Министерство просвещения Республики Молдова

Технический университет Молдовы

Кафедра Автоматики и Информационных технологий

Отчет

По лабораторной работе №5

по предмету: **Архитектура компьютера**.

# *Тема:* СОЗДАНИЕ И ОБРАБОТКА СЛОЖНЫХ СТРУКТУР ДАННЫХ.

Выполнила: ст.гр.TI-176 Изотова Влада

Проверил: Колесник В.

Кишинев 2019

***Цели работы:***

Работа рассматривает создание и обработку массивов, структур, объединений и записей.

***Задания к лабораторной работе :***

* Необходимо выполнить ассемблирование приведенных примеров (примеры 5.1-5.5) и проанализировать их в пошаговом режиме Debug.

***Индивидуальные задания:***

* Предполагаются две последовательности символов mas1, mas2 и некий символ *simv*. Произвести поиск и удаление *simv* из массива mas1 и отсортированный массив занести в mas2. Вывести все массивы на монитор.
* Предполагается последовательность mas1 из n символов (mas=1, 2,…, i,…, k,…, n). Необходимо: а) объединить последовательности от 1 до i и от k до n и занести в mas2; б) последовательность от i до k занести в mas3.

***Выполнение заданий:***

Пример 5.3 Обработка массива элементов с нечетной длиной

**INCLUDE Irvine32.inc**

**.data ;начало сегмента данных**

**N=5 ;количество элементов массива**

**mas byte 5 dup (3 dup (0))**

**.code ;сегмент кода**

**main proc ;точка входа в программу**

**xor eax,eax ;обнуление eax**

**mov esi,0 ;0 в esi**

**mov ecx,N ;N в ecx**

**go:**

**mov dl,mas[esi] ;первый байт поля в dl**

**inc dl ;увеличение dl на 1 (по условию)**

**mov mas[esi],dl ;запись обратно в массив**

**add esi,3 ;сдвиг на следующий элемент массива**

**loop go ;повтор цикла**

**mov esi,0 ;подготовка к выводу на экран**

**mov ecx,N**

**show: ;вывод на экран содержимого**

**;первых байт полей**

**movsx eax,mas[esi]**

**call WriteDec**

**add esi,3**

**loop show**

**quit:**

**call crlf**

**exit ; выход**

**main ENDP**

**END main ;конец программы**

**Листинг программы:**

*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 ;начало сегмента данных*

*= 00000005 N=5 ;количество элементов массива*

*00000000 00000005 [ mas byte 5 dup (3 dup (0))*

*00000003 [*

*00*

*]*

*]*

*00000000 .code ;сегмент кода*

*00000000 main proc ;точка входа в программу*

*00000000 33 C0 xor eax,eax ;обнуление eax*

*00000002 BE 00000000 mov esi,0 ;0 в esi*

*00000007 B9 00000005 mov ecx,N ;N в ecx*

*0000000C go:*

*0000000C 8A 96 00000000 R mov dl,mas[esi] ;первый байт поля в dl*

*00000012 FE C2 inc dl ;увеличение dl на 1 (по условию)*

*00000014 88 96 00000000 R mov mas[esi],dl ;запись обратно в массив*

*0000001A 83 C6 03 add esi,3 ;сдвиг на следующий элемент массива*

*0000001D E2 ED loop go ;повтор цикла*

*0000001F BE 00000000 mov esi,0 ;подготовка к выводу на экран*

*00000024 B9 00000005 mov ecx,N*

*00000029 show: ;вывод на экран содержимого*

*;первых байт полей*

*00000029 0F BE 86 movsx eax,mas[esi]*

*00000000 R*

*00000030 E8 00000000 E call WriteDec*

*00000035 83 C6 03 add esi,3*

*00000038 E2 EF loop show*

*0000003A quit:*

*0000003A E8 00000000 E call crlf*

*exit ; выход*

*0000003F 6A 00 \* push +000000000h*

*00000041 E8 00000000 E \* call ExitProcess*

*00000046 main ENDP*

*END main ;конец программы*

Пример 5.4 - Поиск элемента в двухмерном массиве.

