Name Saeed Ahmed Asad

ID 14273

Semester BS CS 5th

Assignment 05

Subject

Microprocessor and Assembly language

Scanned with CamScanner

Assignment 05

Question 1

Ans:- Extended stack pointer

Question 2

Ans:- It is called LIFO structure because it contains set of memory blocks, in which data is retrieved in order.
i.e The last value pushed into the stack is the first value popped out from the stack.

Question 3

Ans:- ESP is Decremented by 4

Question 4:

Ans:- Execution would continue beyond the end of the procedure, possibly into the beginning of another procedure

Question 5

Ans:- A list of input parameter and their usage, labeled by a

Scanned with CamScanner

word such as receivers. A description of any values returned by the procedure, labeled by a word such as returns.

Question 6

Ans:- Random Range procedure.

Question 7

Ans:- Wait msg procedure

Question 8

Ans:- Code example mov eax, 700
call Delay

Question 9

Ans:- WriteDec procedure

Question 10

Ans:- Gotoxy

Question 11

Ans:- Imagine two possible way of calling the DumpMem procedure.

Scanned with CamScanner

Require input parameters

Pushed
mov esi, OFFSET array
mov ecx, LENGTHOF array
mov ebx, TYPE array
call DumpMem
poped

Question 12

Ans:- EDX contain the offset of an array of bytes, and ECX contain the maximum number of character to read.

Question 13

Ans:- Random Range

Question 14

Ans:- Final value in eax after these instruction execution. (5)

Question 15

Ans:- This one I would believe it would equal 10 because of LIFO (last in first out). EAX

Scanned with CamScanner

would equal 10 because it was the last one in.

Question 16

Ans:- By looking at this, I think that EAX would still equal 30 at the end of line 6 because eax was just pushed on the stack not a change in value.

Question 17

Ans:- EAX will be equal 30 on line 6.

Question 18

Ans:- A. EDX will equal 40 on line 6.

Question 19

Ans:-

```
push ebx
push eax
pop ebx
pop eax
```

Scanned with CamScanner

A sequence of statement are
push ebx ; Assume EBX = X and
EAX = Y, here the content of
EBX (i.e x) is pushed push
eax, which is assumed to be
Y pop ebx; Y from stack is
unsigned to EBX, therefore EBX=Y
pop eax; x from stack is assigned
to EAX therefore EAX = X.

Question 21

Ans:-

program:-

```
; Random strings.
INCLUDE irvine32.inc
TAB = 9                ;ASCII code for tab
strlen = 10            ;length of string

.386
.model flat stdcall
.stack 4096
Exit process PROTO, dw Exit code:DWORD
.data
Str1 BYTE "The 20 random string
                are :", 0
```

Scanned with CamScanner

```
.Code
main PROC
mov edx, OFFSET str1 ; "The 20 random
                       strings are:"
Call WriteString ; writes string
Call crlf        ; writes an end-of-line
                   sequence to the console
                   window

mov ecx, 20 ; Create 20 strings.

L1: mov edx, OFFSET arr1
mov eax, strLen ; EAX: String length
Call RandomString ; EAX: String length
Call Display      ; generates the random string

mov al, TAB
Call WriteChar
exit
main ENDP   ; leave a tab space
RandomString PROC USES eax esi
mov ecx, eax ; ECX = String length

L1: mov eax, 26
```

Scanned with CamScanner

```
    call RandomRange
    add eax, 65 ; EAX gets ASCII of
                  a capital letter.
mov arr1[esi], eax
inc esi
loop L1
RandomString ENDP
Display PROC USES eax esi; Display
                  the generate random-
                  strings.
mov ecx, eax ; ECX = String length.

L1: mov eax, arr1[esi]; EAX = ASCII
                  value
call WriteChar ; Write the letter
inc esi
loop L1
Display ENDP
Call dumpregs
INVOKE ExitProcess, 0
END main
```

Scanned with CamScanner

Question 22

Ans:- Title Random characters
(Source.cpp) // This program display a single character at 100 random screen locations.

```
include Irvine32.inc

.data
row WORD ? ; // rows variable to hold
                num of rows.
cols WORD ? ; // cols variable to hold
                num of colums.
.code
main PROC
Call clrscr ; // set courson top left
mov eax, 100 ; // loop for 100 times

L1: Call GetMaxXY ; // Size console window
    mov rows, ax ; // return rows
    mov cols, dx ; // return column
    movzx eax, rows ; // moving rows eax
    Call RandomRange ; // generate integer
    mov dh, al ; // setting range boundaries
    movzx eax cols ; // moving column to eax
    Call RandomRange ; // generates integers
    mov dl, al ; // setting rang boundaries
```

Scanned with CamScanner

```
Call Gotoxy ;// Cursor allocation
Call Writechar ;// Write random character
mov eax, 100  ;// time 100 milliseconds
Call delay  ; //pauses program, 100 milliseconds

Loop L1  ; looping
exit
main ENDP
END main
```

Question 23

Ans:-

```
; Color Matrix
; Write a program that displays
a single characters in all
possible
; Combination of foreground and
background color (16x16 = 256)
; Color are numbered from 0 to 15
So you can use a nested loop
to generate all possible
combinations.
; Last update:
INCLUDE Irvine32.inc
CR = 0DH  ; carriage return
```

Scanned with CamScanner

```
LF = 0Ah  ; line feed

• data
Prompt 1 BYTE "Please type a
character :", 0
dispchr DWORD ?

• Code
main PROC
; Set text color to white text
on black background.
; even though these are the
default defined in Irvine32.inc

mov eax, white + (black 16)
Call SetTextcolor
Call clrscr    ; clear the screen
; Get the user to type some
character for our display:
mov edx, OFFSET prompt1; please
                        type a .....
Call writeString
Call Readchar
```

Scanned with CamScanner

```
;One now has the desired character
in AL,
;but we can only used it
letter, So we save it in
variable dispchar:

mov dispchar, eax
Call crlf    ; new line

; Generating all the color combination
In a nested loop require a bit
more worth than simply a single
loop.
Exc will be preserved on the
stack across the inner loop:

mov ecx ; olter loop counter
L1 :- push ecx; save  "   "

;As the text attributes above
illustrate to set colors we need
to Compute EAx in a way
that combine both attributes
Set EAX using the steck copy
of EAx slowest (with the outerloop
```

Scanned with CamScanner

counter).

```
mov ecx, 16 ; inner loop counter
L2: pop eax ; get the outer loop
              counter in EAX
push eax ; get the outer loop
           counter in EAX
push eax ; saves it back again
sub eax, 1 ; shift range from 1-16
             to 0-15

Vol eax, 4 ; Sets Al for background
Vol eax, ecx ; add the inner loop
               counter of Ecx

Sub eax, 1 ; Set AL for the
             foreground color
Call SetTextcolor
; EAx is now available for
restoring the display character.
mov eax, dispch
Call write char
loop L2
; Return the cursor to beginning
  of the next row for matrix
```

Scanned with CamScanner

```
Call CrlF ; new line
pop ecx ; restore outerloop counter

loop L1
; Show the result on screen
until user hit enter, then exit

mov eax, white + (black * 16)
Call SetTextColor ; otherwise we
                    leave black on black

Call CrlF ; new line
Call waitMsg ; "press [Enter]..."

exit
main ENDP
END main
```

Scanned with CamScanner