**Федеральное государственное автономное образовательное** **учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики»**

**Факультет компьютерных наук**

**Департамент программной инженерии**

**Дисциплина:** Архитектура Вычислительных Систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММЕ**

**Вариант 24**

**Группа БПИ191**

**Студент:** Удачин Данил Андреевич

**Преподаватель:** Легалов Александр Иванович

**Москва 2020**

**Содержание**

[**Постановка задачи и условие** 3](#_Toc55247193)

[**Решение задачи** 3](#_Toc55247194)

[**Текст программы** 3](#_Toc55247195)

[**Тестирование программы** 8](#_Toc55247196)

[**Список использованных источников** 9](#_Toc55247197)

# **Постановка задачи и условие**

Разработать программу определения количества обобщённых чисел Ферма , где n - неотрицательное целое число при a=2, b=3 от 1 до беззнакового двойного машинного слова.

# **Решение задачи**

Для начала теория. Число Ферма – это число вида , где . При этом по условию необходимо найти обобщённые числа Ферма. А это числа вида

Где a и b – заранее заданные константы. Соответственно, для нахождения данных чисел воспользуемся следующим алгоритмом:

У нас есть беззнаковое двойное машинное слово (uint в C#, или же unsigned int в других языках). Сохраним данное слово и будем использовать его как предел, вплоть и до которого будет осуществляться работа цикла. При переполнении и превышении данного числа – цикл будет прерываться.

Также, есть счётчик n, который увеличивается на 1 с каждой итерацией цикла. Счётчик служит для возведения чисел в степень и подсчёта итераций. В случае переполнения на экран выводится n - 1.

В данной программе этот алгоритм реализован на практике. Двойное машинное слово хранится в переменной maxValue, а значение n – в переменной mark.

# **Текст программы**

; Udachin Daniil, BSE191, Variant 24

; Develop a program for determining the number of

; generalized Fermat numbers a^2^n+b^2^n, where n is a

; non-negative integer for a=2, b=3 from 1 to an unsigned double machine word.

format PE console

include 'win32ax.inc'

entry start

;--------------------------------------------------------------------------------------------------

; Data

section '.data' data readable writable

strPrintNum db '%u', 0

strResult db 'Count of Fermat numbers in range from 1 to unsigned doubleword maxvalue: %d', 10, 0

strEnd db 'Press any key to exit.'

maxValue dd 4294967295 ; unsigned integer max value

mark dd 0 ; value for overflow control

NULL = 0

;--------------------------------------------------------------------------------------------------

; Main code

section '.code' code readable executable

start:

push [maxValue]

call searchFermatCount ; Search Fermat nums count

add esp, 4

push eax

push strResult

call [printf] ; Print result of calculations

add esp, 8

push strEnd

call [printf]

call [getch]

push NULL

call ExitProcess ; End

;------------------------------------------------

; Pow function

pow:

push ecx

push ebp

mov ebp, esp

sub esp, 4

i equ ebp-4

num equ ebp+16

power equ ebp+12

mov eax, 1

mov [i], dword 0

powLoop:

mov ecx, [i]

cmp ecx, [power]

jge finishPowLoop

imul eax, dword [num]

inc dword [i]

jmp powLoop

finishPowLoop:

mov esp, ebp

pop ebp

pop ecx

ret

;------------------------------------------------

; Fermat numbers counting function

searchFermatCount:

push ecx

push ebp

mov ebp, esp

sub esp, 8

i equ ebp-4

oldVal equ ebp-8

val equ ebp-12

maxVal equ ebp+12

mov [i], dword 0

mov [oldVal], dword 0

searchNum:

push dword [i]

call searchFermatNumber

add esp, 4

mov [val], eax

mov ecx, [val]

cmp ecx, [oldVal]

jb endSearch

cmp [mark], dword 1

je endSearch

mov [oldVal], ecx

inc dword [i]

jmp searchNum

endSearch:

mov eax, [i]

mov esp, ebp

pop ebp

pop ecx

ret

;------------------------------------------------

; Fermat number search function

searchFermatNumber:

push edx

push ecx

push ebp

mov ebp, esp

n equ ebp+16

push 2

push dword [n]

call pow

add esp, 8

mov edx, eax

cmp edx, 32

jl fermatPows

mov [mark], dword 1

fermatPows:

push 2

push eax

call pow

add esp, 8

mov ecx, eax

push 3

push edx

call pow

add esp, 8

add eax, ecx

mov esp, ebp

pop ebp

pop ecx

pop edx

ret

;--------------------------------------------------------------------------------------------------

; Libraries and imports

section '.idata' data readable import

library kernel, 'kernel32.dll',\

msvcrt, 'msvcrt.dll'

import kernel,\

ExitProcess, 'ExitProcess'

import msvcrt,\

printf, 'printf',\

getch, '\_getch'

# **Тестирование программы**



*Рисунок 1 – результат работы программы с выводом в консоль.*

Работа программы продемонстрирована на рис. 1. Входные данные не требуются.

# **Список использованных источников**

1. Flat Assembler Documentation // URL: https://flatassembler.net/docs.php (дата обращения 01.11.2020)
2. Wikipedia, Fermat number // URL: https://en.wikipedia.org/wiki/Fermat\_number