**INCLUDE Irvine32.inc**

**.data**

**;матрица размером 2x5 — если ее не инициализировать,**

**;то для наглядности она может быть описана так:**

**;array dw 2 DUP (5 DUP (?))**

**;но мы ее инициализируем:**

**array dw 1,2,3,4,5,6,7,3,9,0**

**;логически это будет выглядеть так:**

**;array= {1 2}**

**; {3 4}**

**; {5 6}**

**; {7 3}**

**; {9 0}**

**elem dw 3 ;элемент для поиска**

**failed db 0ah,0dh,'Net takogo elementa v masive!',0**

**success db 0ah,0dh,'Takoi element v masive prisutstvuet ',0**

**foundtime db ? ;количество найденных элементов**

**fnd db ' raz(a)',0ah,0dh,0**

**.code**

**main proc**

**xor eax,eax**

**mov esi,0 ;esi=столбцы в матрице**

**mov ebx,0 ;ebx=строки в матрице**

**mov ecx,5 ;число для внешнего цикла (по строкам)**

**external: ;внешний цикл по строкам**

**mov ax,array[ebx][esi] ;в ax первый элемент матрицы**

**push ecx ;сохранение в стеке счётчика внешнего цикла**

**mov ecx,2 ;число для внутреннего цикла (по столбцам)**

**mov esi,0**

**iternal: ;внутренний цикл по строкам**

**inc esi ;передвижение на следующий элемент в строке**

**;сравниваем содержимое текущего элемента в ax с искомым элементом:**

**cmp ax,elem**

**;если текущий совпал с искомым, то переход на here для обработки,**

**;иначе цикл продолжения поиска**

**Je here**

**;иначе — цикл по строке ecx=2 раз**

**loop iternal**

**here:**

**jecxz move\_next ;просмотрели строку?**

**inc foundtime ;иначе увеличиваем счётчик совпавших**

**move\_next: ;продвижение в матрице**

**pop ecx ;восстанавливаем ECX из стека (5)**

**add ebx,1 ;передвигаемся на следующую строку**

**loop external ;цикл (внешний)**

**cmp foundtime,0h ;сравнение числа совпавших с 0**

**ja eql ;если больше 0, то переход**

**not\_equal: ;нет элементов, совпавших с искомым**

**mov edx,OFFSET failed**

**call WriteString ;вывод сообщения на экран**

**jmp quit ;на выход**

**eql: ;есть элементы, совпавшие с искомым**

**mov edx,OFFSET success**

**call WriteString ;вывод сообщения на экран**

**movsx eax,foundtime**

**call WriteDec**

**mov edx,OFFSET fnd**

**call WriteString**

**quit: ;выход**

**call crlf**

**exit ; выход**

**main ENDP**

**END main ;конец программы**

**Листинг программы:**

*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*

*;матрица размером 2x5 — если ее не инициализировать,*

*;то для наглядности она может быть описана так:*

*;array dw 2 DUP (5 DUP (?))*

*;но мы ее инициализируем:*

*00000000 0001 0002 0003 array dw 1,2,3,4,5,6,7,3,9,0*

*0004 0005 0006*

*0007 0003 0009*

*0000*

*;логически это будет выглядеть так:*

*;array= {1 2}*

*; {3 4}*

*; {5 6}*

*; {7 3}*

*; {9 0}*

*00000014 0003 elem dw 3 ;элемент для поиска*

*00000016 0A 0D 4E 65 74 failed db 0ah,0dh,'Net takogo elementa v masive!',0*

*20 74 61 6B 6F*

*67 6F 20 65 6C*

*65 6D 65 6E 74*

*61 20 76 20 6D*

*61 73 69 76 65*

*21 00*

*00000036 0A 0D 54 61 6B success db 0ah,0dh,'Takoi element v masive prisutstvuet ',0*

