**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

по теме Команды передачи управления

Выполнил: ст. группы ВТ-31  
Новожен Н.В

Проверил: Осипов.О.В

**Белгород 2019**

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

**Задания для выполнения к работе**

1. Написать программу для вычисления значения арифметического выражения, используя команды условного и безусловного перехода согласно варианту задания. Подобрать набор тестовых данных (не менее 3). При выполнении операций с числами, преобразовывать их к 4-байтовым числам со знаком. Результат вывести на экран.
2. Написать программу для вычисления значения арифметического выражения, содержащего функцию. Вычисление функции организовать в виде отдельной подпрограммы по всем правилам, описанным выше. Для обработки массивов использовать команды для работы с циклами и команды условного перехода. Подобрать набор тестовых данных (не менее 3). Результат вывести на экран.

**ЗАДАНИЕ 1**

|  |  |  |
| --- | --- | --- |
| 11 |  | *x* – знаковое 4-байтовое  *y* – знаковое 2-байтовое  *z* – беззнаковое однобайтовое |

.386

.MODEL FLAT, STDCALL

OPTION CASEMAP: NONE

include E:\masm32\include\windows.inc

include E:\masm32\include\kernel32.inc

include E:\masm32\include\user32.inc

includelib E:\masm32\lib\user32.lib

includelib E:\masm32\lib\kernel32.lib

.DATA

x dd -20

y dw 44

z db 44

s db 8 dup(?) ; Для результата резервируется на один байт больше

res dd ?

.CODE

START:

CMP x,-15 ;**x<-15**

JL i1 ;**IF TRUE**

JMP j3

i1:;x<-15

CMP y,-10

JGE i2;**IF операнд\_1 ≥ операнд\_2**

JMP j2

i2:

CMP y,20

JLE j1; **IF операнд\_1 ≤ операнд\_2**

JMP j2

JMP j1

j1:

MOVSX EAX ,y

MOVSX EBX ,y

CDQ

IMUL EAX ;**EAX=Y\*Y**

IMUL EBX,EBX,14 ;**EBX=Y\*14**

IMUL EBX ;**EAX=Y\*14\*Y\*Y**

IMUL x ;**EAX=Y\*14\*Y\*Y\*X**

MOV dword ptr s[0],EAX

MOV dword ptr s[4],EDX

JMP j\_end

JMP j2

j2:

MOVZX EBX,z ;**EBX=Z**

ADD EBX,x ;**EBX=X+Z**

MOV EAX,1 ;**EAX=1**

CDQ

MOVSX ECX,y ;**ECX=Y**

;;CMP ECX,0

;;JE null ;IF Y==0

IDIV ECX ;**EAX=1/Y**

ADD EAX,EBX ;**EAX=1/Y +X+Z**

MOV dword ptr s[0],EAX

XOR EAX,EAX

ADC EAX,0

MOV dword ptr s[4],EAX

JMP j\_end

JMP j3

j3:

MOVZX EAX,z ;**EAX=Z**

MOVZX EBX,z ;**EBX=Z**

CDQ

IMUL EAX ;**EAX=Z\*Z**

IMUL EBX ;**EAX=Z\*Z\*Z**

MOV EBX,3

IMUL EBX ;**EAX=Z\*Z\*Z\*3**

IMUL x ;**EAX=Z\*Z\*Z\*3\*X**

MOV dword ptr s[0],EAX

MOV dword ptr s[4],EDX

JMP j\_end

;;null:

j\_end:

push NULL

call ExitProcess

END START

Набор тестовых данных

|  |  |  |  |
| --- | --- | --- | --- |
| x | y | z | S |
| -20 | 5 | 77 | FFFF7748 |
| -20 | 44 | 44 | 18 |
| 1 | 2 | 44 | 3E640 |

**Вариант задания задачи №2**

|  |  |  |
| --- | --- | --- |
| 11 |  | *x*, *y* — массивы 2-байтовых знаковых чисел  *n* – беззнаковая переменная размером 1 байт |

.386

.MODEL FLAT, STDCALL

OPTION CASEMAP: NONE

include E:\masm32\include\windows.inc

include E:\masm32\include\kernel32.inc

include E:\masm32\include\user32.inc

includelib E:\masm32\lib\user32.lib

includelib E:\masm32\lib\kernel32.lib

.DATA

k db 1

n db 2

x dw 00001h,00000h,00000h, 00000h ; 8 bite

y dw 00001h,00000h,00000h,00000h ; 8 bite

s dd 2 dup(?)

cons\_2 dd 2

cons\_8 dd 8

.CODE

t proc

MOV EAX,[ESP+4];EAX

MOV EBX,[ESP+8];EBX

CMP EAX,5

JA j1;операнд\_1 > операнд\_2

INC EBX

j1:

MOV ECX,EAX

MOV EAX,EBX

CDQ

lop:

IMUL EBX

LOOPE lop

RET 8

t endp

cln proc

XOR EAX,EAX

XOR EBX,EBX

XOR EDI,EDI

XOR EBP,EBP

XOR EDX,EDX

RET 0

cln endp

START:

MOVZX ECX,n;СЧЁТЧИК

XOR ESI,ESI

CALL cln

j8:

MOVSX EBX,x[ESI]

MOVSX EAX,y[ESI]

CDQ

IMUL EAX

IMUL EBX

MOV EDI,2

IMUL EDI

XOR EDI,EDI

ADD EAX,EBX ;EAX=Xi+2\*Xi\*Yi\*Yi

ADC EDX ,0

MOV EDI,EAX; SAVE RES IN REGISTRS

MOV EBP,EDX

XOR EAX,EAX

XOR EDX,EDX

MOVSX EAX,y[ESI]

CDQ

IMUL EAX

IMUL y[ESI]

IMUL cons\_8

ADD EDI,EAX

ADC EBP,EDX ;=Xi+2\*Xi\*Yi\*Yi +8\*Yi\*Yi\*Yi

;

PUSH EBX

PUSH ECX

XOR EBX,EBX

XOR ECX,ECX

JMP j6

j7:

XOR EAX,ECX

XOR ECX,EAX

XOR EAX,ECX

JMP j8

j6:

MOVSX EBX,x[ESI]

MOVZX ECX,k

CALL t;RETURN RES EDX:EAX

POP ECX

POP EBX

ADD EDI,EAX

ADC EBP,EDX

MOV EAX,s[0]

MOV EDX,s[4]

ADD EAX,EDI

ADC EDX,EBP

MOV s[0],EAX; SAVE RES

MOV s[4],EDX

;CLEAN

CALL cln

MOVSX AX,n

DEC AX

MOV n,AL

INC ECX

INC ECX

XOR EAX,ECX

XOR ECX,EAX

XOR EAX,ECX

LOOP j7

push NULL

call ExitProcess

END START

|  |  |  |
| --- | --- | --- |
| K | N | RES |
| 1 | 2 | F |

Вывод: мы изучениели команды перехода для организации циклов и ветвлений, получили навыки создания процедур с аргументами