Question No 1: Write an Assembly code that will print A-Z using loop.

Code:

```
.model small
 .data
 .code
 main proc
    mov cl,26
    mov dl,'A'
    L1:
    mov ah,2
    int 21h

    add dl,1
    Loop L1

    Exit:
    mov ah,4ch
    main endp
 end main
```

Explanation:

1) .model small
These lines define the memory model (small) and indicate that a data section will follow.

2) .data
The '.data' section is where you would define your data variables.

3) .code
    main proc
Switches to the code section and defines the entry point of the program named main.

4) mov cl, 26
This instruction moves the value 26 into the 'cl' register. The 'cl' register is often used as a counter in loop operations.

5) mov dl, 'A'
This instruction moves the ASCII value of the character 'A' into the 'dl' register. 'dl' is typically used as a data register.

6) L1:
This is a label named L1. Labels are used as targets for jumps and are often used to mark the beginning or end of loops.

7) mov ah,2
   int 21h
mov ah,2: Moves the value 2 into the ‘ah’ register, indicating that an output operation is to be performed.
int 21h: Triggers an interrupt 21h, which is interrupt for displaying a character. It uses the value in the ‘dl’ register as the character to display.

8) add dl, 1
This instruction increments the value in the ‘dl’ register by 1.

9) Loop L1
This is an unconditional loop instruction that jumps back to the label L1 as long as the value in the cl register is not zero. It effectively creates a loop that displays characters starting from 'A' and going up to 'Z' (26 characters in total).

10) Exit:
    mov ah,4ch
    int 21h
This section sets up the program exit. The Exit label signifies the end of the program.

mov ah,4ch: Moves the value 4ch (hex) into the ‘ah’ register, indicating a program termination with a return code.

int 21h: Triggers an interrupt 21h to terminate the program.

11) main endp
    end main
Ends the ‘main’ procedure and marks the end of the program.

Question No 2: **Write a Assembly code that will take three input from user and find the smallest number among these three inputs.**

Code:

```
.model small
.data
.code
main proc
    mov ah, 1
    int 21h
    mov bl, al
    mov dl, 13
    mov ah, 2
    int 21h
    mov dl, 10
    mov ah, 2
    int 21h

    mov ah, 1
    int 21h
    mov bh, al

    mov dl, 13
    mov ah, 2
    int 21h
    mov dl, 10
    mov ah, 2
    int 21h

    mov ah, 1
    int 21h
    mov cl, al

    mov dl, 13
    mov ah, 2
    int 21h
    mov dl, 10
    mov ah, 2
    int 21h

    cmp bl, bh
    jng L1
```

```
    cmp bh, cl
    jg L2

    mov dl, bh
    mov ah, 2
    int 21h
    jmp Exit

    L1:
    cmp bl, cl
    jg L2
    mov dl, bl
    mov ah, 2
    int 21h
    jmp Exit

    L2:
    mov dl, cl
    mov ah, 2
    int 21h
    jmp Exit

    Exit:
    mov ah, 4ch
    int 21h
    main endp
end main
```

Explanation:

1) .model small
These lines define the memory model (small) and indicate that a data section will follow.

2) .data
The '.data' section is where you would define your data variables.

3) .code
    main proc
Switches to the code section and defines the entry point of the program named main.

4) mov ah, 1
   int 21h
   mov bl, al

This sequence captures a single character from the standard input and stores it in the ‘bl’ register.

5) mov dl, 13
  mov ah, 2
  int 21h
  mov dl, 10
  mov ah, 2
  int 21h
These lines output a carriage return (13) and a line feed (10) to the standard output, creating a newline in the console.

6) mov ah, 1
   int 21h
   mov bh, al
Similar to the first input, this captures a second single character from the standard input and stores it in the ‘bh’ register.

7) mov dl, 13
  mov ah, 2
  int 21h
  mov dl, 10
  mov ah, 2
  int 21h
These lines output a carriage return (13) and a line feed (10) to the standard output, creating a newline in the console.

8) mov ah, 1
   int 21h
   mov cl, al
Captures a third single character from the standard input and stores it in the ‘cl’ register.

9) mov dl, 13
  mov ah, 2
  int 21h
  mov dl, 10
  mov ah, 2
  int 21h
These lines output a carriage return (13) and a line feed (10) to the standard output, creating a newline in the console.

10) cmp bl, bh
    jng L1
Compares ‘bl’ and ‘bh’ (the first and second inputs). If ‘bl’ is not greater than ‘bh’ or they are equal, it jumps to the label L1.

11) cmp bh, cl
    jg L2
Compares ‘bh’ and ‘cl’ (the second and third inputs). If ‘bh’ is greater than ‘cl’, it jumps to the label L2.

12) mov dl, bh
    mov ah, 2
    int 21h
   jmp Exit
If neither of the jumps in the previous comparisons is taken, it means ‘bh’ is the middle value. This block outputs the character in ‘bh’ and jumps to the Exit label.