*6F 69 20 65 6C*

*65 6D 65 6E 74*

*20 76 20 6D 61*

*73 69 76 65 20*

*70 72 69 73 75*

*74 73 74 76 75*

*65 74 20 00*

*0000005D 00 foundtime db ? ;количество найденных элементов*

*0000005E 20 72 61 7A 28 fnd db ' raz(a)',0ah,0dh,0*

*61 29 0A 0D 00*

*00000000 .code*

*00000000 main proc*

*00000000 33 C0 xor eax,eax*

*00000002 BE 00000000 mov esi,0 ;esi=столбцы в матрице*

*00000007 BB 00000000 mov ebx,0 ;ebx=строки в матрице*

*0000000C B9 00000005 mov ecx,5 ;число для внешнего цикла (по строкам)*

*00000011 external: ;внешний цикл по строкам*

*00000011 66| 8B 84 1E mov ax,array[ebx][esi] ;в ax первый элемент матрицы*

*00000000 R*

*00000019 51 push ecx ;сохранение в стеке счётчика внешнего цикла*

*0000001A B9 00000002 mov ecx,2 ;число для внутреннего цикла (по столбцам)*

*0000001F BE 00000000 mov esi,0*

*00000024 iternal: ;внутренний цикл по строкам*

*00000024 46 inc esi ;передвижение на следующий элемент в строке*

*;сравниваем содержимое текущего элемента в ax с искомым элементом:*

*00000025 66| 3B 05 cmp ax,elem*

*00000014 R*

*;если текущий совпал с искомым, то переход на here для обработки,*

*;иначе цикл продолжения поиска*

*0000002C 74 02 Je here*

*;иначе — цикл по строке ecx=2 раз*

*0000002E E2 F4 loop iternal*

*00000030 here:*

*00000030 E3 06 jecxz move\_next ;просмотрели строку?*

*00000032 FE 05 0000005D R inc foundtime ;иначе увеличиваем счётчик совпавших*

*00000038 move\_next: ;продвижение в матрице*

*00000038 59 pop ecx ;восстанавливаем ECX из стека (5)*

*00000039 83 C3 01 add ebx,1 ;передвигаемся на следующую строку*

*0000003C E2 D3 loop external ;цикл (внешний)*

*0000003E 80 3D 0000005D R cmp foundtime,0h ;сравнение числа совпавших с 0*

*00*

*00000045 77 0C ja eql ;если больше 0, то переход*

*00000047 not\_equal: ;нет элементов, совпавших с искомым*

*00000047 BA 00000016 R mov edx,OFFSET failed*

*0000004C E8 00000000 E call WriteString ;вывод сообщения на экран*

*00000051 EB 20 jmp quit ;на выход*

*00000053 eql: ;есть элементы, совпавшие с искомым*

*00000053 BA 00000036 R mov edx,OFFSET success*

*00000058 E8 00000000 E call WriteString ;вывод сообщения на экран*

*0000005D 0F BE 05 movsx eax,foundtime*

*0000005D R*

*00000064 E8 00000000 E call WriteDec*

*00000069 BA 0000005E R mov edx,OFFSET fnd*

*0000006E E8 00000000 E call WriteString*

*00000073 quit: ;выход*

*00000073 E8 00000000 E call crlf*

*exit ; выход*

*00000078 6A 00 \* push +000000000h*

*0000007A E8 00000000 E \* call ExitProcess*

*0000007F main ENDP*

*END main ;конец программы*

Пример 5.5 - Сортировка массива

**INCLUDE Irvine32.inc**

**.data**

**mes1 db 0ah,0dh,'Ishodnii massiv'**

**;некоторые сообщения**

**mes2 db 0ah,0dh,'Otsortirovanii massiv',0ah,0dh,0**

**n equ 9 ;количество элементов в массиве, считая с 0**

**mas dw 2,7,4,0,1,9,3,6,5,8 ;исходный массив**

**tmp dw 0 ;переменные для работы с массивом**

**I dw 0**

**J dw 0**

**.code**

**main proc**

**xor eax,eax**

**;вывод на экран исходного массива**

**mov edx,OFFSET mes1 ;вывод сообщения mes1**

**call WriteString**

**mov ecx,10**

**mov esi,0**

**show\_primary: ;вывод значения элементов**

**;исходного массива на экран**

**movsx eax,mas[esi]**

**call WriteDec**

**add esi,2**

**loop show\_primary**

**;строки с метки M1 до метки M2 программы эквивалентны ;следующему коду на языке С:**

**;for (i=0;i<9;i++)**

**; for (j=9;j>i;j--)**

**; if (mas[i]>mas[j])**

**; {tmp=mas[i];**

**; mas[i]=mas[j];**

**; mas[j]=tmp;}**

**M1: mov i,0 ;инициализация i**

**;внутренний цикл по j**

**internal:**

**mov j,9 ;инициализация j**

**jmp cycl\_j ;переход на тело цикла**

**exchange:**

**movsx ebx,i ;bx=i**

**shl ebx,1**

**mov ax,mas[ebx] ;ax=mas[i]**

**movsx ebx,j ;bx=j**

**shl ebx,1**

**cmp ax,mas[ebx] ;mas[i] ? mas[j] — сравнение элементов**

**jle lesser ;если mas[i] меньше, то обмен не нужен и**

**;переход на продвижение далее по массиву**

**;иначе tmp=mas[i], mas[i]=mas[j], mas[j]=tmp:**

**;tmp=mas[i]**

**movsx ebx,i ;ebx=i**

**shl ebx,1 ;умножаем на 2, так как элементы — слова**

**mov tmp,ax ;tmp=mas[i]**

**;mas[i]=mas[j]**

**movsx ebx,j ;bx=j**

**shl ebx,1 ;умножаем на 2, так как элементы — слова**

**mov ax,mas[ebx] ;ax=mas[j]**

**movsx ebx,i ;bx=i**

**shl ebx,1 ;умножаем на 2, так как элементы — слова**

**mov mas[ebx],ax ;mas[i]=mas[j]**

**;mas[j]=tmp**

**movsx ebx,j ;bx=j**

**shl ebx,1 ;умножаем на 2, так как элементы — слова**

**mov ax,tmp ;ax=tmp**

**mov mas[ebx],ax ;mas[j]=tmp**

**lesser: ;продвижение далее по массиву во внутреннем цикле**

**dec j ;j-**

**;тело цикла по j**

**cycl\_j:**

**mov ax,j ;ax=j**

**cmp ax,i ;сравнить j ? i**

**jg exchange ;если j>i, то переход на обмен**

**;иначе на внешний цикл по i**

**inc i ;i+**

**cmp i,n ;сравнить i ? n — прошли до конца массива**

**jl internal ;если i**

**M2: ;вывод отсортированного массива**

**mov edx,OFFSET mes2 ;вывод сообщения mes2**

**call WriteString**

**prepare:**

**mov ecx,10**

**mov esi,0**

**show: ;вывод значения элемента на экран**

**movsx eax,mas[esi]**

**call WriteDec**

**add esi,2**

**loop show**

**quit:**

**call crlf**

**exit ; выход**

**main ENDP**

**END main ;конец программы**

**Листинг программы:**

*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 0A 0D 49 73 68 mes1 db 0ah,0dh,'Ishodnii massiv'*

*6F 64 6E 69 69*

*20 6D 61 73 73*

*69 76*

*;некоторые сообщения*

*00000011 0A 0D 4F 74 73 mes2 db 0ah,0dh,'Otsortirovanii massiv',0ah,0dh,0*

*6F 72 74 69 72*

*6F 76 61 6E 69*

*69 20 6D 61 73*

*73 69 76 0A 0D*

*00*

*= 00000009 n equ 9 ;количество элементов в массиве, считая с 0*

*0000002B 0002 0007 0004 mas dw 2,7,4,0,1,9,3,6,5,8 ;исходный массив*

*0000 0001 0009*

*0003 0006 0005*

*0008*

*0000003F 0000 tmp dw 0 ;переменные для работы с массивом*

*00000041 0000 I dw 0*

*00000043 0000 J dw 0*

*00000000 .code*

*00000000 main proc*

*00000000 33 C0 xor eax,eax*

*;вывод на экран исходного массива*

*00000002 BA 00000000 R mov edx,OFFSET mes1 ;вывод сообщения mes1*

*00000007 E8 00000000 E call WriteString*

*0000000C B9 0000000A mov ecx,10*

*00000011 BE 00000000 mov esi,0*

*00000016 show\_primary: ;вывод значения элементов*

*;исходного массива на экран*

*00000016 0F BF 86 movsx eax,mas[esi]*

*0000002B R*

*0000001D E8 00000000 E call WriteDec*

*00000022 83 C6 02 add esi,2*

*00000025 E2 EF loop show\_primary*

*;строки с метки M1 до метки M2 программы эквивалентны ;следующему коду на языке С:*

