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 7/2/2

Выполнил:
студент 106 группы
Марков И. В.

Преподаватель:
Корухова Л. С

Москва
2019

Содержание

1. Постановка задачи

Требуется реализовать программу, вычисляющую площадь фигуры, образованной пересечением графиков трех функций:

1. $f_1(x) = \ln(x)$

2. $f_2(x) = -2x + 14$

3. $f_3(x) = 1 / (2 - x) + 6$

с заданной точностью $eps = 10^{-3}$.

Найти точки пересечения требуется численно, применяя метод хорд, подобрав аналитически подходящие отрезки для поиска корней. Для получения площади необходимо выразить ее как сумму определенных интегралов, которые нужно вычислить, воспользовавшись методом трапеций.

Функции $f_i(x)$ должны быть реализованы на языке ассемблера, а остальная часть программы на языке Си. В программе реализованы флаги:

- -help – Вывести краткую справку
- -iterations – Вывести количество итераций, затраченных на поиск корня (в порядке следования корней)
- -crosspoints – Вывести точки пересечения (в порядке следования корней)
- -root_test – Вход в режим отладки функции root. Далее программа просит ввести переменные для запуска функции в формате <f> <g> <a> <b> <eps>. Функция выводит результат. Далее пользователю дается выбор: запустить функцию еще раз с новыми входными данными, либо завершить выполнение программы
- -integral_test – Вход в режим отладки функции integral. Далее программа просит ввести переменные для запуска функции в формате <f> <a> <b> <eps>. Функция выводит результат. Далее пользователю дается выбор: запустить функцию еще раз с новыми входными данными, либо завершить выполнение программы

Сборку программы необходимо производить с использованием утилиты make.

2. Математическое обоснование

2.1. Выбор промежутков поиска корней

Будем искать абсциссы точек пересечения, как решения уравнений вида

$$F_{ij}(x) = f_i(x) - f_j(x),\ i, j = 1::3;\ i < j$$

Для этого выберем такой интервал, на котором $F_{ij}(x)$ будут определены и будет существовать такой x_0, что $F_{ij}(x_0) = 0$.

Путем несложных вычислений получаем, что:

Точка пересечения x_1 1 и 2 функции лежит на отрезке [5;7]

Точка пересечения x_2 1 и 3 функции лежит на отрезке [2.01;3]

Точка пересечения x_3 2 и 3 функции лежит на отрезке [3;5]

Заметим также, что $x_2 < x_3 < x_1$, поэтому площадь вычисляется как $S = i_2 + i_3 - i_1$, где i_j - интеграл от j-й функции с границами в соответствующих точках пересечения.

2.2. Выбор *eps*

Выберем точность вычисления интеграла eps2 и нахождения точек пересечения eps1 таким образом, чтобы общая погрешность была меньше, чем eps.

Общая погрешность равна:

$$\leq 3 * eps2 + eps1 * (\max_{x\in[2.01;7]} |f1(x) - f3(x)| + \max_{x\in[2.01;7]} |f2(x) - f3(x)| + \max_{x\in[2.01;7]} |f2(x) - f3(x)|) \leq 3 * eps2 + eps1 * (3 * \max_{x\in[2.01;7]} |2 * f2(x)|) \leq 3 * eps2 + 2 * 3 * 7 * eps1 = 3 * eps2 + 42 * eps1 \leq 0.001 = eps$$

Положим eps1 = eps2 = 10^{-6}. Получим:

, а, следовательно, eps1 = eps2 = 10^{-6} удовлетворяют поставленному в задаче условию.

