**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Программирование»**

**Тема: РЕКУРРЕНТНЫЕ СООТНОШЕНИЯ. ЦЕЛОЧИСЛЕННЫЕ ТИПЫ ДАННЫХ**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6382 |  | Вайгачёв А.О |
| Преподаватель |  | Шолохова О.М. |

**Цель работы.**

Получение навыков использования операторов циклов с неизвестным количеством повторений. Ознакомление с особенностями вычислений с целыми числами.

**Постановка задачи.**

Для заданного целого n ≥ 0 вычислить:

(2n + 1)!! = 1\*3\*5\*...\*(2k + 1);

**Основные теоретические положения.**

1)Я столкнулся с проблемой переполнения данных. Для реализации я использовал такой тип переменных как unsigned long long int. Он отличается от переменной вида long long тем ,что первое использует столько же памяти, сколько первый тип данных , но только положительные значения.

2)Использовав цикл while, я добился правильного ввода исходных данных

3)Решил задачу с помощью рекуррентной последовательности pn=pN+1\*k , где k=2n+1

4) unsigned short 0-255

unsigned int 216

unsigned long 0-4 294 967 295

unsigned long long 0-18 446 744 073 709 551 615

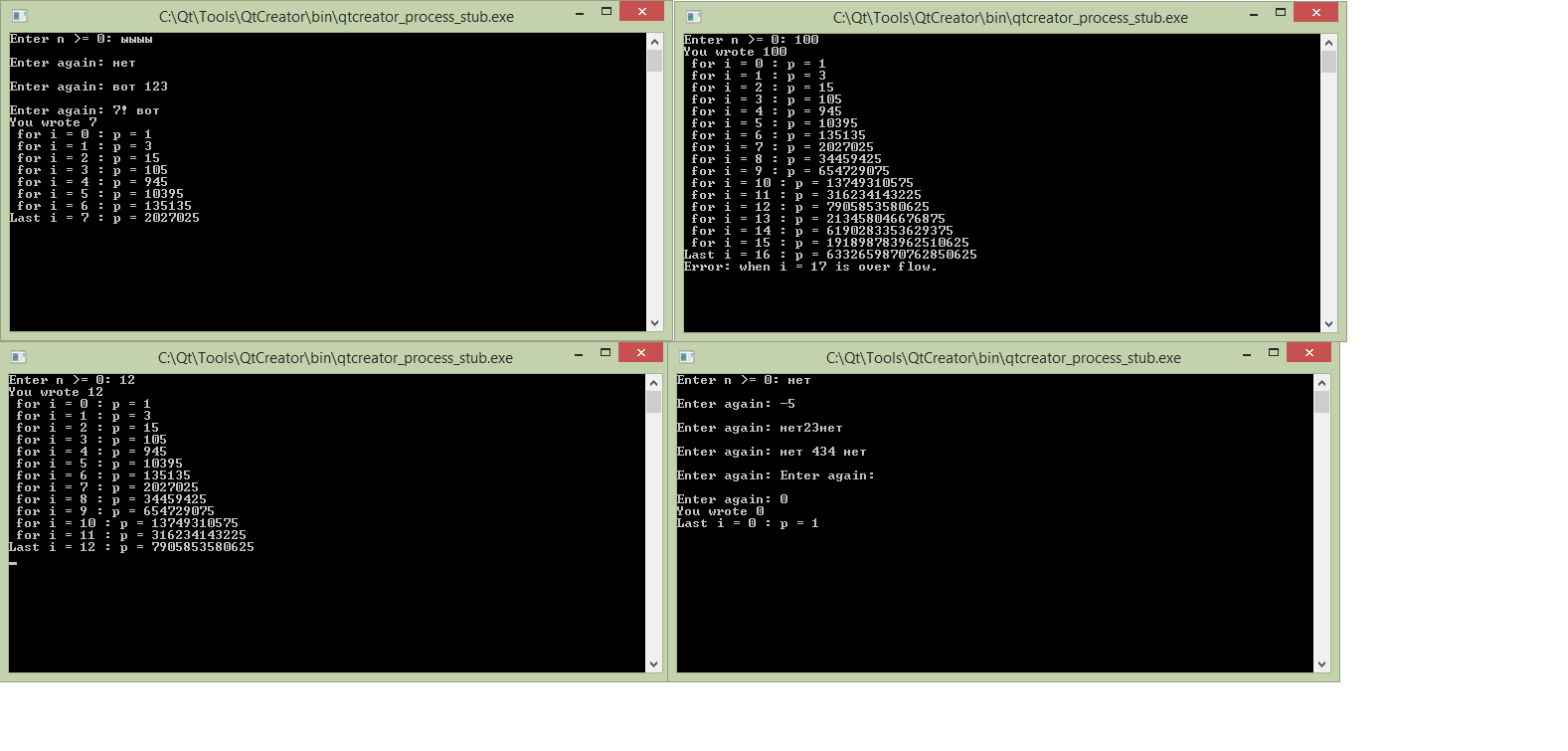
**Спецификация программы.**

Программа предназначена для вычисление геометрической прогрессии не четных чисел.Программа написана на языке C с использованием компилятора qmake Входными данными для программы является значение угла в радианах, вводимое пользователем с клавиатуры. Проверка значений, вводимых пользователем присутствует, поведение программы при некорректном будет таковым: программа будет предлагать ввести значения до тех пор , пока пользователь не введёт не отрицательное число. Выходными данными являются:

1. Порядок итератора ( шага )

2. Значение прогрессии на текущем шаге

