Q1. Program to Print Hello world.

A1.

org 100h

mov ax, 3 ; text mode 80x25, 16 colors, 8 pages (ah=0, al=3)

int 10h ; do it!

; cancel blinking and enable all 16 colors:

mov ax, 1003h

mov bx, 0

int 10h

; set segment register:

mov ax, 0b800h

mov ds, ax

; print "hello world"

; first byte is ascii code, second byte is color code.

mov [02h], 'h'

mov [04h], 'e'

mov [06h], 'l'

mov [08h], 'l'

mov [0ah], 'o'

mov [0ch], ','

mov [0eh], 'w'

mov [10h], 'o'

mov [12h], 'r'

mov [14h], 'l'

mov [16h], 'd'

mov [18h], '!'

; color all characters:

mov cx, 12 ; number of characters.

mov di, 03h ; start from byte after 'h'

c: mov [di], 11101100b ; light red(1100) on yellow(1110)

add di, 2 ; skip over next ascii code in vga memory.

loop c

; wait for any key press:

mov ah, 0

int 16h

ret

Q2. Program to add and subtract two numbers.

A2.

org 100h

mov al, 5

mov bl, 10

add bl, al

sub bl, 1

; print result in binary:

mov cx, 8

print: mov ah, 2 ; print function.

mov dl, '0'

test bl, 10000000b ; test first bit.

jz zero

mov dl, '1'

zero: int 21h

shl bl, 1

loop print

; print binary suffix:

mov dl, 'b'

int 21h

; wait for any key press:

mov ah, 0

int 16h

ret

Q3. Program to calculate sum of elements in vector form.

A3.

org 100h

; number of elements:

mov cx, 5

; al will store the sum:

mov al, 0

; bx is an index:

mov bx, 0

; sum elements:

next: add al, vector[bx]

; next byte:

inc bx

; loop until cx=0:

loop next

; store result in m:

mov m, al

; print result in binary:

mov bl, m

mov cx, 8

print: mov ah, 2 ; print function.

mov dl, '0'

test bl, 10000000b ; test first bit.

jz zero

mov dl, '1'

zero: int 21h

shl bl, 1

loop print

; print binary suffix:

mov dl, 'b'

int 21h

mov dl, 0ah ; new line.

int 21h

mov dl, 0dh ; carrige return.

int 21h;

print result in decimal:

mov al, m

call print\_al

; wait for any key press:

mov ah, 0

int 16h

ret

; variables:

vector db 5, 4, 5, 2, 1

m db 0

print\_al proc

cmp al, 0

jne print\_al\_r

push ax

mov al, '0'

mov ah, 0eh

int 10h

pop ax

ret

print\_al\_r:

pusha

mov ah, 0

cmp ax, 0

je pn\_done

mov dl, 10

div dl

call print\_al\_r

mov al, ah

add al, 30h

mov ah, 0eh

int 10h

jmp pn\_done

pn\_done:

popa

ret

endp

Q 4. Program to compare the 2 numbers??

A4.

org 100h

; (signed/unsigned)

; 4 is equal to 4

mov ah, 4

mov al, 4

cmp ah, al

nop

; (signed/unsigned)

; 4 is above and greater then 3

mov ah, 4

mov al, 3

cmp ah, al

nop

; -5 = 251 = 0fbh

; (signed)

; 1 is greater then -5

mov ah, 1

mov al, -5

cmp ah, al

nop

; (unsigned)

; 1 is below 251

mov ah, 1

mov al, 251

cmp ah, al

nop

; (signed)

; -3 is less then -2

mov ah, -3

mov al, -2

cmp ah, al

nop

; (signed)

; -2 is greater then -3

mov ah, -2

mov al, -3

cmp ah, al

nop

; (unsigned)

; 255 is above 1

mov ah, 255

mov al, 1

cmp ah, al

nop

stop: ret ; stop

Q5. Write a program to print binary, hexadecimal and octal values.

A5.

org 100h

; load binary value:

; (hex: 5h)

mov al, 00000101b

; load hex value:

mov bl, 0ah

; load octal value:

; (hex: 8h)

mov cl, 10o

; 5 + 10 = 15 (0fh)

add al, bl

; 15 - 8 = 7

sub al, cl

; print result in binary:

mov bl, al

mov cx, 8

print: mov ah, 2 ; print function.

mov dl, '0'

test bl, 10000000b ; test first bit.

jz zero

mov dl, '1'

zero: int 21h

shl bl, 1

loop print

; print binary suffix:

mov dl, 'b'

int 21h

; wait for any key press:

mov ah, 0

int 16h

ret

Q6. Program to check whether a string is palindrome or not.

