Ministerul Educaţiei și Cercetării

al Republicii Moldova   
Universitatea Tehnică a Moldovei

Faculatea Calculatoare, Infromatică și Microelectronică   
  
  
  
  
  
  
  
RAPORT

# LUCRARE DE LABORATOR NR. 3

**la Arhitectura Calculatoarelor**

**Tema: PROGRAME LINIARE**

A efectuat: Popa Cătălin

st. gr. TI-211

A verificat: asist. univ. Secrieru Adrian

UTM, Chișinău 2022

**Tema:**

Elaborare programelor în limbajul de asamblare MASM în Visual Studio.

**Scopul lucrării:**

Lucrarea prezinta instructiunile pentru transferuri de date, instructiuni in aritmetica binara si in aritmetica BCD.

**1. Enunțul problemelor de rezolvat**

Varianta 11

**2. Exemplele de programe comentate**

**3. Listingul programului comentat conform variantei si codul sursa**

1. Varianta cu introducerea datelor de la tastatura si afisarea rezultatelor pe ecran.

INCLUDE Irvine32.inc ; includem fisierul de antet

.data ; segmentul de date

mes1 byte "Introduceti valoarea X:", 0 ; string pentru mesajul de introducere a valorii X

mes2 byte "Introduceti valoarea Y:", 0 ; string pentru mesajul de introducere a valorii Y

mes3 byte "Rezultatul este:", 0 ; string pentru afisarea rezultatului

vrx dword 0 ; variabila pentru valoarea X, initializata cu 0

vry dword 0 ; variabila pentru valoarea Y, initializata cu 0

rez dword 0 ; variabila pentru stocarea rezultatului, initializata cu 0

.code ; segmentul de cod

main PROC ; incepem procedura principala

mov edx, offset mes1 ; mutam adresa mesajului 1 in registrul edx

call WriteString ; afisam mesajul 1

call ReadInt ; citim o valoare intreaga de la tastatura

mov vrx, eax ; mutam valoarea citita in variabila vrx

mov edx, offset mes2 ; mutam adresa mesajului 2 in registrul edx

call WriteString ; afisam mesajul 2

call ReadInt ; citim o alta valoare intreaga de la tastatura

mov vry, eax ; mutam valoarea citita in variabila vry

mov eax, vrx ; Salvam valoarea lui X in registru EAX

and eax, 1 ; Operatie bit la bit intre X si 1

cmp eax, 0 ; Comparam rezultatul cu 0

jz par ; Daca ZF este setat, atunci X este par

jmp impar ; Daca ZF nu este setat, atunci X este impar

par:

mov eax, vrx ; mutam valoarea lui vrx in eax

mov ebx, vry ; mutam valoarea lui vry in ebx

sub ebx, eax ; scadem valoarea din eax din valoarea lui ebx

mov eax, ebx ; mutam valoarea lui ebx in eax

mov bx, 2 ; mutam valoarea 2 in partea inferioara a registrului bx

cwd ; extindem semnul din dx in ax

mul bx ; inmultim valoarea din eax cu 2

sub eax, 121 ; scadem din valoarea din eax 121

mov rez, eax ; mutam valoarea din eax in variabila rez

jmp sfarsit ; sarim la eticheta sfarsit

impar:

mov eax, vrx ; mutam valoarea lui vrx in eax

shr eax, 2 ; impartim valoarea lui vrx la 4

mov ebx, vry ; mutam valoarea lui vry in ebx

shl ebx, 1 ; inmultim valoarea lui ebx cu 2

add eax, ebx ; rezultatul adunaii valorii din ebx cu eax

sfarsit:

mov edx, offset mes3

call WriteString

call WriteInt

call Crlf

exit

main ENDP

END main

**Fisierul listing**

INCLUDE Irvine32.inc ; includem fisierul de antet

C ; Include file for Irvine32.lib (Irvine32.inc)

