R. NITHYASRI
IT-B
3122 22 5002 086

UIT2401 MICROPROCESSORS - ASSIGNMENT

1)
```
data SEGMENT
    num1   db 10 ;
    num2   db 20 ;
    max-count db 5 ;
    count  db  0 ;

code SEGMENT
    assume CS: code , DS data;
    MOV   AX, data ;
    MOV   DS, AX ;
    MOV   AL, num1
    MOV   BL, num2 ;
    CMP   AL, BL
    JE   numbers - equal ;
    not - equal :
        INC  count ;
        MOV  AL, count
        CMP  AL, max - count ;
        JL  not - equal ;
        JMP  end-program

numbers-equal:
        INC  count ;

end-program :
      MOV  AX, 4cooh ;
      INT  21h ;

CODE  ENDS
```

2) data SEGMENT

```
   prices  db  10H, 20H, 30H, 40H, 50H, 60H, 70H, 8
   correction_factor db  03H;
   Count    DW  8 ;
end data ;
code  SEGMENT
        MOV   AX, data ;
        MOV   DS, AX
        MOV   CX, count ;
        LEA   SI, prices ;
        MOV   AL, correction_factor;
next_element :
        ADD   [SI], AL ;
        INC   SI    ;
        LOOP  next
```

~~End:~~

```
      MOV  AX , 4Cooh;
      INT  21h ;

code ENDS
```

3) data SEGMENT

```
     prices db  10H, 20H, 30H, 40H, 50H, 60H,
      70H, 80H ;
     correction - factor db  05H  ;
     count  DW  8 ;
     modified - prices  db  8 DUP (?);

code  SEGMENT
     MOV   AX , DATA;
     mov   DS , AX ;
     MOV   ES, AX ;
     MOV   CX, count ;
     LEA   SI , prices ;
     LEA   DI , modified - prices ;
     MOV   AL, correction _ factor;

next element :
      MOV  BL, [SI] ;
      ADD  BL, AL  ;
      MOV  [DI], BL ;
      INC   SI ;
      INC   DI  ;
      LOOP  next _element ;
```

```
; MOV AX, 4COOH ;
  INT 21H ;

data ENDS

4)
data SEGMENT
    byte1 db 25H ;
    byte2 db 15H ;
    result db ? ;
END data
code SEGMENT
  assume cs: code, Ds: data;
      MOV AX, data ;
      MOV DS, AX ;
      MOV AL, byte1 ;
      MOV BL, byte2 ;
      ADD AL, BL ;
      MOV result, AL ;
      MOV AX, 4cooh ;
      INT 21h
END code ;
```

```
5)
data SEGMENT
    array db 5, 10, 20, 30, 40, 50 ;
    sum dw 0 ;
    average db ? ;
END data
code SEGMENT
      MOV AX, data ;
      MOV DS, AX ;
      LEA SI, array ;
      MOV CL, [SI] ;
      MOV CH, 0 ;
      JCXZ end-program ;
      XOR AX, AX ;
      MOV [sum], AX ;
      INC SI ;
sum_loop : MOV AL, [SI] ;
      ADD [sum], AX ;
      INC SI ;
      LOOP sum-loop ;
      ;
```

```
        MOV AX, [SUM] ;
        MOV CL, [array] ;
        XOR  CH, CH ;
        DIV  CL ;
        ;
         MOV [average], AL ;
         MOV  AX, 4C00h ;
         INT  21h ;

code   ENDS ;

6)
data  SEGMENT
        string  db  'Hello, World!',(
        res      db   15 DUP (?) ;
data   ENDS ;
code   SEGMENT
             assume  CS: code, DS: data
             MOV  AX, Data ;
             MOV  DS, AX ;
             MOV  ES, AX ;
             LEA  SI, string ;
             LEA  DI, res ;
```

```
copy-loop :
            LODSB ;
            STOSB ;
            CMP  AL, 0 ;
            JNZ  copy-loop ;

         ;
            MOV  AX, 4C00h ;
            INT  21h ;

code   ENDS .

7) data  SEGMENT
      array  db   'Hello, World!', 0 ;
      length  dw  ? ;
END  data ;
code   SEGMENT
            MOV  AX, data ;
            MOV  DS , AX ;
            MOV  ES , AX ;
         ;
            LEA  SI, array ;
            MOV  CX, 0  ;
calculate - length :
    LODSB ;
    CMP AL, 0 ;
    JE length-calculated ;
     INC CX;
     JMP calculate-length ;

length-calculated :
        MOV length, cx ; ;
        LEA SI, array ;
        LEA DI, array ;
        ADD DI, CX ;
        DEC DI;
reverse - loop :
        MOV AL, [SI] ;
        MOV BL, [DI] ;
        MOV [SI], BL ;
        MOV [DI], AL ;
        INC SI ;
        DEC SI ;
        CMP SI, DI ;
        JL reverse-loop;

;
        LEA DX, array ;
        MOV AH, 09h;
        INT 21h ; ;

        MOV AX, 4C00h ;
        INT 21h;
```

