

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук  
Программная инженерия

**Микропроект**

Предмет: конструирование программного обеспечения

**Исполнитель:**

Студент группы БПИ194

В. А. Анненков

<b>Текст задания</b>	<b>3</b>
<b>Описание кода программы</b>	<b>4</b>
Объявленные переменные	4
Объявленные метки и процедуры	4
Описание алгоритма	4
Общая идея	4
Пошаговое описание	5
<b>Входные данные</b>	<b>6</b>
<b>Текст программы</b>	<b>7</b>
<b>Использованные источники</b>	<b>10</b>

## Текст задания

Разработать программу, определяющую число элементов последовательности Падована от 1 до беззнакового машинного слова.

# Описание кода программы

## Объявленные переменные

- **string\_extra\_output** — Строка, которая выводится для уточнение у пользователя, требуется ли выводить дополнительную информацию во время работы программы.
- **string\_yes\_no** — Строка, которая просит пользователя ввести вариант ответа в консоль.
- **string\_element** — Строка, которая отображает информацию о текущем посчитанном числе Падована.
- **string\_amount\_of\_elements** — Строка, которая выводит количество элементов в строке Падована.
- **string\_max\_element** — Строка, которая выводит максимальный элемент последовательности.
- **format\_number** — Переменная для ввода чисел в консоль.
- **is\_yes** — Переменная-имитация типа bool. Является ответом на вопрос “string\_extra\_output”.
- **counter** — Переменная для подсчёта количества корректных чисел Падована.
- **number1** — Третье с конца число Падована.
- **number2** — Второе с конца число Падована.
- **number3** — Первое с конца число Падована.

## Объявленные метки и процедуры

- **start** — Метка-входная точка программы.
- **input** — Процедура для ввода информации с клавиатуры.
- **process** — Процедура с главным циклом программы.
- **comparison** — Метка для перехода к условию о переполнении **number3**.

## Описание алгоритма

### Общая идея

Программа запускает цикл, который прерывается в случае переполнения числа **number3**. В переменную **counter** записывается общее количество чисел, которые мы можем вместить беззнаковом машинном слове. В переменной **number2** будет храниться максимальное число последовательности Падована, которое может уместиться в беззнаковое машинном слове.

В качестве размера беззнакового машинного слова определим 32 разряда.

## Пошаговое описание

1. Программа начинает исполнение с метки `start`.
2. Программа переходит в метку `input`.
3. Программа спрашивает пользователя, требуется ли выводить дополнительную информацию при решении задачи. (строка 38)
4. Программа считывает ответ пользователя. (строка 44)
5. Осуществляется переход в метку `process` – это основной цикл.
6. В самом начале работы основного цикла увеличивается значение **counter**. Его изначальное значение – 3, потому что первые три числа последовательности Падована мы точно знаем: 1, 1, 1.
7. Зануляется регистр `ecx`, затем в него записывается сумма **number1** и **number2**. (строка 55)
8. С помощью регистра `ebx` копируется значение **number2** в **number1**. (строка 59)
9. С помощью регистра `ebx` копируется значение **number3** в **number2**. (строка 62)
10. В **number3** перемещается результат суммы, который хранился в регистре `ecx`.
11. Сравнивается **is\_yes** с единицей. Если не равны, осуществляется переход на метку `comparison` (пункт ...). Если равны, то продолжается исполнение.
12. Выводится информация о текущем числе последовательности Падована. (строка 70)
13. Осуществляется сравнение **number2** и **number3**. Если **number3**  $\geq$  **number2**, то цикл начинает следующую итерацию. Иначе продолжается исполнение. (строка 76)
14. В конце цикла программа вычитает 1 из **counter**, потому что последнее посчитанное значение – не учитывается.
15. Программа выводит количество элементов и максимальный элемент.
16. Программа завершает исполнение.

## Входные данные

В программе требуется ввести единственное значение: ответ на вопрос “Требуется ли включить расширенный вывод?”.

Ответ требуется ввести после соответствующего вопроса. Возможны два варианта ответа:

“1” — аналогично ответу “да”,

любой другой — аналогично ответу “нет”.

# Текст программы

```
format PE console
entry start
include 'win32a.inc'

section '.data' data readable writable
    string_extra_output    db "Want to have extra output?", 10, 0
    string_yes_no          db "1 -- yes, else -- no: ", 0
    string_element         db "%d'th element is %d.", 10, 0
    string_amount_of_elements db 10, "The number of elements of the Padovan
sequence: %d.", 10, 0
    string_max_element     db "The max element is: %d.", 0

    format_number          db "%d", 0

    is_yes                 dd ?
    counter                dd 3

    number1                dd 1
    number2                dd 1
    number3                dd 1

    NULL = 0

;-----

section '.code' code readable executable
start:
    call input
    call process

    push NULL
    call [ExitProcess]

;-----

input:
    push string_extra_output
    call [printf]
    push string_yes_no
    call [printf]
    add esp, 8
```

```

        push is_yes      ; Input user answer.
        push format_number
        call [scanf]
        add esp, 8

        ret

; -----

process:      inc [counter]

        xor ecx, ecx.    ; Number1 and number2 sum.
        add ecx, [number1]
        add ecx, [number2]

        mov ebx, [number2] ; Copy number2 into number1.
        mov [number1], ebx

        mov ebx, [number3] ; Copy number3 into number2.
        mov [number2], ebx

        mov [number3], ecx ; Copy number1 and number2 sum to
number3.

        cmp [is_yes], 1
        jne comparison

        push [number3]
        push [counter]
        push string_element
        call [printf]
        add esp, 12

comparison:   mov ebx, [number2]
        mov ecx, [number3]
        cmp ecx, ebx
        jge process      ; Jump if 1'th more or equal than 2'th
argument.

                        ; So if 1'th less then overflow has been occurred.

        dec [counter]
        push [counter]
        push string_amount_of_elements
        call [printf]
        add esp, 8

        push [number2]

```



```
push string_max_element
call [printf]
add esp, 8

call [getch]

ret
```

```
;-----
```

```
section '.idata' import data readable
library      kernel,  "kernel32.dll",\
             msvcrt,  "msvcrt.dll"

import      kernel,\
             ExitProcess, "ExitProcess"

import      msvcrt,\
             printf,    "printf",\
             getch,     "_getch",\
             scanf,     "scanf"
```

## Использованные источники

1. [https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C\\_%D0%9F%D0%B0%D0%B4%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%9F%D0%B0%D0%B4%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0)
2. <http://av-assembler.ru/>