**Ministerul Educației, Culturii și Cercetării**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Tema: Programare Liniare**

Lucrare de laborator nr.4

la disciplina Arhitectura Calculatoarelor

**Student gr. TI-173: Heghea Nicolae**

**Conducător: Colesnic Victor**

# Scopul lucrării

Lucrarea urmărește deprinderea studenților cu proiectarea și implementarea programelor cu subprograme și macroinstrucțiuni, cu ramificații și bucle in limbaj de asamblare. Se prezintă câteva exemple tipice de astfel de programe, incluzând câteva metode elementare de sortare şi căutare. De asemenea se prezintă câteva implementări de operații aritmetice care necesită utilizarea unor structuri de control de tip ramificare şi buclare. Se vor utiliza instrucțiunile de comparare, salt şi buclare. Se prezintă de asemenea şi câteva exemple de utilizare a instrucțiunilor logice.

# Condiția

Conform variantei elaborați 2 variante de program :

1. cu introducerea datelor de la tastatură și afișarea rezultatelor pe ecran.
2. cu generarea datelor de intrare, utilizând procedurile Random32, RandomRange și afișarea rezultatelor pe ecran.

Varianta 9 :

# Realizarea Condiției 4

Codul programului.

|  |
| --- |
| INCLUDE Irvine32.inc  .DATA  ;sir DB 'Elaborati un program care transforma sirul de caractere'  ; DB 'incepand de la adresa [string] in rasturnatul sau,'  ; DB 'numarul de elemente al sirului fiind cunoscut.',0  sir db '123456@890abcdef',0  lens EQU sizeof sir  sir2 DB lens dup(0)  pos DB 5  .CODE  CopyStr macro src, dest  local L1  lea esi, src  lea edi, dest  L1:  mov eax,[esi]  mov [edi], eax  inc esi  inc edi  cmp byte ptr [esi], 0 ; verifica terminatia de 0  jg L1 ; repeta daca nui null    endm  AfisStr macro x  lea edx, x  call writestring  endm  AfisStrLn macro x  Afisstr x  call crlf  endm  AfisInt macro x  mov eax, x  call writeint  call crlf  endm  ReverseStr1 macro src, x, n ; scr - sursa  ; x - pozitia de incepere  ; n - nr. de caractere  local L1  lea esi, src + x  mov ecx, n-1  L1:  mov al, byte ptr [esi]  ;mov dl, byte ptr [esi + ecx]  ;mov byte ptr [esi], dl  xchg al, byte ptr [esi + ecx]  mov byte ptr [esi], al  inc esi  sub ecx, 2  cmp ecx, 0  jg L1  endm  ReverseStr2 macro src ; scr - sursa  local L0, L1, L2  lea esi, src  mov ecx, 0  L0:  inc esi  cmp byte ptr [esi], 40h ; '@' symbol  je L1  inc ecx  jne L0  L1:  lea esi, src  L2:  mov al, byte ptr [esi]  xchg al, byte ptr [esi + ecx]  mov byte ptr [esi], al  inc esi  sub ecx, 2  cmp ecx, 0  jg L2  endm  main proc  afisstrln sir  CopyStr sir, sir2  reverseStr1 sir2, 0, 3  call crlf  afisstrln sir2  CopyStr sir, sir2  reverseStr2 sir2  call crlf  afisstrln sir2  exit  main ENDP  END main |

Listingul programului.

