###### 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, 23201 группы

**Смирнова Гордея Андреевича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

А.С. Матвеев

Новосибирск 2024

**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.

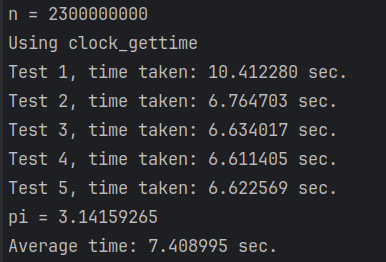
# ЗАДАНИЕ

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

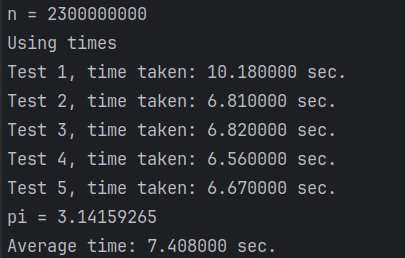
# ОПИСАНИЕ РАБОТЫ

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 библиотечной функции clock\_gettime()



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



**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 <stdio.h>  #include <stdlib.h>  #include <time.h> // for clock\_gettime  #include <sys/times.h> // for times  #include <unistd.h> // for sysconf  #define N\_CONST 2300000000  #define UCGT 1 // 0 to use times, 1 to use clock\_gettime  double PiLeibniz(long long const n) {  double result = 0;  for (long long i = 0; i < n; i++) {  double t = 1.0 / (2.0 \* i + 1.0);  if (i % 2 != 0) t = -t;  result += t;  }  return result \* 4;  }  int main(void) {  long long const n = N\_CONST;  double result = 0;  double avgTime = 0;  printf("n = %lld\n", n);  if (UCGT) printf("Using clock\_gettime\n");  else printf("Using times\n");  for (int i = 1; i <= 5; ++i) {  double time;  if (UCGT) {  struct timespec start, end;  clock\_gettime(CLOCK\_MONOTONIC\_RAW, &start);  result = PiLeibniz(n);  clock\_gettime(CLOCK\_MONOTONIC\_RAW, &end);  time = end.tv\_sec - start.tv\_sec + 0.000000001 \* (end.tv\_nsec - start.tv\_nsec);  }  else { // using times  struct tms start, end;  long clocks\_per\_sec = sysconf(\_SC\_CLK\_TCK);  times(&start);  result = PiLeibniz(n);  times(&end);  long clocks = end.tms\_utime - start.tms\_utime;  time = (double)clocks / clocks\_per\_sec;  }  printf("Test %d, time taken: %lf sec.\n", i, time);  avgTime += time;  }  printf("pi = %.8lf\n", result);  printf("Average time: %lf sec.\n", avgTime / 5);  return 0;  } |