Lab: 11

Department of Computer Science

Iqra University Islamabad

Computer Organization and Assembly Language

Maqsood Ahmed

ID: 38186

5.2 Writing Characters, Strings, and Integers to Standard Output

```
TITLE Writing characters, strings, and integers  (output.asm)

; Testing Following Procedures in the assembly32.lib Library:

; Clrscr:       Clears the console screen
; Crlf:         Write CR and LF characters (end-of-line)
; WriteChar:    Write a single character to standard output
; WriteString:  Write a null-terminated string to standard output
; WriteHex:     Write 32-bit integer in eax in hexadecimal format
; WriteBin:     Write 32-bit integer in eax in binary format
; WriteDec:     Write 32-bit integer in eax in unsigned decimal
format
; WriteInt:     Write 32-bit integer in eax in signed decimal
format

.686
.MODEL flat, stdcall
.STACK

INCLUDE Irvine32.inc

.data

CR  EQU     0Dh         ; carriage return
LF  EQU     0Ah         ; line feed

string  BYTE "Hello World",CR,LF,0
 .code main PROC ;
Clear the screen
    call Clrscr         ; Call procedure Clrscr

; Write a character to standard output     mov  al, 'A'
; al = 'A' (or 41h)     call WriteChar      ; Write
character in register al     call Crlf           ; Write
CR and LF chars (end-of-line)

; Write a null-terminated string to standard output
lea  edx, string    ; load effective address of string into edx
call WriteString    ; write string whose address is in edx

; Write an integer to standard output
mov  eax,0F1A37CBFh ; eax = 0F1A37CBFh
call WriteHex       ; Write eax in hexadecimal format
call Crlf           ; Write CR and LF chars (end-of-line)
```

```
call WriteBin       ; Write eax in binary format     call
Crlf           ; Write CR and LF chars (end-of-line)

call WriteDec       ; Write eax in unsigned decimal format
call Crlf           ; Write CR and LF chars (end-of-line)


call WriteInt       ; Write eax in signed decimal format
call Crlf           ; Write CR and LF chars (end-of-line)
exit
main ENDP
END main
```

5.2.1 Lab Work: Assemble and Link Output.asm 5.2.2 Lab Work: Trace the Execution of Output.exe

Guess the Console Output of the above program and write it in the specified box.

OUTPUT:

```
A
Hello World
0F1A37CBF
11110001101000011101111001111111
4048873599
-1089096913
```

The following program demonstrates the use of the above procedures:

```
TITLE Setting Text Color, Dumping Memory and Registers  (Output2.asm)

; Testing following Output Procedures in the assembly32.lib
Library:

; Clrscr:       Clears the console screen
; SetTextColor: Set the foreground and background colors of text
; DumpMem:      Write a block of memory in hexadecimal
; DumpRegs:     Display basic registers and flags in hexadecimal
; WaitMsg:      Display a message and wait for Enter key to be
pressed
; Gotoxy:       Put the cursor at a specific row/column on the
console

.686
.MODEL flat, stdcall
.STACK
```

```
INCLUDE Irvine32.inc

.data
CR  EQU     0Dh                  ; carriage return
LF  EQU     0Ah                  ; line feed

string   BYTE    "This is a string", CR, LF, 0
 .code
main PROC
; Clear the screen after setting text color
    mov  eax, yellow+(blue*16)  ; yellow = 14 and blue = 1
call SetTextColor           ; set yellow text on blue background
call Clrscr                 ; Call procedure Clrscr

; Call DumpMem that display a block of memory to standard output
mov  esi, OFFSET string     ; esi = address of memory block
mov  ecx, LENGTHOF string   ; ecx = number of elements to display
mov  ebx, TYPE BYTE         ; ebx = type of each element     call
DumpMem                ; write 19 bytes of string

; Call WaitMsg that displays "Press [Enter] to continue ..."
call WaitMsg                ; wait for [Enter] key to be pressed

; Call DumpRegs that display the basic registers and flags in hex
call DumpRegs               ; write basic registers

; Call WaitMsg after locating the cursor on the
console
mov  dh, 10                 ; row 10     mov  dl, 20
; column 20     call Gotoxy                ; locate
cursor
call WaitMsg                ; wait for [Enter] key to be pressed
exit

main ENDP
END main
```

5.3.1 Lab Work: Assemble and Link Output2.asm 5.3.2 Lab Work: Trace the Execution of Output2.exe

OUTPUT:

```
Press [Enter] to continue ...
EAX=00000000 EBX=00000000 ECX=00000013 EDX=77FC0FAF
ESI=0028FD00 EDI=0028FCFC EBP=0028FCF4 ESP=0028FCEC
EIP=004010E7 EFL=00000206 yellow foreground, blue background
Row 10, Column 20
```

The following program demonstrates the use of the above procedures:

```
TITLE Reading characters, strings, and integers
(input.asm) ; Testing Following Procedures in the
assembly32.lib Library:
; ReadChar:     Read a single character from standard input
; ReadString:   Read a null-terminated string from standard input
; ReadHex:      Read hexadecimal integer from standard input
; ReadInt:      Read signed decimal integer from standard input

.686
.MODEL flat, stdcall
.STACK
INCLUDE Irvine32.inc
 .data
charvar     BYTE    0           ; Character variable string
BYTE    21 DUP(0)   ; Extra byte for null char bytecount
DWORD   0           ; Count of bytes read in string hexvar
DWORD   0           ; Unsigned integer variable intvar
SDWORD  0           ; Signed integer variable

prompt1     BYTE    "Enter a character (char will not appear): ",0
prompt2     BYTE    "Enter a string (max 20 chars): ",0 prompt3
BYTE    "Enter a hexadecimal number (max 8 digits): ",0 prompt4
BYTE    "Enter a decimal number with optional sign: ",0
 .code main PROC
call Clrscr

;   Display   prompt1
lea     edx,   prompt1
call WriteString
; Read a character (without echo) from standard input
call ReadChar           ; character is returned in AL
mov  charvar, al        ; save character in charvar
    call Crlf               ; Write end-of-line after reading character

;   Display   prompt2
lea     edx,   prompt2
call WriteString
; Read a null-terminated string from standard input
lea  edx, string        ; edx = address of storage area for string
mov  ecx, SIZEOF string ; ecx = max characters to be stored
call ReadString         ; read string from standard input     mov
bytecount, eax     ; eax = actual number of chars read
```

```
;   Display   prompt3
lea    edx,    prompt3
call WriteString
; Read a hexadecimal string and convert it to a number
call ReadHex            ; number is returned in EAX register
mov  hexvar, eax        ; save number in hexvar

;   Display   prompt4
lea     edx,   prompt4
call WriteString
; Read a signed decimal string and convert it to a number
call ReadInt            ; number is returned in EAX register
mov  intvar, eax        ; save number in intvar
exit
main ENDP
END main
```

5.4.1 Lab Work: Assemble and Link Input.asm 5.4.2 Lab Work: Trace the Execution of Input.exe

OUTPUT:

1. `charvar` **(in hex and as a character) = 0x41 (character 'A')**
2. **12 bytes of** `string` **(in hex) = 4D 79 20 53 74 72 69 6E 67 00 00 00**
3. `bytecount` **(in decimal) = 9**
4. `hexvar` **(in hex) = AB09F**
5. `intvar` **(in decimal) = -12345678**