###### 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**

«ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ»

студента 2 курса, 22201 группы

**Рабецкого Валерия Дмитриевича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

А.С. Матвеев

Новосибирск 2023

**СОДЕРЖАНИЕ**

[ЦЕЛЬ 3](#_gjdgxs)

[ЗАДАНИЕ 3](#_30j0zll)

[ОПИСАНИЕ РАБОТЫ 4](#_1fob9te)

[ЗАКЛЮЧЕНИЕ 5](#_3znysh7)

ПРИЛОЖЕНИЕ 1. Исходный код программы для тестирования 6

# ЦЕЛЬ

Изучение методов измерения времени работы программы и оптимизации этих измерений, а также практическое применение данных навыков.

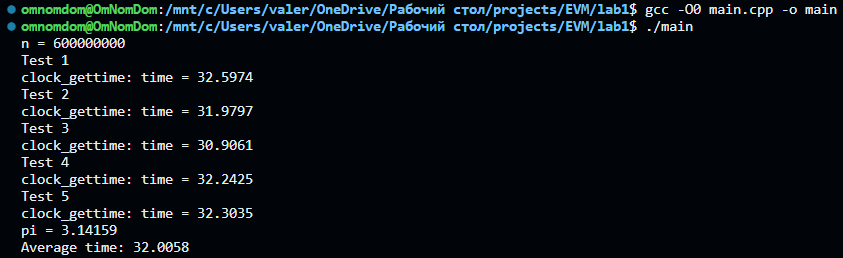
# ЗАДАНИЕ

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 15 секунд.
4. По приведенной методике определить время работы подпрограммы тестовой программы с относительной погрешностью не более 1%.
5. Составить отчет по лабораторной работе

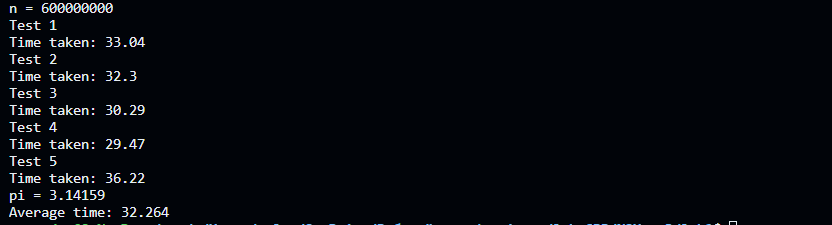
# ОПИСАНИЕ РАБОТЫ

1. 1. На первом этапе был реализован алгоритм подсчёта числа Пи с помощью разложения в ряд (ряд Грегори-Лейбница) по формуле Лейбница N первых членов ряда: на языке программирования С++ и проверена корректность получаемых значений.
2. 2. Следующим шагом была выполнена проверка его работы (Приложение 1). Время замеряется пять раз и ищется среднее время выполнения.

Замер времени произведен с помощью функции clock\_gettime()



3. Произведен замер времени работы программы с помощью функции time()



**ЗАКЛЮЧЕНИЕ**

В ходе данной лабораторной работы мы познакомились с различными методами измерения работы программ и научились пользоваться ими на практике.

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

|  |
| --- |
| #include <iostream>  #include <cmath>  #include <ctime>  #include <sys/times.h>  #include <unistd.h>  using std::cout;  using std::endl;  int main()  {  double averageTime = 0;  double pi;  long long n = 600000000;  cout << "n = " << n << endl;  for (int j = 0; j < 5; ++j)  {  pi = 0;  double temp;  struct tms start, end;  long long clocks\_per\_sec = sysconf(\_SC\_CLK\_TCK);  long long clocks;  times(&start);  // timespec start\_gettime, end\_gettime;  // clock\_gettime(CLOCK\_MONOTONIC\_RAW, &start\_gettime);  for (int i = 0; i < n; i++)  {  temp = ((pow(-1, i)) / (2 \* i + 1));  pi += temp;  }  pi = 4 \* pi;  times(&end);  clocks = end.tms\_utime - start.tms\_utime;  cout << "Test " << j + 1 << endl;  cout << "Time taken: " << (double)clocks / clocks\_per\_sec << endl;  averageTime += (double)clocks / clocks\_per\_sec;  // clock\_gettime(CLOCK\_MONOTONIC\_RAW, &end\_gettime);  // cout << "clock\_gettime: time = " << end\_gettime.tv\_sec - start\_gettime.tv\_sec + 0.000000001\*(end\_gettime.tv\_nsec - start\_gettime.tv\_nsec) << endl;  // averageTime += end\_gettime.tv\_sec - start\_gettime.tv\_sec + 0.000000001\*(end\_gettime.tv\_nsec - start\_gettime.tv\_nsec);  }  cout << "pi = " << pi << endl;  cout << "Average time: " << averageTime / 5 << endl;  return 0;  } |