниченная графиками заданных уравнений

**Результаты экспериментов**

|  |  |  |
| --- | --- | --- |
| Кривые | *x* | *y* |
| и | 1.9562 | 1.5671 |
| и | 0.1874 | 0.8291 |
| и | -0.2033 | 1.2254 |

Таблица 1: Координаты точек пересечения

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| S = 2.339 | |

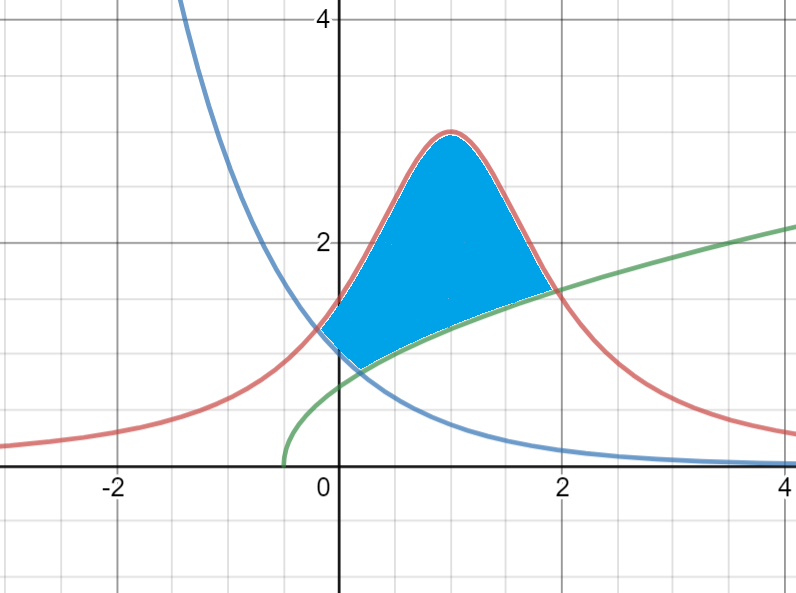


Рис. 2: Плоская фигура, ограниченная графиками заданных уравнений

**Структура программы и спецификация функций**

1. Файл **function.asm** содержит три функции f1, f2, f3, описанные на языке ассемблера. Данные функции получают на вход точку и выдают значение математической функции соответственно. Для работы со значениями с плавающей точкой используется сопроцессор х87.
2. **function.h** – заголовочный файл для файла function.c
3. Файл **integral.c** содержит функцию *integral (f, a, b, eps2)*, написанную на языке С. Функция *integral* должна на выдать значение определённого интеграла от функции f на отрезке [a, b] с точность eps2. В начале программы реализован небольшой цикл for, после которого в переменной *summ* лежит значение равное

0.5, где = 20.

Основой функции является двойной цикл for. Внутренний цикл добавляет в переменную *summ* значения , но только те, которые ранее не вычислялись. Внешний цикл последовательно вычисляет значения при n равном . Для этого достаточно переменную *summ* умножить на переменную *h*, которая делится пополам каждую итерацию. В конце каждой итерации цикла в переменных *int1* и *int2* лежат значения и соответственно. На первой итерации значение уже готово, так как посчитано до двойного цикла. В конце двойного цикла проверяется условие выхода из него (по правилу Рунге проверяется достигло значение интеграла требуемой точности).

1. Файл **main.c** содержит код, позволяющий поддерживать следующие опции командной строки: ключ командной строки -help, печать абсцисс точек пересечения кривых, печать числа итераций, потребовавшихся root, печать исходных функций, печать ординат точек пересечения кривых, вывод площади заданной фигуры (области), тестирование функции root, тестирование функции integral.
2. **Makefile** представлен в следующем разделе.
3. Файл **root.c** содержит функцию *root (f, g, a, b, eps1)*. В начале функции созданы несколько переменных, затем действия функции разбиваются на два случая: если функция *F* (*F = f – g*) возрастает и ее график расположен ниже хорды или если функция *F* убывает и ее график расположен выше хорды, то имеет место случай 1, иначе — случай 2. В первом случае проверяется не попал ли корень в правую eps1-окрестность точки с. Если попал, то значение c + eps1 / 2 будет корнем с нужной точностью. Иначе вызывается root для тех же функций и отрезка [c, b]. Во втором случае рассматривается левая eps1-окрестность точки с и выводится значение c + eps1 / 2, если корень лежит в данной окрестности. Иначе вызывается root для отрезка [a, c].
4. **testintegral.c** содержит три функции, вычисляющие значения математических функций. Подробнее об этих функциях в разделе отладка программы, тестирование функций.
5. **testintegral.h -** заголовочный файл для файла testintegral.c.
6. **testroot.c** содержит три пары функций, вычисляющих значения математических функций. Подробнее об этих функциях в разделе отладка программы, тестирование функций.
7. **testroot.h -** заголовочный файл для файла testroot.c.

