###### 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 ЛАБОРАТОРНОЙ РАБОТЫ**

«ИЗУЧЕНИЕ ОПТИМИЗИРУЮЩЕГО КОМПИЛЯТОРА»

студента (ки) 2 курса, 19201 группы

**Хаецкой Дарьи Владимировны**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

А.Ю. Власенко

Новосибирск 2020

**СОДЕРЖАНИЕ**

[ЦЕЛЬ 3](#_gjdgxs)

[ЗАДАНИЕ 3](#_30j0zll)

[ОПИСАНИЕ РАБОТЫ 4](#_1fob9te)

[ЗАКЛЮЧЕНИЕ 5](#_3znysh7)

Приложение 1 6

Рисунок 1 7

Приложение 1 9

# ЦЕЛИ

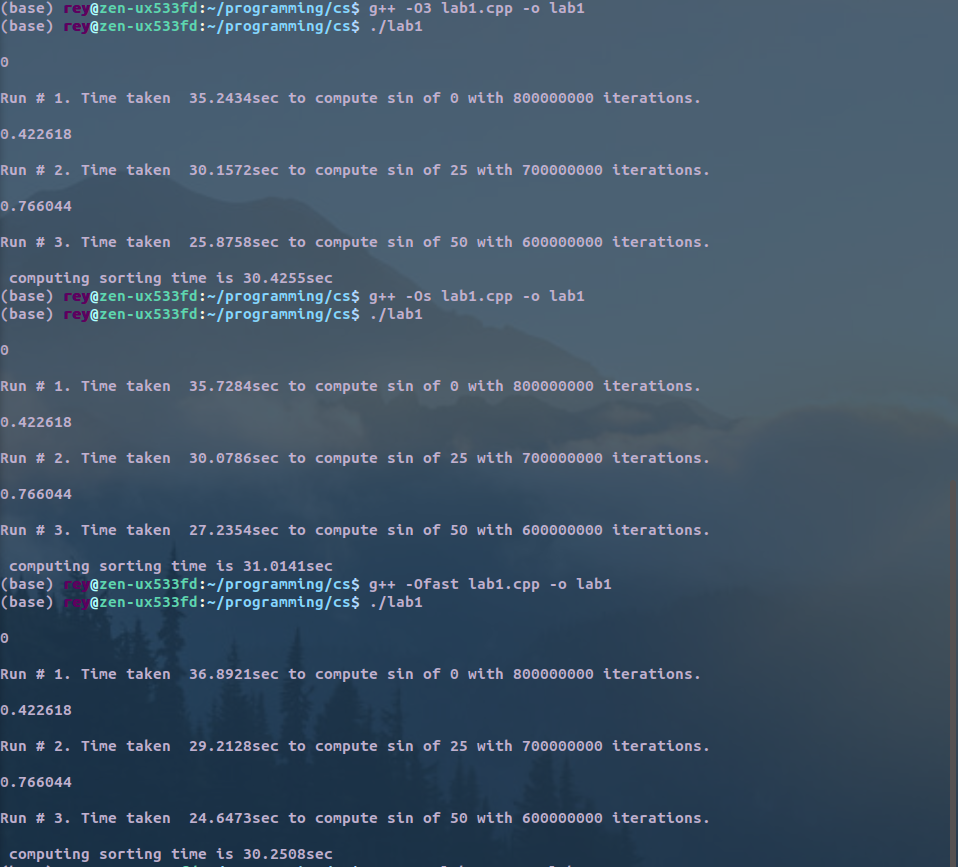