*;for (i=0;i<9;i++)*

*; for (j=9;j>i;j--)*

*; if (mas[i]>mas[j])*

*; {tmp=mas[i];*

*; mas[i]=mas[j];*

*; mas[j]=tmp;}*

*00000027 66| C7 05 M1: mov i,0 ;инициализация i*

*00000041 R*

*0000*

*;внутренний цикл по j*

*00000030 internal:*

*00000030 66| C7 05 mov j,9 ;инициализация j*

*00000043 R*

*0009*

*00000039 EB 6E jmp cycl\_j ;переход на тело цикла*

*0000003B exchange:*

*0000003B 0F BF 1D movsx ebx,i ;bx=i*

*00000041 R*

*00000042 D1 E3 shl ebx,1*

*00000044 66| 8B 83 mov ax,mas[ebx] ;ax=mas[i]*

*0000002B R*

*0000004B 0F BF 1D movsx ebx,j ;bx=j*

*00000043 R*

*00000052 D1 E3 shl ebx,1*

*00000054 66| 3B 83 cmp ax,mas[ebx] ;mas[i] ? mas[j] — сравнение элементов*

*0000002B R*

*0000005B 7E 45 jle lesser ;если mas[i] меньше, то обмен не нужен и*

*;переход на продвижение далее по массиву*

*;иначе tmp=mas[i], mas[i]=mas[j], mas[j]=tmp:*

*;tmp=mas[i]*

*0000005D 0F BF 1D movsx ebx,i ;ebx=i*

*00000041 R*

*00000064 D1 E3 shl ebx,1 ;умножаем на 2, так как элементы — слова*

*00000066 66| A3 mov tmp,ax ;tmp=mas[i]*

*0000003F R*

*;mas[i]=mas[j]*

*0000006C 0F BF 1D movsx ebx,j ;bx=j*

*00000043 R*

*00000073 D1 E3 shl ebx,1 ;умножаем на 2, так как элементы — слова*

*00000075 66| 8B 83 mov ax,mas[ebx] ;ax=mas[j]*

*0000002B R*

*0000007C 0F BF 1D movsx ebx,i ;bx=i*

*00000041 R*

*00000083 D1 E3 shl ebx,1 ;умножаем на 2, так как элементы — слова*

*00000085 66| 89 83 mov mas[ebx],ax ;mas[i]=mas[j]*

*0000002B R*

*;mas[j]=tmp*

*0000008C 0F BF 1D movsx ebx,j ;bx=j*

*00000043 R*

*00000093 D1 E3 shl ebx,1 ;умножаем на 2, так как элементы — слова*

*00000095 66| A1 mov ax,tmp ;ax=tmp*

*0000003F R*

*0000009B 66| 89 83 mov mas[ebx],ax ;mas[j]=tmp*

*0000002B R*

*000000A2 lesser: ;продвижение далее по массиву во внутреннем цикле*

*000000A2 66| FF 0D dec j ;j-*

*00000043 R*

*;тело цикла по j*

*000000A9 cycl\_j:*

*000000A9 66| A1 mov ax,j ;ax=j*

*00000043 R*

*000000AF 66| 3B 05 cmp ax,i ;сравнить j ? i*

*00000041 R*

*000000B6 7F 83 jg exchange ;если j>i, то переход на обмен*

*;иначе на внешний цикл по i*

*000000B8 66| FF 05 inc i ;i+*

*00000041 R*

*000000BF 66| 83 3D cmp i,n ;сравнить i ? n — прошли до конца массива*

*00000041 R 09*

*000000C7 0F 8C FFFFFF63 jl internal ;если i*

*000000CD M2: ;вывод отсортированного массива*

*000000CD BA 00000011 R mov edx,OFFSET mes2 ;вывод сообщения mes2*

*000000D2 E8 00000000 E call WriteString*

*000000D7 prepare:*

*000000D7 B9 0000000A mov ecx,10*

*000000DC BE 00000000 mov esi,0*

*000000E1 show: ;вывод значения элемента на экран*

*000000E1 0F BF 86 movsx eax,mas[esi]*

*0000002B R*

*000000E8 E8 00000000 E call WriteDec*

*000000ED 83 C6 02 add esi,2*

*000000F0 E2 EF loop show*

*000000F2 quit:*

*000000F2 E8 00000000 E call crlf*

*exit ; выход*

*000000F7 6A 00 \* push +000000000h*

*000000F9 E8 00000000 E \* call ExitProcess*

*000000FE main ENDP*

*END main ;конец программы*

***Индивидуальные задания:***

Предполагаются две последовательности символов mas1, mas2 и некий символ *simv*. Произвести поиск и удаление *simv* из массива mas1 и отсортированный массив занести в mas2. Вывести все массивы на монитор.

INCLUDE Irvine32.inc ; резервирует пространство для стека программы (16 байт)

.data ; начало сегмента инициализированных и неинициализированных данных.

mas1 db 81 dup(?)

mas2 db 81 dup(?) ; резервирование в памяти 16 байт по адресу переменной unpacked

n db ? ; байт памяти со значением 8 по адресу переменной count

;mes1 db 'Vvedite kol-vo simvolov ',0

mes2 db 'Vvedite posledovatelinosti simvolov ',0

mes3 db 'Vvedite simvol simv ',0

mes4 db 'Otsortirovannii massiv ', 0

mes5 db 'V massive net elementa simv', 0

simv db ?

i db ?

j db ?

tmp db ?

symbol equ '$'

.code

main PROC

xor eax,eax

mov edx,OFFSET mes2 ;вывод сообщения mes2

call WriteString

xor edx,edx

mov edx,OFFSET mas1 ;ввод сообщения mas1

mov ecx,80

call ReadString

; подсчет колличества символов в mas1

xor ecx,ecx

xor esi,esi

lea esi,mas1

next1:

cmp byte ptr [esi],symbol ; сравниваем с символом

jz notfound

inc ecx ; Если это он увеличиваем счетчик

notfound:

inc esi

cmp byte ptr [esi],0 ; если следующий символ 0...

