**Lab Session 06**

**Procedures**



***CALL* Instruction**

The call instruction is used to call a procedure.

**Procedures in Irvine32 Library**

* **Clrscr**

Clears the console window and locates the cursor at the above left corner.

* **Crlf**

Writes the end of line sequence to the console window.

* **Delay**

Pauses the program execution for a specified interval (in milliseconds).

* **DumpRegs**

Displays the EAX, EBX, ECX, EDX, ESI, EDI, ESP:EIP and EFLAG registers.

* **DumpMem**

Writes the block of memory to the console window in hexadecimal.

* **getDateTime**

Gets the current date and time from system

* **GetMaxXY**

Gets the number of columns and rows in the console window buffer.

* **GetTextColor**

Returns the active foreground and background text colors in the console window.

* **Gotoxy**

Locates the cursor at a specific row and column in the console window. By default X coordinate range is 0-79 and Y coordinate range is 0-24.

* **MsgBox**

Displays a pop-up message box.

* **MsgBoxAsk**

Displays a yes/no question in a pop-up message box.

* **ReadChar**

Waits for single character to be typed at the keyboard and returns that character.

* **ReadDec**

Reads an unsigned 32-bit integer from the keyboard.

* **ReadHex**

Reads a 32-bit hexadecimal integers from the keyboard, terminated by the enter key.

* **ReadInt**

Reads a signed 32-bit integer from the keyboard, terminated by the enter key.

* **ReadString**

Reads a string from the keyboard, terminated by the enter key.

* **SetTextColor**

Sets the foreground and background colors of all subsequent text output to the console.

* **WriteBin**

Writes an unsigned 32-bit integer to the console window in ASCII binary format.

* **WriteChar**

Writes a single character to the console window.

* **WriteDec**

Writes an unsigned 32-bit integer to the console window in decimal format.

* **WriteHex**

Writes a 32-bit integer to the console window in hexadecimal format.

* **WriteInt**

Writes a signed 32-bit integer to the console window in decimal format.

* **WriteString**

Write a null-terminated string to the console window.

* **Randomize**

Seeds the random number generator with a unique value.

* **WaitMsg**

Display a message and wait for the Enter key to be pressed.

**EXAMPLE # 01:**

**WriteDec:** The integer to be displayed is passed in EAX **WriteString:** The offset of string to be written is passed in EDX **WriteChar:** The character to be displayed is passed in AL

.data

divider BYTE " - ", 0

codepage DWORD 1252

.code

mov ecx, 255

mov eax,1

mov edx, OFFSET divider

L1:

call WriteDec ; EAX is a counter

call WriteString ; EDX points to string

call WriteChar ; AL is the character

call Crlf

inc al ; next character

Loop L1

**EXAMPLE # 02:**

**SetTextColor:** Background & foreground colors are passed to EAX

.data

str1 BYTE "Sample string in color", 0dh, 0ah, 0

.code

mov eax, yellow + (blue \* 16)

call SetTextColor

mov edx, OFFSET str1

call WriteString

call DumpRegs

exit

**EXAMPLE # 03:**

**MsgBox:** Offset of message to be displayed inside the pop-up is passed in EDX. Offset of caption (optional) is passed in EBX.

.data

caption BYTE "Dialog Title", 0

HelloMsg BYTE "This is a pop-up message box.", 0dh,0ah

BYTE "Click OK to continue...", 0

.code

mov ebx, 0 ; no caption

mov edx, OFFSET HelloMsg ; contents

call MsgBox

mov ebx, OFFSET caption ; caption

mov edx, OFFSET HelloMsg ; contents

call MsgBox

**EXAMPLE # 04:**

**DumpMem:** Pass offset of array in ESI, length of array in ECX & type in EBX **ReadInt:** Reads the signed integer into EAX **WriteInt:** Signed integer to be written is passed in EAX **WriteHex:** Hex value to be written is passed in EAX

**WriteBin:** Binary value to be written is passed in EAX

.data

COUNT = 4

BlueTextOnGray = blue + (lightGray \* 16)

DefaultColor = lightGray + (black \* 16)

arrayD SDWORD 12345678h, 1A4B2000h, 3434h, 7AB9h

prompt BYTE "Enter a 32-bit signed integer: ", 0

.code

; Set text color to blue text on a light gray background

mov eax, BlueTextOnGray

call SetTextColor

call Clrscr ; clear the screen

; Display an array using DumpMem.

mov esi, OFFSET arrayD ; starting OFFSET

mov ebx, TYPE arrayD ; doubleword = 4 bytes

mov ecx, LENGTHOF arrayD ; number of units in arrayD

call DumpMem ; display memory

; Ask the user to input a sequence of signed integers

call Crlf ; new line

mov ecx, COUNT

L1:

mov edx, OFFSET prompt

call WriteString

call ReadInt ; input integer into EAX

call Crlf ; new line

; Display the integer in decimal, hexadecimal, and binary

call WriteInt ; display in signed decimal

call Crlf

call WriteHex ; display in hexadecimal

call Crlf

call WriteBin ; display in binary

call Crlf

call Crlf

Loop L1 ; repeat the loop

; Return console window to default colors.

call WaitMsg ; "Press any key..."

mov eax, DefaultColor

call SetTextColor

call Clrscr

**EXAMPLE # 05:**

**MsgBoxAsk:** Offset of question string is passed in EDX. Offset of caption is passed in EBX. Selected value is returned in EAX (IDYES equal to 6 or IDNO equal to 7)

.data

caption BYTE "Survey Completed",0

question BYTE "Thank you for completing the survey."