8)

```
data SEGMENT.
    LENGTH equ ($ - LIST);
    found db '66H found in LIST',
     0AH, 0DH, '$'
     not - found db '66H not found in
     LIST', 0AH, 0DH, '$'.
     LIST db 10, 20, 66, 30, 40, 66, 50, 60;

;
code SEGMENT
     assume CS= code, DS = data;
     MOV AX, data ;
     MOV DS, AX ;
     }
     MOV CX, length ;
     LEA SI, LIST ;
search-loop :
      MOV AL, [SI] ;
      CMP AL, 66h ;
      JE found ;
      INC SI ;
      LOOP search-loop ; ;
    MOV DX, OFFSET not-found-ms
    MOV AH, 09h;
    INT 21h ;

    ;
    MOV AX, 4C00h;
    INT 21h;

found :
    MOV DX, offSET found-msg
    MOV AH, 09h;
    INT 21h; ;

    MOV AX, 4C00h ;
    INT 21h ;

Code ENDS.

9)
data SEGMENT
    temperature db ? ;
    below - 30- msg db `Light
    yellow lamp', '$';
    above_ 30- msg db `Light green
    lamp', '$';
END data

code SEGMENT
    MOV AX, data ;
    MOV DS, AX ;
    MOV AH, 09h ;
    LEA DX, prompt-msg ;
    INT 21h ;

    ;
    MOV AH, 01h ;
    INT 21h ;
    SUB AL, 30h ;
    MOV temperature, AL ;

    ;
    MOV AL, temperature;
    CMP AL, 30 ;
    JL below - 30 ;
    JMP above - 30 ;

below_30 :
    MOV AH, 09h ;
    LEA DX, below - 30- msg ;
    INT 21h ;
    JMP end - program;
```

```
above_30 :
        MOV AH, 09h;
        LEA DX , above_30_msg;
        INT 21h;
~~en~~. MOV AX, 4C00h;
        INT 21h;
Prompt_msg db 'ENTER TEMP : $';
code ENDS.
10) data SEGMENT
        password DB 'password'
        buffer DB 20 DUP(?)
        msg_correct DB
        'Correct password entered'.
        0DH, 0AH, $;
        msg_incorrect DB
        'Incorrect pwd entered',
        0DH, 0AH, $;

        prompt_msg DB
        'Enter the pwd : $';
```

```
data ENDS
code SEGMENT
        MOV AX, DATA ;
        MOV DS, AX ;
        MOV AH, 09h ;
        LEA DX, prompt_msg ;
        INT 21h ;

        ;
        MOV SI, offset buffer ;
        MOV DI, offset password;
        MOV CX, 8;
        REPE CMPSB ;
        JNE incorrect_pwd;
        MOV AH, 09h;
        LEA DX, msg_correct;
        INT 21h;
        JMP end ;
incorrect_pwd :
        MOV AH, 09h ;
        LEA DX, msg_incorrect;
        INT 21h ;
```

```
end :
        MOV  AX, 4C00h ;
        INT  21h;
code  ENDS.
```

2024, 09:28