jz o ; ,то это конец строки

jmp next1 ; если нет, то повторяем.

o:

mov n,cl ;колличество символов в n

mov edx,OFFSET mas1 ;вывод сообщения mas1

call WriteString

call Crlf

xor esi,esi ; очистка регистров

xor edx,edx

xor ecx,ecx

xor eax,eax

mov edx,OFFSET mes3 ;вывод сообщения mes3

call WriteString

mov ecx,1

call ReadChar

call WriteChar

call Crlf

mov simv,al

mov bl,al

mov di,0

next:

xor eax,eax

xor ecx,ecx

mov al,n

mov cl,al

mov si,0

go1: mov al,mas1[si]

cmp al,bl

jz udalenie

inc si

loop go1

jmp vivod

udalenie:

mov al,mas1[si+1]

mov mas1[si],al

inc si

loop udalenie

inc di

mov al,n

sub al,1

mov n,al

jmp next

da:

mov edx,offset mes5

call WriteString

jmp sort

vivod:

xor esi,esi ; очистка регистров

xor edx,edx

xor ecx,ecx

xor eax,eax

mov al,n

mov ecx,eax ; запись в регистр cx значения 10h

dec eax

mov si,ax

xor edx,edx

mov edx,OFFSET mas1 ;вывод сообщения mas1

call WriteString

call Crlf

cmp di,0

jz da

sort:

mov esi,0

mov edi,0

;mov al,mas1[si]

mov i,0

internal:

mov bh,n

mov j,bh

mov di,0

mov cl,n

l1:

inc di

loop l1

dec di

jmp cycl\_j

exchange:

xor eax,eax

mov al,mas1[si]

cmp al,mas1[di]

jle lesser

mov tmp,al

mov al,mas1[di]

mov mas1[si],al

mov al,tmp

mov mas1[di],al

lesser:

dec di

dec j

cycl\_j:

mov bl,j

cmp bl,i

jg exchange

inc si

inc i

mov al,i

cmp al,n

jl internal

mov edx,offset mes4

call WriteString

mov cl,n

mov si,0

xor esi,esi ; очистка регистров

xor edx,edx

xor ecx,ecx

xor eax,eax

mov al,n

mov ecx,eax ; запись в регистр cx значения 10h

;dec eax

mov si,0

mov edx,OFFSET mas1 ;вывод сообщения mas1

call WriteString

call Crlf

xor esi,esi ; очистка регистров

xor edx,edx

xor ecx,ecx

xor eax,eax

mov al,n

mov ecx,eax ; запись в регистр cx значения 10h

;dec eax

mov si,0

zanos:

mov al,mas1[si]