BYTE 0dh, 0ah

BYTE "Would you like to receive the results?", 0

.code

mov ebx, OFFSET caption

mov edx, OFFSET question

call MsgBoxAsk

;(check return value in EAX)

**EXAMPLE # 06:**

**GetMSeconds:** Value is returned in EAX

.data

startTime DWORD ?

.code

call GetMseconds

mov startTime, eax

L1:

; (loop body)

loop L1

call GetMseconds

sub eax, startTime

**Creating a New File**

EAX contains the newly created file’s handle or *INVALID\_HANDLE\_VALUE* if creation is unsuccessful

**EXAMPLE:**

.data  
 filehandle DWORD ?  
 filename BYTE “MyFile.txt”, 0

.code  
 mov edx, offset filename  
 call CreateOutputFile  
 mov filehandle, eax

**Opening an Existing File**

Offset of file name is passed to EDX. Handle of opened file is returned in EAX

**EXAMPLE:**

.data  
 filehandle DWORD ?  
 filename BYTE “MyExistingFile.txt”, 0

.code  
 mov edx,OFFSET filename  
 call OpenInputFile  
 mov filehandle, EAX

**Reading From a File**

**Call arguments:**   
 EAX = an open file handle  
 EDX = offset of the input buffer  
 ECX = maximum number of bytes to read

**Return arguments:**  
 If CF = 0, EAX contains the number of bytes read.  
 If CF = 1, EAX contains a system error code

**EXAMPLE:**

.data  
 buffSize = 10 ; if we want to read just 10 bytes  
 buffer BYTE buffSize DUP(?) ; buffer will contain the text read from the file

.code  
 mov eax, filehandle ;assuming filehandle contains handle of an open file  
 mov edx, OFFSET buffer ;buffer will contain the text read from the file  
 mov ecx, BUFSIZE ;specify how many bytes to read  
 call ReadFromFile

**Writing To a File:**

**Call arguments:**   
 EAX = an open file handle  
 EDX = offset of the buffer  
 ECX = maximum number of bytes to write

**Return arguments:**

If CF = 0, EAX contains the number of bytes written.   
 If CF = 1, EAX contains a system error code.

**EXAMPLE:**

.data  
 bufferSize = 10 ;if we want to write just 10 bytes  
 buffer BYTE bufferSize DUP(?) ;uninitialized in this example but buffer will contain the text to be written to file

.code  
 mov eax, filehandle ; assuming that filehandle contains handle of an open file  
 mov edx, OFFSET buffer ;buffer from where text will be written to file  
 mov ecx, bufferSize ;number of bytes to be written to file from the buffer  
 call WriteToFile

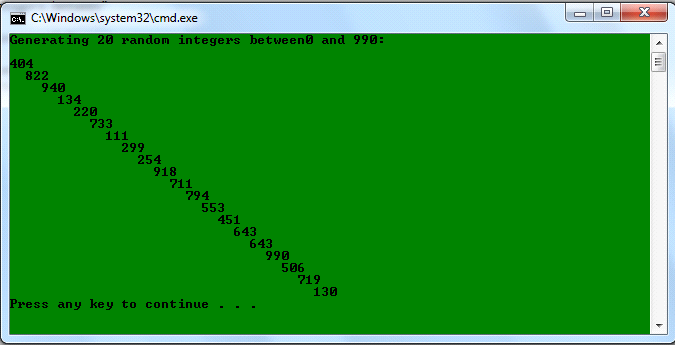
**Closing a File**

**EXAMPLE:**

mov eax, filehandle ;assuming filehandle contains handle of an open file  
 call CloseFile

**ACTIVITIES:**

1. Write a program to display a list of 20 random numbers in diagonal pattern. Add a 5 millisecond delay before displaying each number.



2. Write a program that takes an input from the user and displays it in decimal, hexadecimal and binary format. Also display all register values.

3. Write a program to take input data for an employee and store it in appropriate variables. The program should ask for Employee ID, Name, Year of Birth & Annual Salary from the user. The program should then calculate the annual tax on that employee’s annual salary if it exceeds Rs. 50,000 and display the tax in a message box. The tax is calculated according to formula:

*Tax = Monthly Salary / 2*

4. Make a program to create a text file name Fibo.txt and write the first 8 fibonnaci numbers to that file.

5. Print the following pattern (using GotoXY and any other library procedure) without using the “Space" character.

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*