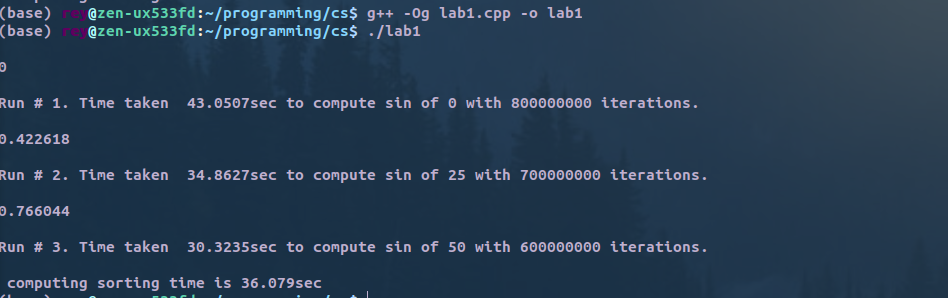
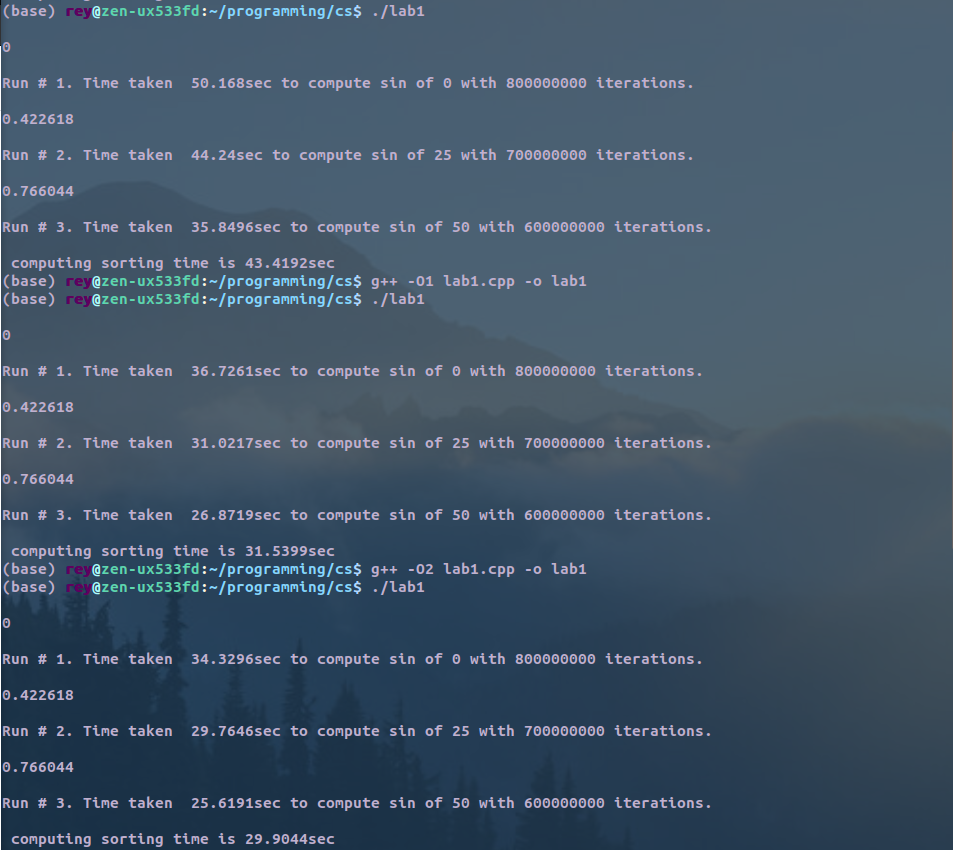
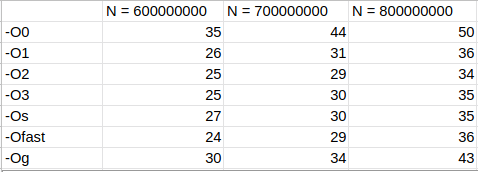
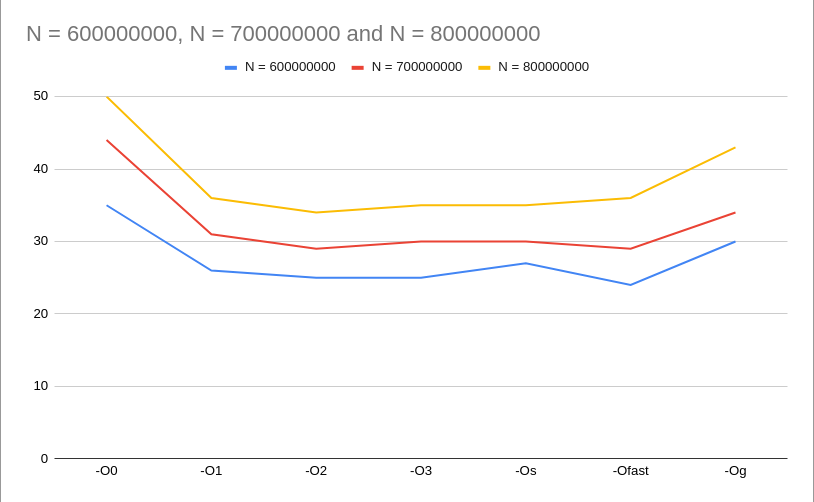
1. Изучение основных функций оптимизирующего компилятора, и некоторых примеров оптимизирующих преобразований и уровней оптимизации.
2. Получение базовых навыков работы с компилятором GCC.
3. Исследование влияния оптимизационных настроек компилятора GCC на время исполнения программы.