|  |
| --- |
| Microsoft (R) Macro Assembler Version 14.16.27026.1 04/15/19 19:19:03  programator.asm Page 1 - 1  INCLUDE Irvine32.inc  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  ;sir DB 'Elaborati un program care transforma sirul de caractere'  ; DB 'incepand de la adresa [string] in rasturnatul sau,'  ; DB 'numarul de elemente al sirului fiind cunoscut.',0  00000000 31 32 33 34 35 sir db '123456@890abcdef',0  36 40 38 39 30  61 62 63 64 65  66 00  = 00000011 lens EQU sizeof sir  00000011 sir2 DB lens dup(0)  00000022 05 pos DB 5  00000000 .CODE  CopyStr macro src, dest  local L1  lea esi, src  lea edi, dest  L1:  mov eax,[esi]  mov [edi], eax  inc esi  inc edi  cmp byte ptr [esi], 0 ; verifica terminatia de 0  jg L1 ; repeta daca nui null    endm  AfisStr macro x  lea edx, x  call writestring  endm  AfisStrLn macro x  afisstr x  call crlf  endm  AfisInt macro x  mov eax, x  call writeint  call crlf  endm  ReverseStr1 macro src, x, n ; scr - sursa  ; x - pozitia de incepere  ; n - nr. de caractere  local L1  lea esi, src + x  mov ecx, n-1  L1:  mov al, byte ptr [esi]  ;mov dl, byte ptr [esi + ecx]  ;mov byte ptr [esi], dl  xchg al, byte ptr [esi + ecx]  mov byte ptr [esi], al  inc esi  sub ecx, 2  cmp ecx, 0  jg L1  endm  ReverseStr2 macro src ; scr - sursa  local L0, L1, L2  lea esi, src  mov ecx, 0  L0:  inc esi  cmp byte ptr [esi], 40h ; '@' symbol  je L1  inc ecx  jne L0  L1:  lea esi, src  L2:  mov al, byte ptr [esi]  xchg al, byte ptr [esi + ecx]  mov byte ptr [esi], al  inc esi  sub ecx, 2  cmp ecx, 0  jg L2  endm  00000000 main proc  afisstrln sir  00000000 8D 15 00000000 R 2 lea edx, sir  00000006 E8 00000000 E 2 call writestring  0000000B E8 00000000 E 1 call crlf  CopyStr sir, sir2  00000010 8D 35 00000000 R 1 lea esi, sir  00000016 8D 3D 00000011 R 1 lea edi, sir2  0000001C 1 ??0000:  0000001C 8B 06 1 mov eax,[esi]  0000001E 89 07 1 mov [edi], eax  00000020 46 1 inc esi  00000021 47 1 inc edi  00000022 80 3E 00 1 cmp byte ptr [esi], 0 ; verifica terminatia de 0  00000025 7F F5 1 jg ??0000 ; repeta daca nui null  reverseStr1 sir2, 0, 3  00000027 8D 35 00000011 R 1 lea esi, sir2 + 0  0000002D B9 00000002 1 mov ecx, 3-1  00000032 1 ??0001:  00000032 8A 06 1 mov al, byte ptr [esi]  00000034 86 04 0E 1 xchg al, byte ptr [esi + ecx]  00000037 88 06 1 mov byte ptr [esi], al  00000039 46 1 inc esi  0000003A 83 E9 02 1 sub ecx, 2  0000003D 83 F9 00 1 cmp ecx, 0  00000040 7F F0 1 jg ??0001  00000042 E8 00000000 E call crlf  afisstrln sir2  00000047 8D 15 00000011 R 2 lea edx, sir2  0000004D E8 00000000 E 2 call writestring  00000052 E8 00000000 E 1 call crlf  CopyStr sir, sir2  00000057 8D 35 00000000 R 1 lea esi, sir  0000005D 8D 3D 00000011 R 1 lea edi, sir2  00000063 1 ??0002:  00000063 8B 06 1 mov eax,[esi]  00000065 89 07 1 mov [edi], eax  00000067 46 1 inc esi  00000068 47 1 inc edi  00000069 80 3E 00 1 cmp byte ptr [esi], 0 ; verifica terminatia de 0  0000006C 7F F5 1 jg ??0002 ; repeta daca nui null  reverseStr2 sir2  0000006E 8D 35 00000011 R 1 lea esi, sir2  00000074 B9 00000000 1 mov ecx, 0  00000079 1 ??0003:  00000079 46 1 inc esi  0000007A 80 3E 40 1 cmp byte ptr [esi], 40h ; '@' symbol  0000007D 74 03 1 je ??0004  0000007F 41 1 inc ecx  00000080 75 F7 1 jne ??0003  00000082 1 ??0004:  00000082 8D 35 00000011 R 1 lea esi, sir2  00000088 1 ??0005:  00000088 8A 06 1 mov al, byte ptr [esi]  0000008A 86 04 0E 1 xchg al, byte ptr [esi + ecx]  0000008D 88 06 1 mov byte ptr [esi], al  0000008F 46 1 inc esi  00000090 83 E9 02 1 sub ecx, 2  00000093 83 F9 00 1 cmp ecx, 0  00000096 7F F0 1 jg ??0005  00000098 E8 00000000 E call crlf  afisstrln sir2  0000009D 8D 15 00000011 R 2 lea edx, sir2  000000A3 E8 00000000 E 2 call writestring  000000A8 E8 00000000 E 1 call crlf  exit  000000AD 6A 00 \* push +000000000h  000000AF E8 00000000 E \* call ExitProcess  000000B4 main ENDP  END main |