mov mas2[si],al

inc si

loop zanos

xor esi,esi ; очистка регистров

xor edx,edx

xor ecx,ecx

xor eax,eax

mov al,n

mov ecx,eax ; запись в регистр cx значения 10h

;dec eax

mov si,0

mov edx,OFFSET mas2 ;вывод сообщения mas2

call WriteString

call Crlf

exit

main ENDP

END main

**/\*INCLUDE Irvine32.inc**

**.data ; начало сегмента инициализированных и неинициализированных данных.**

**mas1 db 10 dup(?)**

**mas2 db 10 dup(?)**

**n db ?**

**mes1 db 0ah,0dh, 'Vvedite kol-vo simvolov ',0ah,0dh,0**

**mes2 db 0ah,0dh,'Vvedite posledovatelinosti simvolov ',0ah,0dh,0**

**mes3 db 0ah,0dh,'Vvedite simvol simv ',0ah,0dh,0**

**mes4 db 0ah,0dh,'Otsortirovannii massiv ',0ah,0dh,0**

**mes5 db 0ah,0dh,'V massive net elementa simv',0ah,0dh,0**

**simv db ?**

**i db ?**

**j db ?**

**tmp db ?**

**.code**

**main PROC**

**xor eax,eax**

**mov edx,OFFSET mes1 ;вывод сообщения mes1**

**call WriteString**

**call ReadDec**

**mov n,al**

**xor eax,eax**

**mov edx,OFFSET mes2 ;вывод сообщения mes2**

**call WriteString**

**xor eax,eax**

**mov al,n**

**mov esi,eax**

**m1:**

**xor eax,eax**

**call ReadDec**

**mov mas1[si-1],al**

**dec si**

**jnz m1**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**dec eax**

**mov si,ax**

**show\_primary: ; метка начала цикла**

**xor edx, edx ; очистка регистра dx**

**mov dl, mas1[si]**

**mov al,dl**

**call WriteDec**

**sub esi,1**

**loop show\_primary**

**call crlf**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**mov esi,eax**

**xor eax,eax**

**mov edx,OFFSET mes3 ;вывод сообщения mes3**

**call WriteString**

**call ReadDec**

**mov simv,al**

**mov bl,simv**

**mov di,0**

**next:**

**xor eax,eax**

**xor ecx,ecx**

**mov al,n**

**mov cl,al**

**mov si,0**

**go1:**

**mov al,mas1[si]**

**cmp al,bl**

**jz udalenie**

**inc si**

**loop go1**

**jmp vivod**

**udalenie:**

**mov al,mas1[si+1]**

**mov mas1[si],al**

**inc si**

**loop udalenie**

**inc di**

**mov al,n**

**sub al,1**

**mov n,al**

**jmp next**

**da:**

**mov edx,offset mes5**

**call WriteString**

**jmp sort**

**vivod:**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**dec eax**

**mov si,ax**

**show\_primar: ; метка начала цикла**

**xor edx, edx ; очистка регистра edx**

**mov dl, mas1[si]**

**mov al,dl**

**call WriteDec**

**sub esi,1**

**loop show\_primar**

**call crlf**

**cmp di,0**

**jz da**

**sort:**

**mov esi,0**

**mov i,0**

**internal:**

**mov bh,n ;j=n**

**mov j,bh**

**mov edi,0**

**mov cl,n**

**l1:**

**inc di**

**loop l1**

**dec di ;n-1**

**jmp cycl\_j**

**exchange:**

**mov al,mas1[si] ;Помещает в al элемент массива с конца**

**cmp al,mas1[di] ;Сравнивает с первым и т.д**

**jle lesser ;Если последний меньше первого, то переход на метку**

**mov tmp,al ;Перестановка (обмен местами)**

**mov al,mas1[di]**

**mov mas1[si],al**

**mov al,tmp**

**mov mas1[di],al**

**lesser:**

**dec di ;Уменьшает di**

**dec j ;Уменьшает j**

**cycl\_j:**

**mov bl,j ;j=n**

**cmp bl,I ;i=0**

**jg exchange**

**inc si**

**inc i**

**mov al,i**

**cmp al,n**

**jl internal**

**mov edx,offset mes4**

**call WriteString**

**mov cl,n**

**mov si,0**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**mov si,0**

**show\_primar1: ; метка начала цикла**

**xor edx, edx ; очистка регистра edx**

**mov dl, mas1[si]**

**mov al,dl**

**call WriteDec**

**add esi,1**

**loop show\_primar1**

**call crlf**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**mov si,0**

**zanos:**

**mov al,mas1[si]**

**mov mas2[si],al**

**inc si**

**loop zanos**

**xor esi,esi ; очистка регистров**

**xor edx,edx**

**xor ecx,ecx**

**xor eax,eax**

**mov al,n**

**mov ecx,eax**

**mov si,0**

**show\_primar2: ; метка начала цикла**

**xor edx, edx ; очистка регистра dx**

**mov dl, mas1[si]**

**mov al,dl**

**call WriteDec**

**add esi,1**

**loop show\_primar2**

**call crlf**

**exit**

**main ENDP**

**END main\*/**

**Листинг программы:**

*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 ; начало сегмента инициализированных и неинициализированных данных.*

*00000000 0000000A [ mas1 db 10 dup(?)*

*00*

*]*

*0000000A 0000000A [ mas2 db 10 dup(?)*

*00*

*]*

*00000014 00 n db ?*

*00000015 0A 0D 56 76 65 mes1 db 0ah,0dh, 'Vvedite kol-vo simvolov ',0ah,0dh,0*

*64 69 74 65 20*

*6B 6F 6C 2D 76*

*6F 20 73 69 6D*

*76 6F 6C 6F 76*

*20 0A 0D 00*

*00000032 0A 0D 56 76 65 mes2 db 0ah,0dh,'Vvedite posledovatelinosti simvolov ',0ah,0dh,0*

*64 69 74 65 20*

*70 6F 73 6C 65*

*64 6F 76 61 74*

*65 6C 69 6E 6F*

*73 74 69 20 73*

*69 6D 76 6F 6C*

*6F 76 20 0A 0D*

*00*

*0000005B 0A 0D 56 76 65 mes3 db 0ah,0dh,'Vvedite simvol simv ',0ah,0dh,0*

*64 69 74 65 20*

*73 69 6D 76 6F*

*6C 20 73 69 6D*

*76 20 0A 0D 00*

*00000074 0A 0D 4F 74 73 mes4 db 0ah,0dh,'Otsortirovannii massiv ',0ah,0dh,0*

*6F 72 74 69 72*

*6F 76 61 6E 6E*

*69 69 20 6D 61*

*73 73 69 76 20*

*0A 0D 00*

*00000090 0A 0D 56 20 6D mes5 db 0ah,0dh,'V massive net elementa simv',0ah,0dh,0*

