**Write a program that rearranges the value of doubleword array in reverse order using Xchg.**

|  |
| --- |
| INCLUDE Irvine32.inc  .data  val1 DWORD 10h, 20h, 30h, 40h  str1 Byte "ARRAY BEFORE REVERSING",0AH, 0DH, 0  str2 Byte "ARRAY AFTER REVERSING" ,0AH, 0DH, 0  .code  main PROC  MOV EDX,offset str1  call writeString  MOv eSi, OFFSET val1  MOV ecX, LENGTHOF val1  mov ebx, 4  Call DumpMem  call crlf  MOV EDX, offset str2  call writeString  mov eax, val1  xchg eax, [val1+12]  mov val1,eax  mov eax,[val1+4]  xchg eax, [val1+8]  mov [val1+4], eax  mov esi, OFFSET val1  MOV eCX, LENGTHOF val1  mov ebx,4  call DumpMem  call waitmsg  INVOKE ExitProcess,0  Main ENDP  END main |

**Write a code to implement**

**Xval-(-yval+zval) and store value in Rval**

|  |
| --- |
| INCLUDE Irvine32.inc  ExitProcess PROTO, dwExitCode:DWORD  DumpRegs PROTO  .data  Xval DWORD 15h  yval DWORD 10h  zval DWORD 15h  Rval DWORD ?  .code  main PROC  mov eax, yval  neg eax  add eax,zval  mov ebx, xval  sub ebx, eax  mov Rval, ebx  call dumpRegs  call waitmsg  INVOKE ExitProcess,0  main ENDP  END main |

**Smallest character to larger**

|  |
| --- |
| INCLUDE Irvine32.inc  ExitProcess PROTO, dwExitCode: DWORD  .data  Msg1 BYTE "Enter any character",0AH,0DH,0  Msg2 BYTE "Converted character is",0AH,0DH,0  .code  main PROC  MOV edx, offset Msg1  call writeString  call readchar  call writeChar  call crlf  xor al, 00100000b  MOV edx, offset Msg2  call writeString  call writeChar  call crlf  call waitMsg  INVOKE ExitProcess,0  main ENDP  END main |

**Swap two arrays of type dword using loops Display using dumpmem before and after swap use length, type operator and loops where required.**

|  |
| --- |
| INCLUDE Irvine32.inc  .data  Msg1 BYTE "-----Arrays before swap-----",0AH,0DH,0  Msg2 BYTE "-----Arrays after Swap-----",0AH,0DH,0  Msg3 BYTE "STRINGS ARE NOT EQUAL!!!!!!!!" ,0AH,0DH,0  Msg4 BYTE "Character Array", 0AH, 0DH, 0  Msg5 BYTE "Integer Array", 0AH, 0DH, 0  CharArray BYTE 'A','B','C','D'  intArray BYTE 10h, 20h, 30h, 48h  .code  main PROC  mov ebx, LENGTHOF intArray  mov edx, LENGTHOF CharArray  cmp ebx, edx  jne L1  Je L2  L1:  mov edx, offset Msg1  call WriteString  hlt  L2:  mov edx, offset Msg1  call writeString  call crlf  mov edx, offset Msg5  call WriteString  mov esi, offset intArray  MOV ecX, LENGTHOF intArray  mov ebx, Type intArray  call DumpMem  call crlf  mov edx, offset Msg4  call WriteString  mov ecx, LENGTHOF charArray  mov esi, offset charArray  L3:  mov eax,[esi]  add esi,TYPE charArray  call writeChar  loop L3  mov ecx, LENGTHOF intArray  mov esi,0  L4:  mov al, intarray[esi]  xchg al, charArray[esi]  mov intarray[esi],al  add esi, TYPE intArray  loop L4  call crlf  call crlf  mov edx, offset Msg2  call writeString  call crlf  mov edx, offset Msg5  call WriteString  mov edx, offset intArray  call writeString  call crlf  call crlf  mov edx, offset Msg4  call WriteString  mov esi, offset charArray  mov ecX, LENGTHOF charArray  mov ebx, TYPE charArray  call DumpMem  call crlf  call waitmsg  INVOKE ExitProcess,8  main ENDP  END main |