C

C ;OPTION CASEMAP:NONE ; optional: make identifiers case-sensitive

C

C INCLUDE SmallWin.inc ; MS-Windows prototypes, structures, and constants

C .NOLIST

C .LIST

C

C INCLUDE VirtualKeys.inc

C ; VirtualKeys.inc

C .NOLIST

C .LIST

C

C

C .NOLIST

C .LIST

C

00000000 .data ; segmentul de date

00000000 49 6E 74 72 6F mes1 byte "Introduceti valoarea X:", 0 ; string pentru mesajul de introducere a valorii X

64 75 63 65 74

69 20 76 61 6C

6F 61 72 65 61

20 58 3A 00

00000018 49 6E 74 72 6F mes2 byte "Introduceti valoarea Y:", 0 ; string pentru mesajul de introducere a valorii Y

64 75 63 65 74

69 20 76 61 6C

6F 61 72 65 61

20 59 3A 00

00000030 52 65 7A 75 6C mes3 byte "Rezultatul este:", 0 ; string pentru afisarea rezultatului

74 61 74 75 6C

20 65 73 74 65

3A 00

00000041 00000000 vrx dword 0 ; variabila pentru valoarea X, initializata cu 0

00000045 00000000 vry dword 0 ; variabila pentru valoarea Y, initializata cu 0

00000049 00000000 rez dword 0 ; variabila pentru stocarea rezultatului, initializata cu 0

00000000 .code ; segmentul de cod

00000000 main PROC ; incepem procedura principala

00000000 BA 00000000 R mov edx, offset mes1 ; mutam adresa mesajului 1 in registrul edx

00000005 E8 00000000 E call WriteString ; afisam mesajul 1

0000000A E8 00000000 E call ReadInt ; citim o valoare intreaga de la tastatura

0000000F A3 00000041 R mov vrx, eax ; mutam valoarea citita in variabila vrx

00000014 BA 00000018 R mov edx, offset mes2 ; mutam adresa mesajului 2 in registrul edx

00000019 E8 00000000 E call WriteString ; afisam mesajul 2

0000001E E8 00000000 E call ReadInt ; citim o alta valoare intreaga de la tastatura

00000023 A3 00000045 R mov vry, eax ; mutam valoarea citita in variabila vry

00000028 A1 00000041 R mov eax, vrx ; Salvam valoarea lui X in registru EAX

0000002D 83 E0 01 and eax, 1 ; Operatie bit la bit intre X si 1

00000030 83 F8 00 cmp eax, 0 ; Comparam rezultatul cu 0

00000033 74 02 jz par ; Daca ZF este setat, atunci X este par

00000035 EB 22 jmp impar ; Daca ZF nu este setat, atunci X este impar

00000037 par:

00000037 A1 00000041 R mov eax, vrx ; mutam valoarea lui vrx in eax

0000003C 8B 1D 00000045 R mov ebx, vry ; mutam valoarea lui vry in ebx

00000042 2B D8 sub ebx, eax ; scadem valoarea din eax din valoarea lui ebx

00000044 8B C3 mov eax, ebx ; mutam valoarea lui ebx in eax

00000046 66| BB 0002 mov bx, 2 ; mutam valoarea 2 in partea inferioara a registrului bx

0000004A 66| 99 cwd ; extindem semnul din dx in ax

0000004C 66| F7 E3 mul bx ; inmultim valoarea din eax cu 2

0000004F 83 E8 79 sub eax, 121 ; scadem din valoarea din eax 121

00000052 A3 00000049 R mov rez, eax ; mutam valoarea din eax in variabila rez

00000057 EB 12 jmp sfarsit ; sarim la eticheta sfarsit

00000059 impar:

00000059 A1 00000041 R mov eax, vrx ; mutam valoarea lui vrx in eax

0000005E C1 E8 02 shr eax, 2 ; impartim valoarea lui vrx la 4

00000061 8B 1D 00000045 R mov ebx, vry ; mutam valoarea lui vry in ebx

00000067 D1 E3 shl ebx, 1 ; inmultim valoarea lui ebx cu 2

00000069 03 C3 add eax, ebx ; rezultatul adunaii valorii din ebx cu eax

0000006B sfarsit:

0000006B BA 00000030 R mov edx, offset mes3

00000070 E8 00000000 E call WriteString

00000075 E8 00000000 E call WriteInt

0000007A E8 00000000 E call Crlf

exit

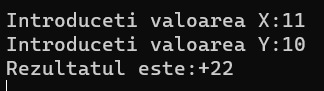
0000007F 6A 00 \* push +000000000h

00000081 E8 00000000 E \* call ExitProcess

00000086 main ENDP

END main

**Rezultatul**



2. Varianta cu generarea datelor de intrare , utilizand procedurile Rnadom32, RandomRange:

INCLUDE Irvine32.inc ; includem fisierul de antet

.data ; segmentul de date

mes1 byte "Introduceti valoarea X:", 0 ; string pentru mesajul de introducere a valorii X

mes2 byte "Introduceti valoarea Y:", 0 ; string pentru mesajul de introducere a valorii Y

mes3 byte "Rezultatul este:", 0 ; string pentru afisarea rezultatului

vrx dword 0 ; variabila pentru valoarea X, initializata cu 0

vry dword 0 ; variabila pentru valoarea Y, initializata cu 0

rez dword 0 ; variabila pentru stocarea rezultatului, initializata cu 0

.code ; segmentul de cod

main PROC ; incepem procedura principala

call Randomize

mov edx,offset mes1

call WriteString;

call Random32;

call WriteInt

call Crlf

mov vrx,eax;

mov edx,offset mes2

call WriteString;

call Random32;

call WriteInt

call Crlf

mov vry, eax ; mutam valoarea citita in variabila vry

mov eax, vrx ; Salvam valoarea lui X in registru EAX

and eax, 1 ; Operatie bit la bit intre X si 1

cmp eax, 0 ; Comparam rezultatul cu 0

jz par ; Daca ZF este setat, atunci X este par

jmp impar ; Daca ZF nu este setat, atunci X este impar

par:

mov eax, vrx ; mutam valoarea lui vrx in eax

mov ebx, vry ; mutam valoarea lui vry in ebx

sub ebx, eax ; scadem valoarea din eax din valoarea lui ebx

mov eax, ebx ; mutam valoarea lui ebx in eax

mov bx, 2 ; mutam valoarea 2 in partea inferioara a registrului bx

cwd ; extindem semnul din dx in ax

mul bx ; inmultim valoarea din eax cu 2

sub eax, 121 ; scadem din valoarea din eax 121

mov rez, eax ; mutam valoarea din eax in variabila rez

jmp sfarsit ; sarim la eticheta sfarsit

impar:

mov eax, vrx ; mutam valoarea lui vrx in eax

shr eax, 2 ; impartim valoarea lui vrx la 4

mov ebx, vry ; mutam valoarea lui vry in ebx

shl ebx, 1 ; inmultim valoarea lui ebx cu 2

add eax, ebx ; rezultatul adunaii valorii din ebx cu eax

sfarsit:

mov edx, offset mes3

call WriteString

call WriteInt

call Crlf

exit

main ENDP

END main

**Fisierul listing**

INCLUDE Irvine32.inc ; includem fisierul de antet

C ; Include file for Irvine32.lib (Irvine32.inc)

C

C ;OPTION CASEMAP:NONE ; optional: make identifiers case-sensitive

C

C INCLUDE SmallWin.inc ; MS-Windows prototypes, structures, and constants

C .NOLIST

C .LIST

C

C INCLUDE VirtualKeys.inc

C ; VirtualKeys.inc

C .NOLIST

C .LIST

C

C

C .NOLIST

C .LIST

C

00000000 .data ; segmentul de date

00000000 49 6E 74 72 6F mes1 byte "Introduceti valoarea X:", 0 ; string pentru mesajul de introducere a valorii X