*61 73 73 69 76*

*65 20 6E 65 74*

*20 65 6C 65 6D*

*65 6E 74 61 20*

*73 69 6D 76 0A*

*0D 00*

*000000B0 00 simv db ?*

*000000B1 00 i db ?*

*000000B2 00 j db ?*

*000000B3 00 tmp db ?*

*00000000 .code*

*00000000 main PROC*

*00000000 33 C0 xor eax,eax*

*00000002 BA 00000015 R mov edx,OFFSET mes1 ;вывод сообщения mes1*

*00000007 E8 00000000 E call WriteString*

*0000000C E8 00000000 E call ReadDec*

*00000011 A2 00000014 R mov n,al*

*00000016 33 C0 xor eax,eax*

*00000018 BA 00000032 R mov edx,OFFSET mes2 ;вывод сообщения mes2*

*0000001D E8 00000000 E call WriteString*

*00000022 33 C0 xor eax,eax*

*00000024 A0 00000014 R mov al,n*

*00000029 8B F0 mov esi,eax*

*0000002B m1:*

*0000002B 33 C0 xor eax,eax*

*0000002D E8 00000000 E call ReadDec*

*00000032 88 04 35 mov mas1[si-1],al*

*FFFFFFFF R*

*00000039 66| 4E dec si*

*0000003B 75 EE jnz m1*

*0000003D 33 F6 xor esi,esi ; очистка регистров*

*0000003F 33 D2 xor edx,edx*

*00000041 33 C9 xor ecx,ecx*

*00000043 33 C0 xor eax,eax*

*00000045 A0 00000014 R mov al,n*

*0000004A 8B C8 mov ecx,eax*

*0000004C 48 dec eax*

*0000004D 66| 8B F0 mov si,ax*

*00000050 show\_primary: ; метка начала цикла*

*00000050 33 D2 xor edx, edx ; очистка регистра dx*

*00000052 8A 14 35 mov dl, mas1[si]*

*00000000 R*

*00000059 8A C2 mov al,dl*

*0000005B E8 00000000 E call WriteDec*

*00000060 83 EE 01 sub esi,1*

*00000063 E2 EB loop show\_primary*

*00000065 E8 00000000 E call crlf*

*0000006A 33 F6 xor esi,esi ; очистка регистров*

*0000006C 33 D2 xor edx,edx*

*0000006E 33 C9 xor ecx,ecx*

*00000070 33 C0 xor eax,eax*

*00000072 A0 00000014 R mov al,n*

*00000077 8B C8 mov ecx,eax*

*00000079 8B F0 mov esi,eax*

*0000007B 33 C0 xor eax,eax*

*0000007D BA 0000005B R mov edx,OFFSET mes3 ;вывод сообщения mes3*

*00000082 E8 00000000 E call WriteString*

*00000087 E8 00000000 E call ReadDec*

*0000008C A2 000000B0 R mov simv,al*

*00000091 8A 1D 000000B0 R mov bl,simv*

*00000097 66| BF 0000 mov di,0*

*0000009B next:*

*0000009B 33 C0 xor eax,eax*

*0000009D 33 C9 xor ecx,ecx*

*0000009F A0 00000014 R mov al,n*

*000000A4 8A C8 mov cl,al*

*000000A6 66| BE 0000 mov si,0*

*000000AA go1:*

*000000AA 8A 04 35 mov al,mas1[si]*

*00000000 R*

*000000B1 3A C3 cmp al,bl*

*000000B3 74 06 jz udalenie*

*000000B5 66| 46 inc si*

*000000B7 E2 F1 loop go1*

*000000B9 EB 2E jmp vivod*

*000000BB udalenie:*

*000000BB 8A 04 35 mov al,mas1[si+1]*

*00000001 R*

*000000C2 88 04 35 mov mas1[si],al*

*00000000 R*

*000000C9 66| 46 inc si*

*000000CB E2 EE loop udalenie*

*000000CD 66| 47 inc di*

*000000CF A0 00000014 R mov al,n*

*000000D4 2C 01 sub al,1*

*000000D6 A2 00000014 R mov n,al*

*000000DB EB BE jmp next*

*000000DD da:*

*000000DD BA 00000090 R mov edx,offset mes5*

*000000E2 E8 00000000 E call WriteString*

*000000E7 EB 33 jmp sort*

*000000E9 vivod:*

*000000E9 33 F6 xor esi,esi ; очистка регистров*

*000000EB 33 D2 xor edx,edx*

*000000ED 33 C9 xor ecx,ecx*

*000000EF 33 C0 xor eax,eax*

*000000F1 A0 00000014 R mov al,n*

*000000F6 8B C8 mov ecx,eax*

*000000F8 48 dec eax*

*000000F9 66| 8B F0 mov si,ax*

*000000FC show\_primar: ; метка начала цикла*

*000000FC 33 D2 xor edx, edx ; очистка регистра edx*

*000000FE 8A 14 35 mov dl, mas1[si]*

*00000000 R*

*00000105 8A C2 mov al,dl*

*00000107 E8 00000000 E call WriteDec*

*0000010C 83 EE 01 sub esi,1*

*0000010F E2 EB loop show\_primar*

*00000111 E8 00000000 E call crlf*

*00000116 66| 83 FF 00 cmp di,0*

*0000011A 74 C1 jz da*

*0000011C sort:*

*0000011C BE 00000000 mov esi,0*

*00000121 C6 05 000000B1 R mov i,0*

*00*

*00000128 internal:*

*00000128 8A 3D 00000014 R mov bh,n*

*0000012E 88 3D 000000B2 R mov j,bh*

*00000134 BF 00000000 mov edi,0*

*00000139 8A 0D 00000014 R mov cl,n*

*0000013F l1:*

*0000013F 66| 47 inc di*

*00000141 E2 FC loop l1*

*00000143 66| 4F dec di*

*00000145 EB 37 jmp cycl\_j*

*00000147 exchange:*

*00000147 8A 04 35 mov al,mas1[si]*

*00000000 R*

*0000014E 3A 04 3D cmp al,mas1[di]*

*00000000 R*

*00000155 7E 1F jle lesser*

*00000157 A2 000000B3 R mov tmp,al*

*0000015C 8A 04 3D mov al,mas1[di]*

*00000000 R*

*00000163 88 04 35 mov mas1[si],al*

*00000000 R*

*0000016A A0 000000B3 R mov al,tmp*

*0000016F 88 04 3D mov mas1[di],al*

*00000000 R*

*00000176 lesser:*

*00000176 66| 4F dec di*

*00000178 FE 0D 000000B2 R dec j*

*0000017E cycl\_j:*

*0000017E 8A 1D 000000B2 R mov bl,j*

*00000184 3A 1D 000000B1 R cmp bl,i*

*0000018A 7F BB jg exchange*

