**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**

«название работы»

студента (ки) \_\_\_\_2\_\_\_\_\_ курса, \_\_19211\_\_ группы

**Олимпиева Юрия Юрьевича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

**Н.А. Беляев**

2020\_год

**СОДЕРЖАНИЕ**

Цели……….………………………………………………………………….3

Задание...….………………………………………………………………….3

Описание работы....………………………………………………………….3

Вывод………….......………………………………………………………….3

Заключение.………………………………………………………………….3

Приложение 1………………………………………………………………..4

Приложение 2………………………………………………………………..5

Приложение 3……………………………………………………………...5-6

**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.Взять программу на языке C из прошлой лабораторной работы, которая реализует **алгоритм вычисления функции e^x с помощью разложения в ряд Маклорена по первым N членам этого ряда**. (приложение 1)

2. Выбрать значение параметра N таким, чтобы время работы программы было порядка 30 секунд (предположительно – N следует взять в 2 раза больше, так как сложность алгоритма - линейная).

4. Провести измерения времени работы для разных уровней оптимизации

(-O0, -O1, -O2, -O3, -Os, -Ofast, -Og под архитектуру процессора x86.)

5. Построить графики зависимости времени выполнения подпрограммы от N для разных уровней оптимизации (приложение 3).

5. Составить отчет по лабораторной работе.

**Описание работы**

* Эмпирическим путём подобрал необходимое значение переменной для ограничения счетчика цикла так, чтобы результирующее время выполнения было близко к 30 секундам.
* Для каждого уровня оптимизации замерил время для нескольких значений N для выявления зависимости.
* Занес данные в таблицу (приложение 2).
* Составил отчёт о выполненной работе.

**Вывод**

По данным в таблице видно, что при любом уровне оптимизации

сложность алгоритма подпрограммы O(n). Наибольшую скорость выполнения программы достигается при уровне оптимизации –Ofast

и –O3 (Путем раскручивания цикла и использования регистров-векторов для вещественной арифметики).

**Заключение**

Изучил основы работы с компилятором GCC, возможности и способы оптимизации программы,

Исследовал влияние оптимизационных настроек на время выполнения программы.

Приложение 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <malloc.h>

#include <memory.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define CIRCLE\_COUNTER 6400000

using namespace std;

double McLorenExp(double x, int N) {

double result = 0.0;

double element = 1.0;

for (int i = 0; i < N; i++) {

result += element;

element = element \* x / (i + 1);

}

return result;

};

int main(void) {

double \* arr = (double \*)malloc(CIRCLE\_COUNTER \* sizeof(double));

int Start = clock();

for (int i = 0; i < CIRCLE\_COUNTER; i++)

arr[i] = McLorenExp(1.0/(double)i, 1000);

int Finish = clock();

printf("Result time:\t%f", ((float)(Finish - Start) / (float)CLK\_TCK) /\*/CIRCLE\_COUNTER\*/);

FILE \* out = fopen("out.txt", "wb");

for(int i = 0; i < CIRCLE\_COUNTER; i++)

fprintf(out, "%lf", arr[i]);

free(arr);

fclose(out);

return 0;

}

Команды компиляции:

gcc -O0 labECM.cpp

gcc -O1 labECM.cpp

gcc -O2 labECM.cpp

gcc -O3 labECM.cpp

gcc -Ofast labECM.cpp

gcc -Og labECM.cpp

gcc -Os labECM.cpp

Команды для запуска:

a.exe

Приложение 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Уровень оптимизации\N** | **3200000** | **4000000** | **4800000** | **5600000** | **6400000** |
| O0 | 29.299999 | 36.499001 | 43.883999 | 50.844002 | 58.539001 |
| O1 | 16.795000 | 20.669001 | 24.587000 | 30.218000 | 32.978001 |
| O2 | 16.607000 | 20.712999 | 24.497999 | 29.080000 | 32.951000 |
| O3 | 4.162000 | 5.270000 | 6.376000 | 7.306000 | 8.296000 |
| Ofast | 4.156000 | 5.196000 | 6.245000 | 7.261000 | 8.212001 |
| Os | 16.472000 | 20.840000 | 24.712999 | 28.834999 | 33.213001 |
| Og | 16.552999 | 20.514999 | 24.709000 | 28.789000 | 33.000999 |

Приложение 3