3. Результаты экспериментов

В результате выполнения реализованной программы, были получены следующие результаты:

- $f_1(x)$ пересекает $f_2(x)$ в точке x12 ≈ 6.096170
- $f_1(x)$ пересекает $f_3(x)$ в точке x13 ≈ 2.191761
- $f_2(x)$ пересекает $f_3(x)$ в точке x23 ≈ 4.224745
- Получение приведенных точек пересечения с заданной точностью eps = 10^{-6} заняло 4, 205 и 7 итераций соответственно
- Площадь образуемой ими фигуры S ≈ 11.236376

f1(x) = ln(x) *f2(x) = -2x + 14* *f3(x) = 1 / (2 - x) + 6*

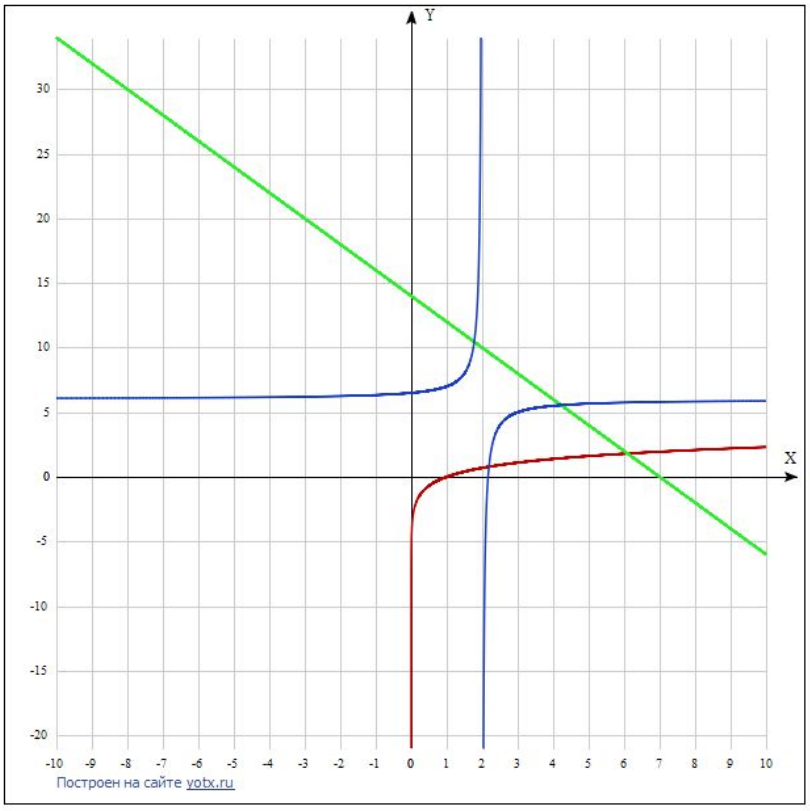

рис.1: Графики функций

4. Структура программы и спецификация функций

Программа реализована в два модуля:

1. `f.asm` - вспомогательный модуль, содержащий реализации функций f1, f2, f3. Сами функции экспортируются в основной модуль в качестве глобальных символов.

2. `main.c` - основной модуль, содержащий реализацию алгоритмов численного интегрирования и нахождения корней уравнений. В данном модуле содержатся следующие функции:

- `int main(int argc, char *argv[])` - отправная точка программы, принимающая на вход опции командной строки
- `double root (double (*f)(double), double (*g)(double), double a, double b, double eps1)` - функция, возвращающая корень уравнения f(x) = g(x) на отрезке [a;b] с точностью eps1
- `double integral (double (*f)(double), double a, double b, double eps2)` - функция, возвращающая значение определенного интеграла функции f(x) на отрезке [a;b] с точностью eps2
- `void root_test (void)` - реализация режима тестирования функции *root*. Запускается в случае приема на вход опции командной строки *-root_test*
- `void integral_test (void)` - реализация режима тестирования функции *integral*. Запускается в случае приема на вход опции командной строки *-integral_test*

5. Makefile и сборка программы

Сборка исполняемых файлов производится с использованием утилиты GNU make. В нем присутствуют несколько целей для сборки:

- `prog.exe` - основная программа
- `all` - собирает prog.exe
- `clean` - очищает промежуточные объектные файлы
- различные промежуточные цели