*0000018C 66| 46 inc si*

*0000018E FE 05 000000B1 R inc i*

*00000194 A0 000000B1 R mov al,i*

*00000199 3A 05 00000014 R cmp al,n*

*0000019F 7C 87 jl internal*

*000001A1 BA 00000074 R mov edx,offset mes4*

*000001A6 E8 00000000 E call WriteString*

*000001AB 8A 0D 00000014 R mov cl,n*

*000001B1 66| BE 0000 mov si,0*

*000001B5 33 F6 xor esi,esi ; очистка регистров*

*000001B7 33 D2 xor edx,edx*

*000001B9 33 C9 xor ecx,ecx*

*000001BB 33 C0 xor eax,eax*

*000001BD A0 00000014 R mov al,n*

*000001C2 8B C8 mov ecx,eax*

*000001C4 66| BE 0000 mov si,0*

*000001C8 show\_primar1: ; метка начала цикла*

*000001C8 33 D2 xor edx, edx ; очистка регистра edx*

*000001CA 8A 14 35 mov dl, mas1[si]*

*00000000 R*

*000001D1 8A C2 mov al,dl*

*000001D3 E8 00000000 E call WriteDec*

*000001D8 83 C6 01 add esi,1*

*000001DB E2 EB loop show\_primar1*

*000001DD E8 00000000 E call crlf*

*000001E2 33 F6 xor esi,esi ; очистка регистров*

*000001E4 33 D2 xor edx,edx*

*000001E6 33 C9 xor ecx,ecx*

*000001E8 33 C0 xor eax,eax*

*000001EA A0 00000014 R mov al,n*

*000001EF 8B C8 mov ecx,eax*

*000001F1 66| BE 0000 mov si,0*

*000001F5 zanos:*

*000001F5 8A 04 35 mov al,mas1[si]*

*00000000 R*

*000001FC 88 04 35 mov mas2[si],al*

*0000000A R*

*00000203 66| 46 inc si*

*00000205 E2 EE loop zanos*

*00000207 33 F6 xor esi,esi ; очистка регистров*

*00000209 33 D2 xor edx,edx*

*0000020B 33 C9 xor ecx,ecx*

*0000020D 33 C0 xor eax,eax*

*0000020F A0 00000014 R mov al,n*

*00000214 8B C8 mov ecx,eax*

*00000216 66| BE 0000 mov si,0*

*0000021A show\_primar2: ; метка начала цикла*

*0000021A 33 D2 xor edx, edx ; очистка регистра dx*

*0000021C 8A 14 35 mov dl, mas1[si]*

*00000000 R*

*00000223 8A C2 mov al,dl*

*00000225 E8 00000000 E call WriteDec*

*0000022A 83 C6 01 add esi,1*

*0000022D E2 EB loop show\_primar2*

*0000022F E8 00000000 E call crlf*

*exit*

*00000234 6A 00 \* push +000000000h*

*00000236 E8 00000000 E \* call ExitProcess*

*0000023B main ENDP*

*END main*

Предполагается последовательность mas1 из n символов (mas=1, 2,…, i,…, k,…, n). Необходимо: а) объединить последовательности от 1 до i и от k до n и занести в mas2; б) последовательность от i до k занести в mas3.

INCLUDE Irvine32.inc

.data ; начало сегмента инициализированных и неинициализированных данных.

;mas1 db 40 DUP(?)

mes1 db 'Vvedite n: ',0

mes2 db 'Vvedite simvoli mas1: ',0

mes3 db 'Vvedite i: ',0

mes4 db 'Vvedite k: ',0

n db ?

mas2 DB 40 DUP(?)

mas3 DB 40 DUP(?)

i db ?

k db ?

mas1 db 81 DUP(?)

.code

main PROC

xor esi,esi

xor eax,eax

mov edx,OFFSET mes2 ;вывод сообщения mas1

call WriteString

xor edx,edx

mov edx,OFFSET mas1 ;вывод сообщения mas1

mov ecx,80

call ReadString

xor ecx,ecx

xor esi,esi

lea esi,mas1

next:

cmp byte ptr [esi],symbol ; сравниваем с символом

jz notfound

inc ecx ; Если это он увеличиваем счетчик

notfound:

inc esi

cmp byte ptr [esi],0 ; если следующий символ 0...

jz o ; ,то это конец строки

jmp next ; если нет, то повторяем.

o:

mov n,cl

xor ecx,ecx

xor eax,eax

mov edx,OFFSET mes3 ;вывод сообщения mas1

call WriteString

call ReadDec

mov i,al

xor eax,eax

mov edx,OFFSET mes4 ;вывод сообщения mas1

call WriteString

call ReadDec

mov k,al

mov edx,OFFSET mas1 ;вывод сообщения mas1

call WriteString

call Crlf

xor eax,eax

xor edi,edi

xor esi,esi

xor ecx,ecx

CLD ;Сброс флага DF.

LEA EDI,mas2 ;Адрес области "куда".

LEA ESI,mas1 ;Адрес области "откуда".

MOV al,i ;Счетчик на i байт.

mov cx, ax

REP MOVSB ;Переслать данные.

xor eax,eax

xor edi,edi

xor esi,esi

xor ecx,ecx

CLD ;Сброс флага DF.

LEA EDI,mas2 ;Адрес области "куда".

LEA ESI,mas1 ;Адрес области "откуда".

MOV al,k

;add al,1

add esi,eax

xor eax,eax

MOV al,i

;add al,1

add edi,eax

xor eax,eax

MOV al,n ;Счетчик на n-k байт.

sub al,k

;add al,1

mov cx, ax

REP MOVSB ;Переслать данные.

xor eax,eax

xor edi,edi

xor esi,esi

xor ecx,ecx

CLD ;Сброс флага DF.

LEA EDI,mas3 ;Адрес области "куда".

LEA ESI,mas1 ;Адрес области "откуда".

MOV al,i

; add al,1

add esi,eax

xor eax,eax

MOV al,k

sub al,i

mov cx, ax ;Счетчик на k-i байт.

REP MOVSB ;Переслать данные.

mov edx,OFFSET mas2 ;вывод сообщения mas3

call WriteString

call Crlf

mov edx,OFFSET mas3 ;вывод сообщения mas2

call WriteString

call Crlf

exit

main ENDP

END main

**Вывод:** В ходе данной лабораторной работы были закреплены навыки создания ассемблерной программы. Были приобретены знания для создания и обработки массивов, структур, объединений и записей.