13)L1:
Label, used for branching.

14)cmp bl,cl
   Jg L2
Similar to line 11, but for the case when ‘bl’ is greater than ‘cl’.

15) mov dl, bl
    mov ah, 2
    int 21h
    jmp Exit
Outputs the character in ‘bl’ and jumps to the ‘Exit’ label.

16) L2:
Label, used for branching.

17) mov dl, cl
    mov ah, 2
    int 21h
   jmp Exit
Outputs the character in ‘cl’ and jumps to the ‘Exit’ label.

18) Exit:
    mov ah,4ch
    int 21h
This section sets up the program exit. The Exit label signifies the end of the program.

mov ah,4ch: Moves the value 4ch (hex) into the ‘ah’ register, indicating a program termination with a return code.

int 21h: Triggers an interrupt 21h to terminate the program.

19) main endp
    end main
Ends the ‘main’ procedure and marks the end of the program.

<u>Question No 3:</u> Write a Assembly code that will print “Welcome” if ypu press 1,will print “Today is your lab exam” if you press 2 and will print “Best of luck!” if you press 3.

<u>Code:</u>

```
.model small
.data
msg1 db "Welcome$"
msg2 db "Today is your lab exam$"
msg3 db "Best of luck!$"

.code
main proc
   mov ax, @data
   mov ds, ax

   mov ah, 1
   int 21h
   mov bl, al

   mov dl, 13
   mov ah, 2
   int 21h
   mov dl, 10
   mov ah,2
   int 21h

   cmp bl, "1"
   je L1

   cmp bl, "2"
   je L2
```

```
    cmp bl, "3"
    je L3

    L1:
    lea dx,msg1
    mov ah, 9
    int 21h
    jmp Exit

    L2:
    lea dx,msg2
    mov ah, 9
    int 21h
    jmp Exit

    L3:
    lea dx,msg3
    mov ah, 9
    int 21h
    jmp Exit

     Exit:
     mov ah, 4ch
     int 21h
     main endp
end main
```

Explanation:

1) .model small

These lines define the memory model (small) and indicate that a data section will follow.

2) .data

The '.data' section is where you would define your data variables.

3) msg1 db "Welcome$"
   msg2 db "Today is your lab exam$"
   msg3 db "Best of luck!$"

These lines define three null-terminated strings (msg1, msg2, and msg3) to be displayed based on the user's input.

4) .code
   main proc
Switches to the code section and defines the entry point of the program named main.

5) mov ax, @data
   mov ds, ax
Sets up the data segment register ‘ds’ with the address of the data section.

6) mov ah, 1
   int 21h
   mov bl, al
This sequence captures a single character from the standard input and stores it in the ‘bl’ register.

7) mov dl, 13
  mov ah, 2
  int 21h
  mov dl, 10
  mov ah, 2
  int 21h
These lines output a carriage return (13) and a line feed (10) to the standard output, creating a newline in the console.

8) cmp bl, "1"
   je L1
Compares the value in ‘bl’ with the ASCII value of the character '1'. If they are equal, jumps to the label L1.

9) cmp bl, "2"
   je L2
Compares the value in ‘bl’ with the ASCII value of the character '2'. If they are equal, jumps to the label L2.

10)cmp bl, "3"
    je L3
Compares the value in ‘bl’ with the ASCII value of the character '3'. If they are equal, jumps to the label L3.

11) L1:
```
    lea dx, msg1
    mov ah, 9
    int 21h
    jmp Exit
```
If the user entered '1', this block loads the address of ‘msg1’ into ‘dx’ (using the ‘lea’ instruction), then displays the message using DOS interrupt 21h, function 9 (‘int 21h’), and jumps to the ‘Exit’ label.

12) L2:
```
    lea dx, msg2
    mov ah, 9
    int 21h
    jmp Exit
```
If the user entered ‘2’, this block loads the address of ‘msg2’ into ‘dx’ (using the ‘lea’ instruction), then displays the message using DOS interrupt 21h, function 9 (‘int 21h’), and jumps to the ‘Exit’ label.

13) L3:
```
    lea dx, msg3
    mov ah, 9
    int 21h
    jmp Exit
```
If the user entered ‘3’, this block loads the address of ‘msg3’ into ‘dx’ (using the ‘lea’ instruction), then displays the message using DOS interrupt 21h, function 9 (‘int 21h’), and jumps to the ‘Exit’ label.

14) Exit:
```
    mov ah,4ch
    int 21h
```
This section sets up the program exit. The Exit label signifies the end of the program.

mov ah,4ch: Moves the value 4ch (hex) into the ‘ah’ register, indicating a program termination with a return code.

int 21h: Triggers an interrupt 21h to terminate the program.

15) main endp
```
    end main
```
Ends the ‘main’ procedure and marks the end of the program.