# Realizarea Condiției 5 “Inpacheteaza4To3”

Codul programului pe 32 biți.

|  |
| --- |
| INCLUDE Irvine32.inc  .DATA  arr DW 0123h, 0456h, 0789h, 0abch  lenArr EQU lengthof arr  arr2 DW lenArr dup(0)  a DB 0  b DB 0  .CODE  AfisStr macro x  lea edx, x  call writestring  endm  AfisStrLn macro x  afisstr x  call crlf  endm  AfisInt macro x  mov eax, x  call writeint  call crlf  endm  AfisArrayHex macro src  Local L0  lea esi, src  mov ecx, lengthof src  L0:  xor eax, eax  movzx eax, word ptr [esi]  call writehex  mov al, ' '  call writechar  add esi, type src  loop L0  call crlf  endm  copyArray macro src, dest  local L0  lea esi, src  lea edi, dest  mov ecx, lengthof src  L0:  movzx eax, word ptr [esi]  mov word ptr [edi], ax  add esi, type src  add edi, type dest  loop L0  endm  Inpacheteaza4To3 macro src ; scr - sursa  Local L0, L1, L2    mov ecx, sizeof arr - type arr  mov edx, 0  lea esi, src + [ecx]  L0:  mov edi, esi  shl word ptr [edi], 4  push ecx  mov ecx, sizeof arr - type arr  L1:    mov ax, word ptr [edi - type src]  mov bx, word ptr [edi]  shld ax, bx, 4  mov word ptr [edi - type src], ax  sub edi, type src  sub ecx, type src  cmp ecx, edx  jne L1  pop ecx  add edx, type src  sub ecx, type src  cmp ecx, 0    jne L0  mov word ptr [esi], 0  endm  main proc  AfisArrayHex arr  CopyArray arr, arr2  Inpacheteaza4To3 arr2  AfisArrayHex arr2  exit  main ENDP  END main |

Listingul programului pe 32 biți.

