###### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

###### Факультет информационных технологий

**Кафедра параллельных вычислений**

ОТЧЕТ

О ВЫПОЛНЕНИИ ПРАКТИЧЕСКОЙ РАБОТЫ

«ИЗУЧЕНИЕ ОПТИМИЗИРУЮЩЕГО КОМПИЛЯТОРА»

студента 2 курса, группы 22204

***Соломенникова Николая Александровича***

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

(доцент)

А.Ю. Власенко

Новосибирск 2023

**СОДЕРЖАНИЕ**

[ЦЕЛЬ 2](#__RefHeading___1)

[ЗАДАНИЕ 2](#__RefHeading___2)

[ОПИСАНИЕ РАБОТЫ 2](#__RefHeading___3)

[ЗАКЛЮЧЕНИЕ 4](#__RefHeading___4)

[Приложение 1 (листинг программы) 4](#__RefHeading___5)

[Приложение 2 (команды для компиляции и запуска программы) 5](#__RefHeading___6)

# ЦЕЛЬ

Изучение основных функций оптимизирующего компилятора, и некоторых примеров оптимизирующих преобразований и уровней оптимизации, а также исследование влияния оптимизационных настроек компилятора GCC на время исполнения программы.

# ЗАДАНИЕ

1. Написать программу на языке C или C++, содержащую функцию, которая реализует алгоритм вычисления функции sin x с помощью разложения в степенной ряд по первым N членам этого ряда. Программа должна принимать значение N через параметр командной строки.

2. Проверить правильность работы программы на нескольких тестовых наборах входных данных.

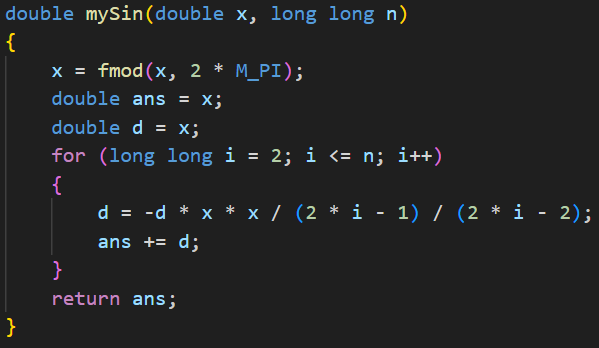
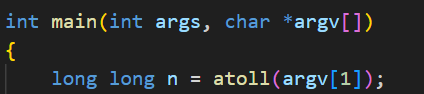
3. Выбрать значение параметра N 0 таким, чтобы время работы функции было от 30 до 60 секунд.

4. Программу скомпилировать компилятором GCC с уровнями оптимизации -O0, -O1, -O2, -O3, -Os, -Ofast, -Og под архитектуру процессора x86 (x86-64).

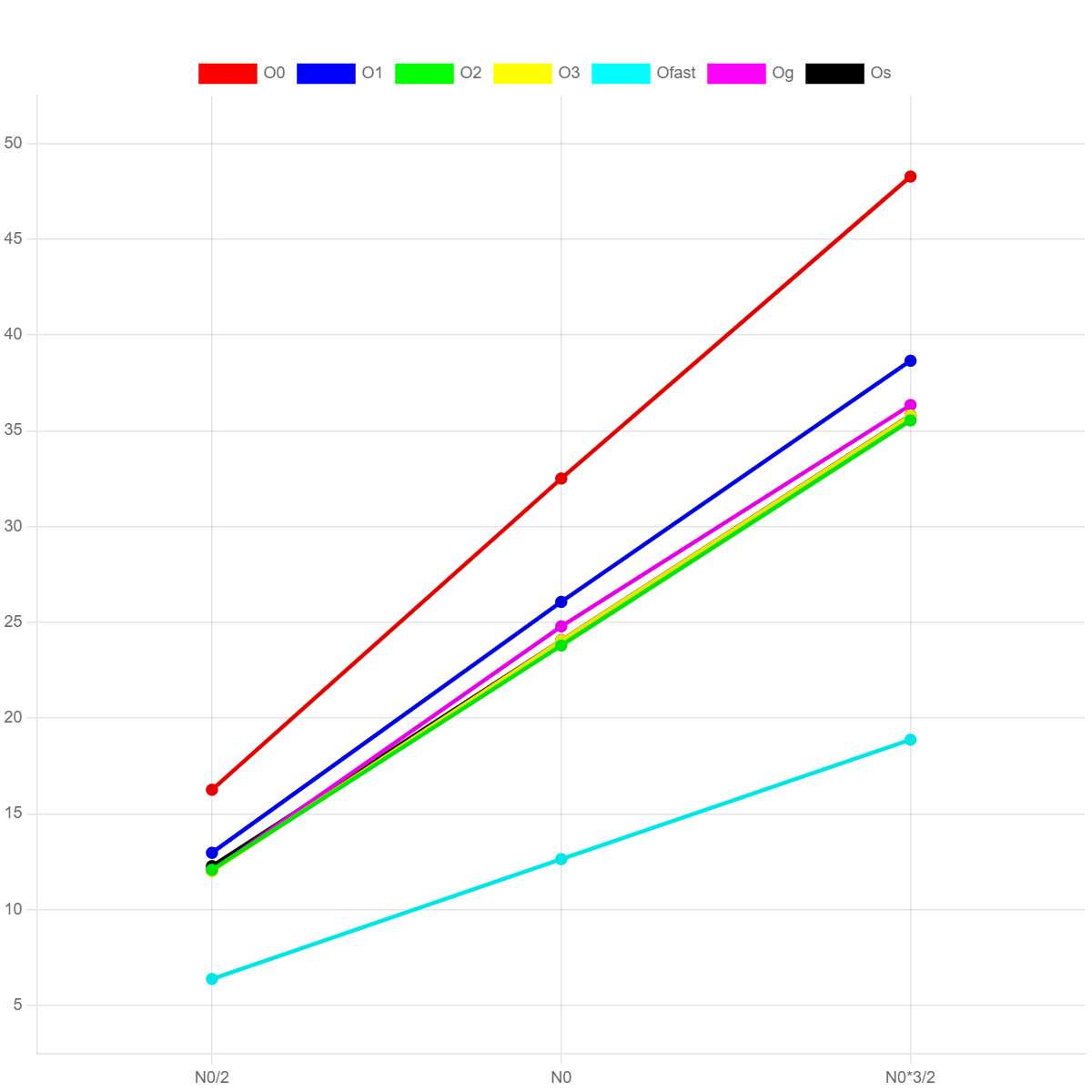
5. Для каждого из семи вариантов компиляции измерить время работы программы при нескольких значениях N (0.5\* N 0 , N 0 , 1.5\* N 0 ).

