###### 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 2 курса, 23208 группы

**Веретенникова Андрея Александровича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Д.С. Иванишкин

Новосибирск 2024

**СОДЕРЖАНИЕ**

[ЦЕЛЬ 3](#_Toc18443921)

[ЗАДАНИЕ 3](#_Toc18443922)

[ОПИСАНИЕ РАБОТЫ 4](#_Toc18443923)

[ЗАКЛЮЧЕНИЕ 5](#_Toc18443924)

[Приложение 1. *Листинг* 6](#_Toc18443925)

# 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

*Вариант №1. Алгоритм вычисления числа Пи с помощью разложения в ряд (ряд Грегори-Лейбница) по формуле Лейбница N первых членов ряда.*

# ОПИСАНИЕ РАБОТЫ

*Пошаговое описание выполненной работы:*

*Возьмем программу из лабораторной работы №1 «*ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ*»,*

* *Выбрать значение параметра N таким, чтобы время работы программы было порядка 30-60 секунд,*
* *Скомпилировать программу, используя разные уровни оптимизации,*
* *Для каждого из уровней оптимизации измерить время работы программы при нескольких значениях N,*
* *Составить график зависимости времени выполнения программы с уровнями оптимизации* ***-O0****,* ***-O1****,* ***-O2****,* ***-O3****,* ***-Os****,* ***-Ofast****,* ***-Og*** *от параметра N.*

*Выбранная методика: утилиты time, gcc.*

*Cтроки компиляции и запуска разработанных программ:*

1. *Оптимизация -O0:*

* *компиляция:* gcc -O0 Pi\_O0.c -o Pi\_O0 -Wall
* *запуск с замером времени:* time ./Pi\_O0

1. *Оптимизация -O1:*

* *компиляция:* gcc -O1 Pi\_O1.c -o Pi\_O1 -Wall
* *запуск с замером времени:* time ./Pi\_O1

1. *Оптимизация -O2:*

* *компиляция:* gcc -O2 Pi\_O2.c -o Pi\_O2 -Wall
* *запуск с замером времени:* time ./Pi\_O2

1. *Оптимизация -O3:*

* *компиляция:* gcc -O3 Pi\_O3.c -o Pi\_O3 -Wall
* *запуск с замером времени:* time ./Pi\_O3

1. *Оптимизация -Os:*

* *компиляция:* gcc -Os Pi\_Os.c -o Pi\_Os -Wall
* *запуск с замером времени:* time ./Pi\_Os

1. *Оптимизация -Ofast:*

* *компиляция:* gcc -Ofast Pi\_Ofast.c -o Pi\_Ofast -Wall
* *запуск с замером времени:* time ./Pi\_Ofast

1. *Оптимизация -Og:*

* *компиляция:* gcc -Og Pi\_Og.c -o Pi\_Og -Wall
* *запуск с замером времени:* time ./Pi\_Og

*Числовые данные:*

1. *Оптимизация -O0:*
   1. *N = 2.000.000.000*

*Время: 14,717 секунд*

* 1. *N = 4.000.000.000*

*Время: 29,146 секунд*

* 1. *N = 6.000.000.000*

*Время: 43,572 секунд*

1. *Оптимизация -O1:*
   1. *N = 2.000.000.000*

*Время: 14,102 секунд*

* 1. *N = 4.000.000.000*

*Время: 27,889 секунд*

* 1. *N = 6.000.000.000*

*Время: 41,684 секунд*

1. *Оптимизация -O2:*
   1. *N = 2.000.000.000*

*Время: 14,083 секунд*

* 1. *N = 4.000.000.000*

*Время: 27,890 секунд*

* 1. *N = 6.000.000.000*

*Время: 41,698 секунд*

1. *Оптимизация -O3:*
   1. *N = 2.000.000.000*

*Время: 14,093 секунд*

* 1. *N = 4.000.000.000*

*Время: 28,151 секунд*

* 1. *N = 6.000.000.000*

*Время: 41,685 секунд*

1. *Оптимизация -Os:*
   1. *N = 2.000.000.000*

*Время: 20,359 секунд*

* 1. *N = 4.000.000.000*

*Время: 40,875 секунд*

* 1. *N = 6.000.000.000*

*Время: 60,621 секунд*

1. *Оптимизация -Ofast:*
   1. *N = 2.000.000.000*

*Время: 14,087 секунд*

* 1. *N = 4.000.000.000*

*Время: 27,883 секунд*

* 1. *N = 6.000.000.000*

*Время: 41,955 секунд*

1. *Оптимизация -Og:*
   1. *N = 2.000.000.000*

*Время: 14,344 секунд*

* 1. *N = 4.000.000.000*

*Время: 27,891 секунд*

* 1. *N = 6 .000.000.000*

*Время: 41,955 секунд*

*Графики и диаграммы:*

# ЗАКЛЮЧЕНИЕ

*В ходе лабораторной работы я изучил основные функции оптимизирующего компилятора, и некоторые примеры оптимизирующих преобразований и уровней оптимизации, получил базовые навыки работы с компилятором GCC и исследовал влияние оптимизационных настроек компилятора GCC на время исполнения программы.*

# Приложение 1. *Листинг*

