ГУАП
КАФЕДРА № 34

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доц., канд. техн. наук		К. А. Жиданов
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

Изучение методов разработки консольных приложений

по курсу: Языки программирования

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	3145	7.04	В. В. Пуговкин
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

Цель работы:

Изучить методы разработки консольных приложений, способы их запуска и обработки кодов возврата.

Вариант 4 - Значение PI с заданной точностью (ряд Леймбница)
Ход работы:

1. Реализовал функцию на языке C, выполняющую заданные вычисления. (Ряд Лейбница)

```
double lei(double count) {

    int count1;
    double pi, a, b;

    count1 = 2;

    for(count; count > 0; count-=2) {

        if(count1 == 2) {
            a = 4 - 1;
            b = 1;
            pi = 4/b - 4/a;
        }

        if(count1 > 2) {
            a = a + 4;
            b = b + 4;
            pi = pi + (4/b-4/a);
        }

        count1 += 2;
    }

    return pi;
}
```

```
Число членов в ряду      Значение Pi
2                 2.6666666666666696273
4                 2.89523809523809561028
6                 2.97604617604617649462
8                 3.01707181707181737451
10                3.04183961892940235572
```

2. Так как ряд у нас касается константы, то у нас нет эталонных значений, так как алгоритм с увеличением подаваемых значений приближается к значению числа Pi. Будем с значением Pi и сравнивать для отчетности.

3. Реализовал трестирующую функцию, которая будет сравнивать получаемые нашей функцией значения с эталонным результатом (в нашем случае - число Pi округленное до двадцати знаков).

```
double test_lei() {

    int r = 0;
    r = r || nabs(lei(2) - 3.141592653589793) >= 0.000000000000001;
    r = r || nabs(lei(4) - 3.141592653589793) >= 0.000000000000001;
    r = r || nabs(lei(6) - 3.141592653589793) >= 0.000000000000001;
    r = r || nabs(lei(8) - 3.141592653589793) >= 0.000000000000001;
    r = r || nabs(lei(10) - 3.141592653589793) >= 0.000000000000001;
    r = r || nabs(lei(12) - 3.141592653589793) >= 0.000000000000001;

    return r;
}
```

В начале программы мы создали функцию, которая возвращает модуль числа, чтобы в результате видеть положительную разницу между значениями. После выполнения мы можем увидеть как отличается значения числа Pi по мере увеличения числа членов в ряду Лейбница. При увеличении значения числа Pi приближается к действительному.

В итоге программа состоит из трех функций. Первая для получения модуля числа, вторая функция считает значение числа Pi с заданным значением членов ряда Лейбница, третья - main функция. Она запускает тестирующую функцию для проверки ответа.

Вывод:

После завершения ЛР мы научились разрабатывать консольные приложения. Реализовывать тестирующий функции для удостоверения работы основной программы.