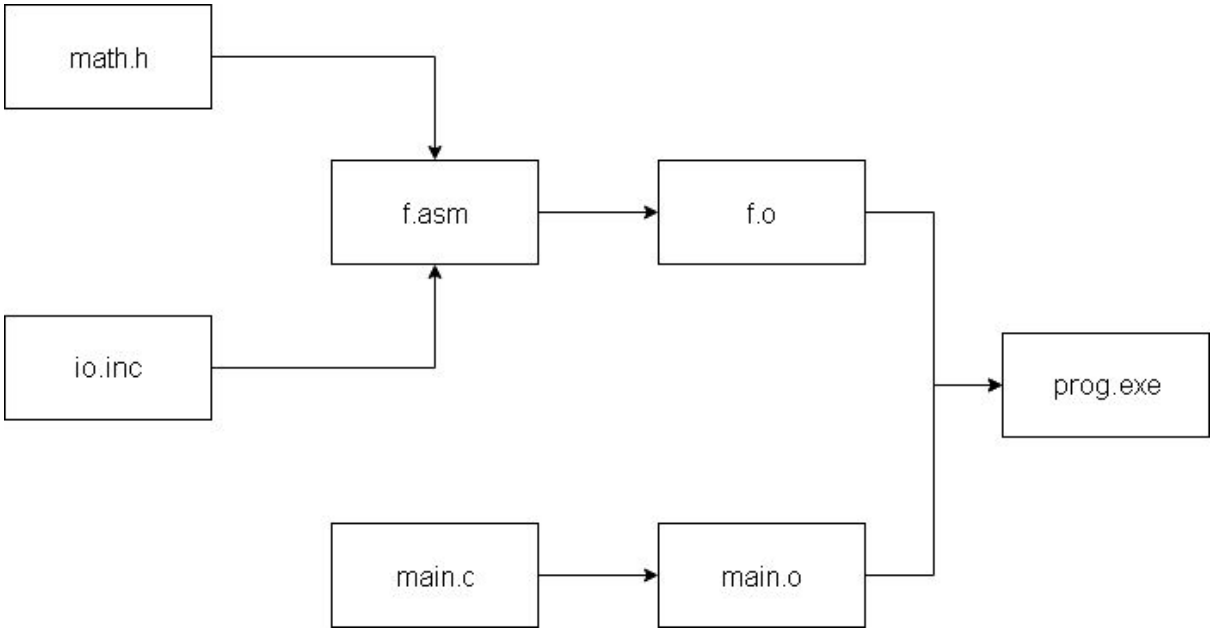

Рис. 2: Зависимость модулей

Текст Makefile:

```
all: prog.exe
prog.exe: main.o f.o
    gcc -o prog.exe main.o f.o -lm
f.o: f.asm
    nasm -fwin32 -o f.o f.asm
main.o: main.c
    gcc main.c -o main.o -c
clean:
    del prog.exe main.o f.o
.PHONY: clean all
```

6. Отладка программы, тестирование функций

Отладка производилась поочередно для каждой функции с заранее известными результатами.

Тесты для функции `integral`:

1. $\int_0^{\pi/2} sinx * dx = 1$

3. $\int_0^1 e^x * dx = e - 1$

Тесты для функции `root`:

2. $sinx = cosx,\ x \in [0;1] \Rightarrow x = \pi/4$
3. $e^x = 2^{x+1},\ x \in [1;4] \Rightarrow x = ln2/(1 - ln2)$

Точность для тестов eps = 10^{-6}. Допустимо отклонение от точного значения в пределах погрешности для всех тестов. Значения результатов тестов можно получить, запустив тестирующий режим соответствующей функции из командной строки.

7. Программа на Си и Ассемблере

Исходные коды программ, а именно:

- Makefile - мэйк-скрипт сборки
- main.c - основной модуль программы
- f.asm - вспомогательный модуль программы
- io.inc - подключаемый заголовочный файл
- math.h - подключаемая библиотека

приложены в архиве вместе с отчетом.

8. Анализ допущенных ошибок

При реализации программы была допущена следующая ошибка: функция integral работала не итерационно, погрешность считалась из формулы остаточного члена. Функция была переписана, соответствующие изменения внесены в отчет.

9. Список литературы

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