**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №4

дисциплина: Архитектура вычислительных систем

тема: «Команды передачи управления»

Выполнил: ст. группы ПВ-31

Донцов А.А.

Проверил: Осипов О.В.

Белгород 2019

Цель работы: изучение команд перехода для организации циклов и ветвлений, получение навыков создания процедур с аргументами.

**Вариант 1**

****

.386

.model flat, stdcall

option casemap: none

include c:\masm32\include\kernel32.inc

include c:\masm32\include\msvcrt.inc

includelib c:\masm32\lib\kernel32.lib

includelib c:\masm32\lib\msvcrt.lib

.data

x dw -10

y dw 4

z dw 2

format db "a = %d", 0

.code

start:

push EBX ; запоминаем EBX

push ECX ; запоминаем ECX

MOV AX, x ; AX = x

CWDE ; расширяем AX до EAX

MOV EBX, EAX ; EBX = x

MOV AX, y ; AX = y

CWDE ; расширяем AX до EAX

MOV ECX, EAX ; ECX = y

CMP EBX, 10 ;сравниваем x и 10

JL j1 ; x < 10

IMUL EBX ; EAX = x\*y

IMUL EAX ; EAX\*EAX

JMP j\_out ; прыжок на выход

j1:

CMP ECX, 2 ; сравниваем y и 2

JL j2 ; y < 2

MOV AX, z ; AX = z

CWDE

MOV ECX, EAX ; ECX = z

MOV EAX, EBX ; EAX = x

CDQ

IDIV ECX ; x/z

SUB EAX, 6 ; x/z - 6

JMP j\_out

j2:

IMUL EBX ; EAX = x\*y

ADD EAX,4

j\_out:

pop ECX

pop EBX

push EAX

push offset format

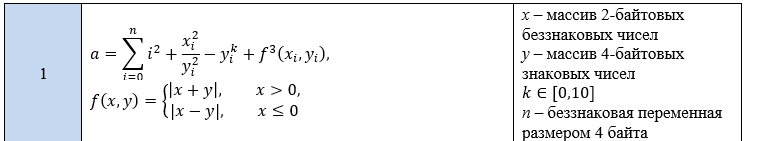
call crt\_printf

call crt\_\_getch ; Задержка ввода

push 0

call ExitProcess

end start



.386

.model flat, stdcall

option casemap: none

include c:\masm32\include\kernel32.inc

include c:\masm32\include\msvcrt.inc

includelib c:\masm32\lib\kernel32.lib

includelib c:\masm32\lib\msvcrt.lib

.data

x dw -1, -2, 1, 15, 5, -50, 170, 8, 45, 10

y dd 1, -1, 7, -15, 30, 20, 35, 40, 10, 10

k dd 4

n dd 2

format db "a = %d", 0

.code

; Функция от двух аргументов

f proc

push EBX

MOV EAX, [ESP+8] ; EAX = y

MOV EBX, EAX ; EBX = y

MOV EAX, [ESP+4] ; EAX = x

CMP EAX, 0

JLE f1

ADD EAX, EBX ; EAX = x+y

CMP EAX, 0

JL fmod

JMP fexit

f1:

SUB EAX, EBX ; EAX = x - y

CMP EAX, 0

JL fmod

JMP fexit

fmod:

MOV EBP, -1

IMUL EBP

JMP fexit

fexit:

pop EBX

ret 8

f endp

start:

XOR EBX, EBX ; В EBX будет накапливаться сумма

XOR ESI, ESI ; ESI - индекс i элементов в массивах

XOR ECX, ECX ; ECX - счётчик итераций

MOV ECX, n ; ECX = n

j1:

MOV EAX, ESI ; EAX = i

MUL EAX ; EAX = i\*i

ADD EBX, EAX ; EBX = EBX + i\*i

XOR EAX,EAX ; EAX = 0

MOV AX, x[2\*ESI] ; AX = x[i]

MUL EAX ; EAX = x[i] \* x[i]

CDQ ; расширение EDX

IDIV y[4\*ESi] ; EAX = EAX / y[i]

CDQ ; расширение EDX

IDIV y[4\*ESi] ; EAX = EAX / y[i]

ADD EBX, EAX ; EBX += EAX

MOV EAX, y[4\*ESI] ; EAX = y[i]

MOV EBP, y[4\*ESI] ; EBP = y[i]

MOV EDI, k ; EDI = k

CMP EDI, 0

JE j3

j2:

MUL EBP

DEC EDI

CMP EDI, 1

JG j2

JMP j4

j3:

MOV EAX, 1

j4:

push y[4\*ESI]

push x[2\*ESI]

call f ; EAX = f(x[i],y[i])

MOV EBP, EAX ;

MUL EBP

MUL EBP

ADD EBX, EAX

INC ESI ; ESI = ESI + 1

LOOP j1 ; ECX = ECX - 1. Переход в начало цикла, если ECX ? 0

MOV EAX, EBX ; Поместить результат в EAX

push EAX

push offset format

call crt\_printf ; Вывод результата на экран

call crt\_\_getch ; Задержка ввода

push 0

call ExitProcess ; Выход из программы

end start