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

«ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ И ИЗУЧЕНИЕ ОПТИМИЗИРУЮЩЕГО КОМПИЛЯТОРА»

студента 2 курса, группы 21206

Балашова Вячеслава Вадимовича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Кандидат технических наук

А.Ю. Власенко

Новосибирск 2022

Оглавление

Цель

1. Изучение методики измерения времени работы подпрограммы
2. Изучение приемов повышения точности измерения времени подпрограммы
3. Изучение способов измерения времени работы подпрограммы
4. Измерение времени работы подпрограммы в прикладной программе
5. Изучение основных функций оптимизирующего компилятора, и некоторых примеров оптимизирующих преобразований и уровней оптимизации
6. Получение базовых навыков работы с компилятором GCC
7. Исследование влияния оптимизационных настроек компилятора GCC на время исполнения программы

Задачи

1. Написать программу на языке C или C++, содержащую **функцию**, которая реализует вычисление функции sin(x) с помощью разложения в ряд по первым N членам этого ряда:

$$\sin(x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1} x^{2n-1}}{(2n-1)!} = x - \frac{\boldsymbol{x^3}}{\boldsymbol{3!}} + \frac{\boldsymbol{x^5}}{\boldsymbol{5!}} - \frac{\boldsymbol{x^7}}{\boldsymbol{7!}} + \cdots + \frac{\boldsymbol{(-1)^{n-1}}}{\boldsymbol{(2n-1)!}} \boldsymbol{x^{2n-1}} + \cdots;$$

$$\boldsymbol{-\infty < x < +\infty}$$

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)**.**
6. Составить отчет по лабораторной работе. Отчет должен содержать следующее:
    a. Титульный лист.
    b. Цель лабораторной работы.
    c. Вариант задания.
    d. Описание методики для определения времени работы программы.
    e. Результаты измерения времени работы программы при различных значениях параметра N с уровнями оптимизации **-O0**, **-O1**, **-O2**, **-O3, -Os, -Ofast, -Og** (лучше в табличном виде).
    f. Графики зависимости времени выполнения программы с уровнями оптимизации **-O0**, **-O1**, **-O2**, **-O3, -Os, -Ofast, -Og** от параметра N.

g. Полный компилируемый листинг реализованной программы, команды для ее компиляции и запуска.
h. Вывод по результатам лабораторной работы.

Описание работы

Ход работы:

1. На языке программирования C++ был написан алгоритм вычисления функции sin(x) – синус от числа «x», заданного в радианах с помощью разложения данной функции в ряд МакЛорена (исходный код находится в приложении 1).

2. Была проверена правильность работы программы на нескольких тестовых значениях «x» и некоторых значениях N, не превосходящих unsigned long long. Также был выбран параметр N_0 = 100000, при котором программа без дополнительных флагов уровня компиляции выполнялась за секунд, что лежит в требуемом диапазоне (от 30 до 60 секунд).

3. Также с помощью библиотеки «ctime» было интегрировано время измерения работы подпрограммы. С помощью функции «clock()» берется количество временных тактов, прошедших с начала запуска программы. А с помощью макроса «CLOCKS_PER_SEC» можно получить время в секундах.

4. Подпрограмма была скомпилирована с помощью G++ с разными уровнями оптимизации (O0, O1, O2, O3, Os, Ofast, Og). Все скомпилированные файлы имеют отличительные названия для их уровня оптимизации.

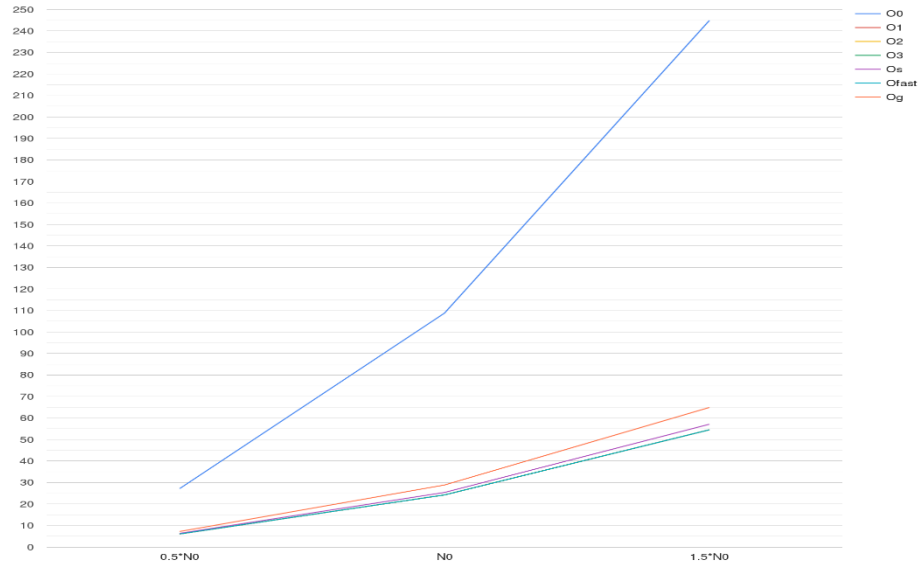
5. Все исполняемые файлы с различными уровнями оптимизации были запущены с параметрами x = 10 и N = {$0.5 * N_0$, N_0, $1.5 * N_0$}. Было проведено три измерения для каждого исполняемого файла. Ниже представлена таблица с результатами измерения:

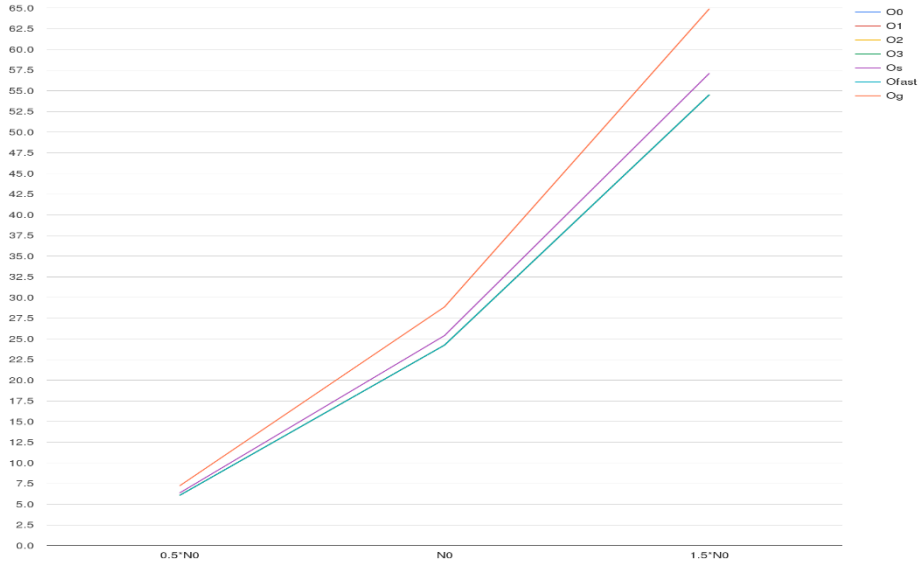
Табл. 1. Результаты измерения времени в зависимости от значения параметра N и уровня оптимизации

N	O0	O1	O2	O3	Os	Ofast	Og
$0.5 * N_0$	27.149967	6.061119	6.059156	6.060209	6.351752	6.058475	7.214765
N_0	108.783502	24.230845	24.233271	24.231695	25.384141	24.231236	28.850568
$1.5 * N_0$	244.910905	54.523959	54.525231	54.522181	57.110325	54.526046	64.917933

6. В программе Microsoft Excel был построен график зависимости времени от N для разных уровней оптимизации:

График 1. Результаты измерения времени в зависимости от значения параметра N и уровня оптимизации (С уровнем O0 и без)





ЗАКЛЮЧЕНИЕ

В ходе выполнения практической работы была проанализирована зависимость времени работы подпрограммы от уровня оптимизации компилятора. Изучение проводилось с помощью подпрограммы, вычисляющей значение sin(x) с помощью разложения в ряд, написанной на языке программирования C++, библиотеки «ctime», с помощью которой замерялось время, и компилятора G++.

Было выяснено влияние уровней оптимизации компилятора на скорость работы подпрограммы путем замера времени работы подпрограмм на разных значениях входного параметра N. Результатом исследования стали построенные таблица и график. Также были получены базовые навыки работы с компилятором G++.

Приложение 1 (Исходный код программы)

```
#include <cmath>
#include <ctime>
#include <iostream>

const double pi = 3.14159265358979323846;

using namespace std;

long double power(long double x, size_t n)
{
   long double forRet = 1;
   for (size_t i = 1; i <= n; i++)
   {
      forRet *= x;
   }

   return forRet;
}

long double factorial(size_t n)
{
   long double forRet = 1;
   for (size_t i = 1; i <= n; i++)
   {
      forRet *= i;
   }

   return forRet;
}

long double Sin(long double x, size_t n)
{
   int sign = (x < 0) ? -1 : 1;
   x = fmod(fabs(x),2 * pi);
   if (x > pi)
   {
      x -= pi;
      sign *= -1;
   }
   if (x > pi/2)
   {
      x = pi - x;
   }

   long double result = 0;
   for (size_t i = 1; i <= n; i++)
   {
      long double buf = power(-1, i - 1) * power(x, 2 * i - 1) / factorial(2 * i - 1);
      if (isnan(buf))
      {
```

```
            result += 0;
        }
        else
        {
            result += buf;
        }
    }

    return result * sign;
}

int main(int argc, char* argv[])
{
    size_t numberOfElements = stoull(argv[1]);
    long double x = 10;

    cout << fixed << dec << "Answer: " << Sin(x, numberOfElements) << endl;
    cout << "Time: " << (long double)(clock()) / CLOCKS_PER_SEC << "s" << endl;

    return 0;
}
```