# ОПИСАНИЕ РАБОТЫ

В данной части приводятся:

1. Для реализации алгоритма вычисляющего sin x через разложение в ряд была написана отдельная функция mySin, принимающая на вход два параметра: угол и количество членов в разложении в ряд.  
     
   В данном фрагменте кода присутствует константа M\_PI из библиотеки math.h, а также функция fmod, которая считает остаток от деления одного нецелого числа на другое. Это нужно чтобы передаваемый в функцию угол x не выходил за пределы интервала (-2PI; +2PI).  
   Был реализован ввод значения N в программу следующим способом:
2. Чтобы убедиться, что программа работает, она была проверенная не нескольких тестах с разными значениями угла и разными значениями n.
3. Для измерения времени работы программы была выбрана стандартная утилита time, так как с помощью неё можно довольно точно определить время, которое работал пользовательский процесс.
4. Путём подбора было выявлено, что чтобы программа с оптимизацией O0 работала в пределах 30 – 60 секунд надо ввести n = 3000000000. Обозначим это число как N0.
5. Было проведено множество тестов программы с разными уровнями компиляции и разными значениями N. Для более достоверного результата проводилось несколько запусков каждого теста и бралось минимальное из полученных времён работы. Данные занесены в таблицу.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | O0 | O1 | O2 | O3 | Ofast | Og | Os |
| N0/2 | 16,255 | 12,963 | 12,079 | 12,015 | 6,376 | 12,026 | 12,266 |
| N0 | 32,504 | 26,065 | 23,787 | 24,047 | 12,633 | 23,786 | 24,052 |
| N0\*3/2 | 48,271 | 38,654 | 35,538 | 35,796 | 18,868 | 36,342 | 35,805 |



# ЗАКЛЮЧЕНИЕ

Исходя из результатов проведённых тестов, видно, что повышение уровня оптимизации не всегда уменьшает время работы программы. Например уровень оптимизации O3 замедляет программу, по сравнению с уровнем O2. Также можно заметить, что с Ofast программа выполняется значительно быстрее, чем с любыми другими оптимизациями. И как и ожидалось, время работы зависит от N линейно. Это чётко видно на графиках.

# Приложение 1 (листинг программы)

#include <iostream>  
#include <math.h>  
double mySin(double x, long long n)  
{  
    x = fmod(x, 2 \* M\_PI);  
    double ans = x;  
    double d = x;  
    for (long long i = 2; i <= n; i++)  
    {  
        d = -d \* x \* x / (2 \* i - 1) / (2 \* i - 2);  
        ans += d;  
    }  
    return ans;  
}

int main(int args, char \*argv[])  
{  
    long long n = atoll(argv[1]);  
    double angle = 100; // for example  
    double res = mySin(angle, n);  
    std::cout << res << std::endl;  
    return 0;  
}

# Приложение 2 (команды для компиляции и запуска программы)

Команды для компиляции:

$ g++ -O0 main.cpp -o mainO0

$ g++ -O1 main.cpp -o mainO1

$ g++ -O2 main.cpp -o mainO2

$ g++ -O3 main.cpp -o mainO3

$ g++ -Ofast main.cpp -o mainOfast

$ g++ -Og main.cpp -o mainOg

$ g++ -Os main.cpp -o mainOs

Пример команды для запуска программы mainO0 со значением N0:

$ time ./mainO0 N0