64 75 63 65 74

69 20 76 61 6C

6F 61 72 65 61

20 58 3A 00

00000018 49 6E 74 72 6F mes2 byte "Introduceti valoarea Y:", 0 ; string pentru mesajul de introducere a valorii Y

64 75 63 65 74

69 20 76 61 6C

6F 61 72 65 61

20 59 3A 00

00000030 52 65 7A 75 6C mes3 byte "Rezultatul este:", 0 ; string pentru afisarea rezultatului

74 61 74 75 6C

20 65 73 74 65

3A 00

00000041 00000000 vrx dword 0 ; variabila pentru valoarea X, initializata cu 0

00000045 00000000 vry dword 0 ; variabila pentru valoarea Y, initializata cu 0

00000049 00000000 rez dword 0 ; variabila pentru stocarea rezultatului, initializata cu 0

00000000 .code ; segmentul de cod

00000000 main PROC ; incepem procedura principala

00000000 E8 00000000 E call Randomize

00000005 BA 00000000 R mov edx,offset mes1

0000000A E8 00000000 E call WriteString;

0000000F E8 00000000 E call Random32;

00000014 E8 00000000 E call WriteInt

00000019 E8 00000000 E call Crlf

0000001E A3 00000041 R mov vrx,eax;

00000023 BA 00000018 R mov edx,offset mes2

00000028 E8 00000000 E call WriteString;

0000002D E8 00000000 E call Random32;

00000032 E8 00000000 E call WriteInt

00000037 E8 00000000 E call Crlf

0000003C A3 00000045 R mov vry, eax ; mutam valoarea citita in variabila vry

00000041 A1 00000041 R mov eax, vrx ; Salvam valoarea lui X in registru EAX

00000046 83 E0 01 and eax, 1 ; Operatie bit la bit intre X si 1

00000049 83 F8 00 cmp eax, 0 ; Comparam rezultatul cu 0

0000004C 74 02 jz par ; Daca ZF este setat, atunci X este par

0000004E EB 22 jmp impar ; Daca ZF nu este setat, atunci X este impar

00000050 par:

00000050 A1 00000041 R mov eax, vrx ; mutam valoarea lui vrx in eax

00000055 8B 1D 00000045 R mov ebx, vry ; mutam valoarea lui vry in ebx

0000005B 2B D8 sub ebx, eax ; scadem valoarea din eax din valoarea lui ebx

0000005D 8B C3 mov eax, ebx ; mutam valoarea lui ebx in eax

0000005F 66| BB 0002 mov bx, 2 ; mutam valoarea 2 in partea inferioara a registrului bx

00000063 66| 99 cwd ; extindem semnul din dx in ax

00000065 66| F7 E3 mul bx ; inmultim valoarea din eax cu 2

00000068 83 E8 79 sub eax, 121 ; scadem din valoarea din eax 121

0000006B A3 00000049 R mov rez, eax ; mutam valoarea din eax in variabila rez

00000070 EB 12 jmp sfarsit ; sarim la eticheta sfarsit

00000072 impar:

00000072 A1 00000041 R mov eax, vrx ; mutam valoarea lui vrx in eax

00000077 C1 E8 02 shr eax, 2 ; impartim valoarea lui vrx la 4

0000007A 8B 1D 00000045 R mov ebx, vry ; mutam valoarea lui vry in ebx

00000080 D1 E3 shl ebx, 1 ; inmultim valoarea lui ebx cu 2

00000082 03 C3 add eax, ebx ; rezultatul adunaii valorii din ebx cu eax

00000084 sfarsit:

00000084 BA 00000030 R mov edx, offset mes3

00000089 E8 00000000 E call WriteString

0000008E E8 00000000 E call WriteInt

00000093 E8 00000000 E call Crlf

exit

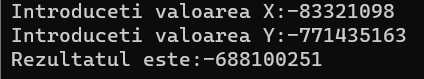
00000098 6A 00 \* push +000000000h

0000009A E8 00000000 E \* call ExitProcess

0000009F main ENDP

END main

**Rezultatul**



**4. Concluzii**