3. В случае переполнения памяти программа выводит последний шаг , последнее значение и завершает работу.

Пример диалога с пользователем:

Глобальные переменные:

1. short isOverflow - флаг переполнения принимает значения 0-1. Если значение 1 , то значит вычисления прервались из-за переполнения

**Тестирование.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ввод значения: | 12 | 0 | 5 | 88 | 2 |
| Вывод: | 7905853580625 | 1 | 10395 | Ошибка | 3 |

**Выводы.**

В ходе выполнения лабораторной работы были получены были получены навыки использования операторов циклов с неизвестным количеством повторений. Ознакомились с особенностями вычислений с целыми числами. Узнали некоторые особенности строения языка C.

**Приложение А. Исходный код.**

/\*

\*Лаб раб №2

\*Lab\_2.c

\*Вайгачёв Андрей Олегович

\*03,10,16

\*

\* Для заданного целого n ≥ 0 вычислить:

\* (2n + 1)!! = 1·3·5·...· (2n + 1)

\*

\* То есть произведение нечетных чисел до числа (2\*n+1)

\*

\*/

#include <stdio.h>

#include <limits.h>

int main()

{

int n = -1;

printf("Enter n >= 0: ");

scanf("%d",&n); //Проверка правильности ввода

while (n < 0){

fflush(stdin); //Очищает буфер введённых значений

printf("\nEnter again: ");

scanf("%d",&n);

}

printf("You wrote %d\n",n);

unsigned long long int p = 1; // Использую тип переменных long long int

int i = 0;

short IsOverFlow = 0; // переменная отвечающая за переполнением

while ( (i < n) && !IsOverFlow ){

printf(" for i = %u : p = %llu \n",i,p);

i += 1;

p = p\*(2\*i + 1);

if (LONG\_LONG\_MAX/(2\*i + 1) <= p){ // проверка переполнения

IsOverFlow = 1;

}

}

printf("Last i = %u : p = %llu \n",i,p);

if (IsOverFlow){

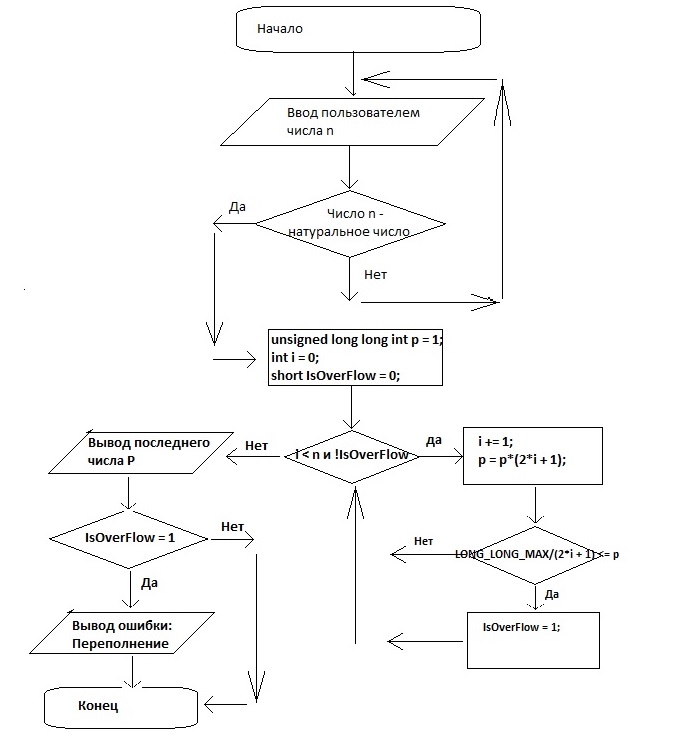
printf("Error: when i = %u is over flow.", i + 1 );

}

return 0;

}

**Приложение Б. Блок-схема.**

****