A6.

org 100h

jmp start

m1:

s db 'able was ere ere saw elba'

s\_size = $ - m1

db 0Dh,0Ah,'$'

start:

; first let's print it:

mov ah, 9

mov dx, offset s

int 21h

lea di, s

mov si, di

add si, s\_size

dec si ; point to last char!

mov cx, s\_size

cmp cx, 1

je is\_palindrome ; single char is always palindrome!

shr cx, 1 ; divide by 2!

next\_char:

mov al, [di]

mov bl, [si]

cmp al, bl

jne not\_palindrome

inc di

dec si

loop next\_char

is\_palindrome:

; the string is "palindrome!"

mov ah, 9

mov dx, offset msg1

int 21h

jmp stop

not\_palindrome:

; the string is "not palindrome!"

mov ah, 9

mov dx, offset msg2

int 21h

stop:

; wait for any key press:

mov ah, 0

int 16h

ret

Q7. Program for simple input output.

A7.

#start=simple.exe#

#make\_bin#

; write byte value 0A7h into the port 110:

mov al, 0A7h

out 110, al

; write word value 1234h into the port 112:

mov ax, 1234h

out 112, ax

mov ax, 0 ; reset register.

; read byte from port 110 into AL:

in al, 110

; read word from port 112 into AX:

in ax, 112

hlt

Q8. Program to implement loop.

A8. org 100h

mov bx, 0 ; total step counter

mov cx, 5

k1: add bx, 1

mov al, '1'

mov ah, 0eh

int 10h

push cx

mov cx, 5

k2: add bx, 1

mov al, '2'

mov ah, 0eh

int 10h

push cx

mov cx, 5

k3: add bx, 1

mov al, '3'

mov ah, 0eh

int 10h

loop k3 ; internal in internal loop.

pop cx

loop k2 ; internal loop.

pop cx

loop k1 ; external loop.

; wait any key...

mov ah, 1

int 21h

ret

Q8. Program to implement Stack.

A8.

org 100h ; create tiny com file.

mov ax, 1234h

push ax

mov dx, 5678h

push dx

pop bx

pop cx

; function call pushes ip value of the next instruction:

call tfunc

mov ax, 7890h

push ax

pop bx

mov ax, 3

int 10h

mov ax, 0b800h

push ax

pop ds

; print "hi":

mov [170h], 'H'

mov [172h], 'i'

; color attribute for 'h'

mov [171h], 11001110b

; color attribute for 'i'

mov [173h], 10011110b

; wait for any key press....

mov ah, 0

int 16h

; here we "pop" the ip value,

; and return control to the operating system:

ret

; the test procedure:

tfunc proc

xor bx, bx

xor cx, cx

; here we "pop" the ip value,

; and return control to the main program:

ret

endp

Q 9. Program to print a string character by character.

A 9.

org 100h

print\_new\_line macro

mov dl, 13

mov ah, 2

int 21h

mov dl, 10

mov ah, 2

int 21h

endm

mov dx, offset msg1

mov ah, 9

int 21h

; input the string:

mov dx, offset s1

mov ah, 0ah

int 21h

; get actual string size:

xor cx, cx

mov cl, s1[1]

print\_new\_line

mov bx, offset s1[2]

print\_char:

mov dl, [bx]

mov ah, 2

int 21h

print\_new\_line

inc bx

loop print\_char

; wait for any key...

mov ax, 0

int 16h

ret

msg1 db "ENTER THE STRING: $"

s1 db 100,?, 100 dup(' ')

end

Q 10. Program for temperature conversion.

A10.

org 100h

jmp start

tc db 10 ; t celsius.

tf db 0 ; t fahrenheit.

result1 db ? ; result in fahrenheit.

result2 db ? ; result in celsius.

start:

; convert celsius to fahrenheit according

; to this formula: f = c \* 9 / 5 + 32

mov cl, tc

mov al, 9

imul cl

mov cl, 5

idiv cl

add al, 32

mov result1, al

mov bl, result1

call print ; print bl

; convert fahrenheit to celsius according

; to this formula: c = (f - 32) \* 5 / 9

mov cl, tf

sub cl, 32

mov al, 5

imul cl

mov cl, 9

idiv cl

mov result2, al

mov bl, result2

call print ; print bl

; wait for any key press...

mov ah, 0

int 16h

ret ; return to the operating system.