# ЗАДАНИЕ

1. Написать программу на языке C или C++, которая реализует выбранный алгоритм из задания.
2. Проверить правильность работы программы на нескольких тестовых наборах входных данных.
3. Выбрать значение параметра N таким, чтобы время работы программы было порядка 30-60 секунд.
4. Программу скомпилировать компилятором GCC с уровнями оптимизации -O0, -O1, -O2, -O3, -Os, -Ofast, -Og под архитектуру процессора x86.
5. Для каждого из семи вариантов компиляции измерить время работы программы при нескольких значениях N.
6. Составить отчет по лабораторной работе.

# ОПИСАНИЕ РАБОТЫ

* 1. 1. На первом этапе был реализован алгоритм подсчёта синуса с помощью ряда Тейлора на языке программирования С++ и проверена корректность получаемых значений.

1. 2. Следующим шагом был добавлен замер времени для трёх различных значений N. (Приложение 1)
2. 3. Произведен замер времени работы программы при разных ключах компиляции****
3. ****
4. 4. Данные о времени занесены в таблицу****
5. 5. Построен график зависимости времени от оптимизации компиляции и параметра N. (Рисунок 2)****

**ЗАКЛЮЧЕНИЕ**

В ходе данной лабораторной работы мы увидели, что использование небезопасных режимов компиляции не всегда оправданно и дает наилучший результат, оптимального значения времени мы достигаем уже при компилировании программы с ключом -О2.

**Приложение 1: *Исходный код программы для тестирования***

|  |
| --- |
| #include <ctime>  #include <iostream>  #include <cstdlib>  #include <vector>  #include <cmath>  #define pi 3.14159265  using namespace std;  double toRadians(double degrees){  return (pi / 180) \* degrees;  }  long double sinus(double x, int N){  long double current, prev = x, sin = 0;  for (int n = 1, f = 1; n < N; n++, f+=2){  sin += prev;  current = (abs(prev) \* pow(-1, n) \* pow(x,2)) / ((f + 1) \* (f + 2));  prev = current;  }  return sin;  }  int main() {  struct timespec start, end;  double totalTime = 0;  int runs = 4;  int degrees = 45;  int N = 800000000;  for (int i = 1; i < runs; i++, N -= 100000000){  clock\_gettime(CLOCK\_MONOTONIC\_RAW, &start);  cout << endl << sinus(toRadians(degrees), N) << endl << endl;  cout << "Run # "<< i << ". Time taken ";  clock\_gettime(CLOCK\_MONOTONIC\_RAW, &end);  double time = end.tv\_sec-start.tv\_sec + 0.000000001 \* (end.tv\_nsec - start.tv\_nsec);  cout << " " << time << "sec to compute sin of "<< degrees << " with "<< N << " iterations." << endl;  totalTime += time;  }  cout << "\n computing time is " << totalTime/(runs - 1) << "sec\n";  return 0;  } |