**Графическое разбиение программы на компоненты**

**Сборка программы (Make-файл)**

**I.Диаграмма зависимостей**

**II.Текст make-файла**

SYSTYPE=UNIX

OBJFORMAT=elf32

CC=gcc

CFLAGS+=-m32 -Wall -g -O2 -W

all: task6

function.o: function.asm

nasm -g -f $(OBJFORMAT) $< -o $@ -D$(SYSTYPE)

main.o: main.c

$(CC) $(CFLAGS) -c $< -o $@

root.o: root.c

$(CC) $(CFLAGS) -c $< -o $@

integral.o: integral.c

$(CC) $(CFLAGS) -c $< -o $@

testroot.o: testroot.c

$(CC) $(CFLAGS) -c $< -o $@

testintegral.o: testintegral.c

$(CC) $(CFLAGS) -c $< -o $@

task6: function.o main.o root.o integral.o testroot.o testintegral.o

$(CC) $(CFLAGS) $^ -lm -o $@

.PHONY: clean

clean:

-rm -rf \*.o

**Отладка программы, тестирование функций**

Для тестирования функции root используются 3 пары функций: и , и , и .

и пересекаются в точках , (2, 4). Для поиска точки можно взять отрезок [, ]. Для поиска точки можно взять отрезок [, ]. непрерывна на всей числовой прямой. . .

и пересекаются в точке (0, 0). Для поиска этой точки возьмём отрезок [, . . = 2 < 0

и пересекаются в точке (4, 2). Для поиска этой точки можно взять отрезок [3, 5]. = непрерывна на [0, +. = ( + 6) (.

Для тестирования функции integral используются три функции , , . и интегрируемы на всей числовой оси, интегрируема на отрезке, принадлежащем интервалу [0, +).

Например, можно проинтегрировать от 0 до 10, – от 0 до , – от 0 до 5. , , .

**Программа на Си и на Ассемблере**

В архиве содержатся следующие файлы:

1. function.asm
2. function.h
3. integral.c
4. main.c
5. Makefile
6. root.c
7. testintegral.c
8. testintegral.h
9. testroot.c
10. testroot.h
11. Отчёт по заданию №6

Перечисленных файлов достаточно для сборки программы.

**Анализ допущенных ошибок**

При отладке функции integral была выявлена ошибка. Счётчик увеличивался на один при каждой итерации, требовалось увеличение в два раза. В связи с этим число новых точек росло следующим образом: 20, 40, 60, 80, 100… . Но на самом деле при размельчении промежутков разбиения (делении их пополам) возникало новых точек 20, 40, 80, 160… . Таким образом, программа учитывала часть значений из-за чего итоговая сумма имела недобор. Ошибка была выявлена при просмотре значений переменных (было заметно, что цикл не доходит до конца заданного отрезка на итерациях больших двух).

В процессе сборки программы возникала ошибка undefined reference по отношению к функциям из библиотеки math.h (sin, pow, sqrt), которые использовались для создания тестирующих функций. Оказалось, что компоновщику не передавалась опция -lm.

**Список литературы**

[1] Ильин В. А., Садовничий В. А., Сендов Бл. Х. Математический анализ. Т.1 — Москва: Наука, 1985.

[2] Митин И. В., Русаков В. С. Анализ и обработка экспериментальных данных 1998. – 48с.