Solution

Assembly Quiz-3(G)

Total Marks: 10

Time Allowed: 15 min

Q1. [3 marks]

Assume that a function FUNC is called as below:

```
PUSH AX; parameters
PUSH BX
PUSH WORD [BX]
CALL FUNC
...
```

And the function is implemented as follows:

```
FUNC:
PUSH BP
MOV BP, SP
SUB SP,4
PUSH CX
PUSH DX
...
```

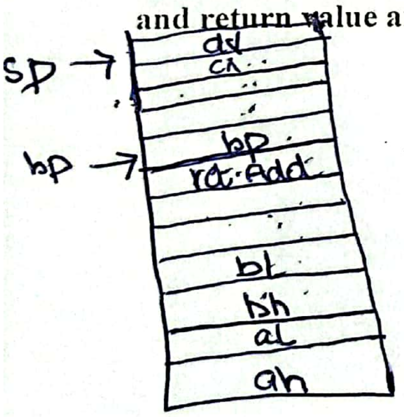
Answer the following questions:

a. The value of AH is stored at **[BP +**_8_**]**

b. The value of CX is stored at **[BP -**_6_**]**

c. The FUNC clears all parameters off the stack so it uses **RET** _6_

Q2. Write a subroutine named copyColumn that copies the 0th column on the display screen to the last (79th) column of the screen. The subroutine does not have any parameter and return value and it should not destroy any register values. [7 marks]



```
2. copyColumn: push bp
               mov bp, sp
               pusha
               mov ax, 0xB800
               mov es, ax
               mov cx, 25
               mov di, 0
         loop1: mov si, di
               add si, 158
               mov ax, [es:di]
               mov [es:si], ax
               add di, 160
               dec cx
               cmp cx, 0
               jne loop1
               popa
               pop bp
               ret
```

Solution

Assembly Quiz-3(G)

Total Marks: 10

Time Allowed: 15 min

Q1. [3 marks]

Assume that a function FUNC is called as below:

```
PUSH AX; parameters
PUSH BX
PUSH WORD [BX]
CALL FUNC
...
```

And the function is implemented as follows:

```
FUNC:
PUSH BP
MOV BP, SP
SUB SP,2
PUSH CX
PUSH DX
...
```

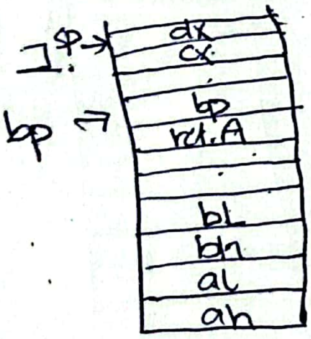
Answer the following questions:

a. The value of AL is stored at **[BP +** 7 **]**

b. The value of CX is stored at **[BP -** 4 **]**

c. The FUNC clears all parameters off the stack so it uses **RET** 6

Q2. Write a subroutine named copyColumn that copies the 0th column on the display screen to the last (79th) column of the screen. The subroutine does not have any parameter and return value and it should not destroy any register values. [7 marks]



2.

```
copyColumn: push bp
            mov bp, sp
            pusha
            mov ax, 0xB800
            mov es, ax
            mov cx, 25
            mov di, 0
loop1:      mov si, di
            add si, 158
            mov ax, [es:di]
            mov [es:si], ax
            add di, 160
            dec cx
            cmp cx, 0
            jne loop1
            popa
            pop bp
            ret
```

Solution

Assembly Quiz-3(H)

Total Marks: 10

Time Allowed: 15 min

Q1. The character 'a' (ASCII 0x61) will be displayed at which row and column of the screen by the following code? Start the column and row number from 0. Show complete working. [3 Marks]

```
mov ax, 0xb800
mov es, ax
mov di, 648
mov word [es:di],0x0761
```

Row: 4

Column: ~~4~~ 0

Q2. Given an integer n as parameter, write a subroutine named showPowerBinary that computes 2^n and displays the number as binary on the screen. For example, if n =4, your subroutine will compute the number $2^4 = 16$ and displays 10000 on the screen. [7 marks]

ASCII of 0: 0x30
ASCII of 1: 0x31

```
ShowPowerBinary: push bp
                 mov bp, sp
                 pusha
                 mov cx, [bp+4]
                 mov ax, 0xB800
                 mov es, ax
                 mov di, 0
                 mov word [es:di], 0x0731
          loop1: add di, 2
                 mov word [es:di], 0x0730
                 dec cx
                 cmp cx, 0
                 jne loop1
                 popa
                 pop bp
                 ret 2
```

Solution

Assembly Quiz-3(H)

Total Marks: 10

Time Allowed: 15 min

Q1. The character 'a' (ASCII 0x61) will be displayed at which row and column of the screen by the following code? Start the column and row number from 0. Show complete working. [3 Marks]

```
mov ax, 0xb800
mov es, ax
mov di, 802
mov word [es:di],0x0761
```

Row: ✓ 5

Column: 1

Q2. Given an integer n as parameter, write a subroutine named showPowerBinary that computes 2^n and displays the number as binary on the screen. For example, if n =4, your subroutine will compute the number $2^4 = 16$ and displays 10000 on the screen. [7 marks]

ASCII of 0: 0x30
ASCII of 1: 0x31

~~ShowPowerBinary: mov ax, 0xB800~~
~~mov~~

1

3
2
1
c

```
showPowerBinary: push bp
                 mov bp, sp
                 pusha
                 mov cx, [bp+4]
                 mov ax, 0xB800
                 mov es, ax
                 mov di,0
                 mov word [es:di], 0x0731
          loop1: add di, 2
                 mov word [es:di], 0x0730
                 dec cx
                 cmp cx, 0
                 jne loop1
                 popa
                 pop bp
                 ret 2
```