**Declare array of type byte, word and double word**

**Using type operator, see their type**

**Using lengthOf operator,see their length**

**Using sizeOf operator,see their size**

|  |
| --- |
| INCLUDE Irvine32.inc  .data  ArrayB BYTE 10h, 20h, 30h, 40h  Arrayw WORD 18h, 20h, 38h, 40h  ArrayD DWORD 18h,28h, 30h, 48h  .code  main PROC  mov eax,type ArrayB  mov ebx,lengthof ArrayB  mov ecx,sizeOf ArrayB  call dumpRegs  mov eax, type Arrayw  mov ebx, lengthof Arrayw  mov ecx, sizeof Arrayw  call dumpRegs  mov eax, type ArrayD  mov ebx, lengthof ArrayD  mov ecx, sizeOf ArrayD  call dumpRegs  call waitmsg  INVOKE ExitProcess,8  main ENDP  END main |

**Write a code that copy a string using loops. Use source and target strings.**

|  |
| --- |
| INCLUDE Irvine32.inc  .data  source BYTE "I am Fatima Afzaal.Student of BSCS-3B", 0  target BYTE LENGTHOF source DUP (2)  .code  main PROC  mov ecx, LENGTHOF source  mov esi,0  L1:  mov al, source[esi]  mov target[esi],al  add esi, TYPE source  Loop L1  mov edx, offset target  call writeString  call crlf  call waitmsg  INVOKE ExitProcess,0  main ENDP  END main |

Boolean Calculator (Create a program that functions as a simple Boolean calculator for 32-bit

integers.

The program do the following functions 1x AND y 2. x OR y 3. x XOR y 4.NOT x

|  |
| --- |
| INCLUDE Irvine32.inc  ExitProcess PROTO, dwExitCode:DWORD  .data  X DWORD ?  Y DWORD ?  Msg1 BYTE "x AND y", 0AH, 0DH, 0  Msg2 BYTE "x OR y", 0AH, 0DH, 0  MSg3 BYTE "× XOR y" ,0AH, 0DH, 0  Msg4 BYTE "NOT x",0AH, 0DH, 0  MSg5 BYTE "Enter x",0AH, 0DH, 0  Msg6 BYTE "Enter y", 0AH, 8DH, 0  Msg7 BYTE "Enter operation which you want to perform", 0AH, 0DH,0  BYTE "FOR AND OPERATION---- (PRESS 1 TO SELECT)", 0AH, 0DH  BYTE "FOR OR OPERATION---- (PRESS 2 TO SELECT)", 0AH, 0DH  BYTE "FOR XOR OPERATION---- (PRESS 3 TO SELECT)", 0AH, 0DH  BYTE "FOR NOT OPERATION---- (PRESS 4 TO SELECT)", 0AH, 0DH, 0  .code  main PROC  MOV edx ,offset Msg5  call writeString  call readint  mOV X, eax  MOV edx,offset Msg5  call writestring  call readint  mov y,eax  MOV edx,offset Msg7  call writeString  call readChar  cmp al,'1'  je L1  cmp al,'2'  je L2  cmp al,'3'  Je L3  cmp al,'4'  je L4  L1:  mov eax, X  and eax, y  MOV edx,offset Msg1  call writestring  call writeBin  Jmp L5  L2:  mov eax, X  or eax, y  Mov edx ,offset Msg2  call writeString  call writeBin  jmp L5  L3:  mov eax, X  xor eax, y  Mov edx ,offset Msg3  call writestring  call writeBin  jmp L5  L4:  mov eax, x  not eax  MOV edx,offset Msg4  call writeString  call writeBin  jmp L5  L5:  call crlf  call waitmsg  INVOKE ExitProcess,0  main Endp  END main |