|  |
| --- |
| Microsoft (R) Macro Assembler Version 14.16.27026.1 04/16/19 00:59:43  programator.asm Page 1 - 1  INCLUDE Irvine32.inc  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  00000000 0123 0456 0789 arr DW 0123h, 0456h, 0789h, 0abch  0ABC  = 00000004 lenArr EQU lengthof arr  00000008 00000004 [ arr2 DW lenArr dup(0)  0000  ]  00000010 00 a DB 0  00000011 00 b DB 0  00000000 .CODE  AfisStr macro x  lea edx, x  call writestring  endm  AfisStrLn macro x  afisstr x  call crlf  endm  AfisInt macro x  mov eax, x  call writeint  call crlf  endm  AfisArrayHex macro src  Local L0  lea esi, src  mov ecx, lengthof src  L0:  xor eax, eax  movzx eax, word ptr [esi]  call writehex  mov al, ' '  call writechar  add esi, type src  loop L0  call crlf  endm  copyArray macro src, dest  local L0  lea esi, src  lea edi, dest  mov ecx, lengthof src  L0:  movzx eax, word ptr [esi]  mov word ptr [edi], ax  add esi, type src  add edi, type dest  loop L0  endm  Inpacheteaza4To3 macro src ; scr - sursa  Local L0, L1, L2    mov ecx, sizeof arr - type arr  mov edx, 0  lea esi, src + [ecx]  L0:  mov edi, esi  shl word ptr [edi], 4  push ecx  mov ecx, sizeof arr - type arr  L1:    mov ax, word ptr [edi - type src]  mov bx, word ptr [edi]  shld ax, bx, 4  mov word ptr [edi - type src], ax  sub edi, type src  sub ecx, type src  cmp ecx, edx  jne L1  pop ecx  add edx, type src  sub ecx, type src  cmp ecx, 0    jne L0  mov word ptr [esi], 0  endm  00000000 main proc  AfisArrayHex arr  00000000 8D 35 00000000 R 1 lea esi, arr  00000006 B9 00000004 1 mov ecx, lengthof arr  0000000B 1 ??0000:  0000000B 33 C0 1 xor eax, eax  0000000D 0F B7 06 1 movzx eax, word ptr [esi]  00000010 E8 00000000 E 1 call writehex  00000015 B0 20 1 mov al, ' '  00000017 E8 00000000 E 1 call writechar  0000001C 83 C6 02 1 add esi, type arr  0000001F E2 EA 1 loop ??0000  00000021 E8 00000000 E 1 call crlf  CopyArray arr, arr2  00000026 8D 35 00000000 R 1 lea esi, arr  0000002C 8D 3D 00000008 R 1 lea edi, arr2  00000032 B9 00000004 1 mov ecx, lengthof arr  00000037 1 ??0001:  00000037 0F B7 06 1 movzx eax, word ptr [esi]  0000003A 66| 89 07 1 mov word ptr [edi], ax  0000003D 83 C6 02 1 add esi, type arr  00000040 83 C7 02 1 add edi, type arr2  00000043 E2 F2 1 loop ??0001  Inpacheteaza4To3 arr2  00000045 B9 00000006 1 mov ecx, sizeof arr - type arr  0000004A BA 00000000 1 mov edx, 0  0000004F 8D B1 00000008 R 1 lea esi, arr2 + [ecx]  00000055 1 ??0002:  00000055 8B FE 1 mov edi, esi  00000057 66| C1 27 04 1 shl word ptr [edi], 4  0000005B 51 1 push ecx  0000005C B9 00000006 1 mov ecx, sizeof arr - type arr  00000061 1 ??0003:  00000061 66| 8B 47 FE 1 mov ax, word ptr [edi - type arr2]  00000065 66| 8B 1F 1 mov bx, word ptr [edi]  00000068 66| 0F A4 D8 1 shld ax, bx, 4  04  0000006D 66| 89 47 FE 1 mov word ptr [edi - type arr2], ax  00000071 83 EF 02 1 sub edi, type arr2  00000074 83 E9 02 1 sub ecx, type arr2  00000077 3B CA 1 cmp ecx, edx  00000079 75 E6 1 jne ??0003  0000007B 59 1 pop ecx  0000007C 83 C2 02 1 add edx, type arr2  0000007F 83 E9 02 1 sub ecx, type arr2  00000082 83 F9 00 1 cmp ecx, 0  00000085 75 CE 1 jne ??0002  00000087 66| C7 06 0000 1 mov word ptr [esi], 0  AfisArrayHex arr2  0000008C 8D 35 00000008 R 1 lea esi, arr2  00000092 B9 00000004 1 mov ecx, lengthof arr2  00000097 1 ??0005:  00000097 33 C0 1 xor eax, eax  00000099 0F B7 06 1 movzx eax, word ptr [esi]  0000009C E8 00000000 E 1 call writehex  000000A1 B0 20 1 mov al, ' '  000000A3 E8 00000000 E 1 call writechar  000000A8 83 C6 02 1 add esi, type arr2  000000AB E2 EA 1 loop ??0005  000000AD E8 00000000 E 1 call crlf  exit  000000B2 6A 00 \* push +000000000h  000000B4 E8 00000000 E \* call ExitProcess  000000B9 main ENDP  END main |