1)Write an ALP that will examine a set of five strings. The five strings are stored in *a **file1.txt***. The length each string is six. The ALP has to find the number of strings that are a palindrome. The count of palindromes must be stored in location ***'palin1'***. The count of palindrome should be displayed on the screen. The ALP should be written as a main and subprogram. The subprogram ***'pal1'*** should detect whether the string is a palindrome or not. The starting address of string is passed as parameter using SI register. Use tiny model.

file1.txt is provided to you. Download it into the current folder.

```
.model  tiny
.486
.data
str1    db      45 dup(?)
fname   db      'file1.txt',0
cnt     db      5
palin1  db      30h
.code
.startup
        lea     dx,fname
        mov     ah,3dh
        mov     al,02h
        int     21h
        mov     bx,ax
        mov     ah,3fh
        mov     cx,45
        lea     dx,str1
        int     21h
        lea     si,str1
        mov     cx,5
x1:     call    pal1
        add     si,8
        loop    x1
        mov     ah,02h
        mov     dl,palin1
        int     21h
.exit
pal1    proc    near
        push    si
        mov     di,si
        add     di,5
        mov     dl,3
x2:     lodsb
        cmp     al,[di]
        jne     x3
        dec     di
        dec     dl
        jnz     x2
        inc     palin1
x3:     pop     si
        ret
pal1    endp
end
```

2) Write an ALP that does the following
Interleaves the content of file a.txt and b.txt and stores the interleaved data in c.txt
For e.g. if
a.txt has data ‘microprocessor’
b.txt has data ‘interfacing’
the file c.txt should have the data
miinctreorpfraocciensgs o r
You can assume files a.txt, b.txt and c.txt are available in the current directory. Maximum number of bytes in a.txt and b.txt is 20 bytes

```
.model tiny
.data
fil1 db 'a.txt',0
fil2 db 'b.txt',0
fil3 db 'c.txt',0
dat1 db 20 dup(20)
dat2 db 20 dup(20)
dat3 db 40 dup(20)
.code
.startup
mov ah,3dh
lea dx,fil1
mov al,02h
int 21h
mov bx,ax
mov ah,3fh
mov cx,20
lea dx,dat1
int 21h
mov ah,3eh
int 21h
mov ah,3dh
lea dx,fil2
mov al,02h
int 21h
mov bx,ax
mov ah,3fh
mov cx,20
lea dx,dat2
int 21h
mov ah,3eh
int 21h
lea si,dat1
lea bx,dat2
lea di,dat3
mov cx,20
x1: mov al,[si]
mov [di],al
inc di
mov al,[bx]
mov [di],al
inc di
inc si
inc bx
loop x1
mov ah,3dh
lea dx,fil3
mov al,02h
int 21h
mov bx,ax
mov ah,40h
mov cx,40
lea dx,dat3
int 21h
mov ah,3eh
int 21h
.exit
end
```

3) Write an ALP that will do the following:
Display “Enter name of file: ”
The contents of file whose name is entered by user will be read. Data is stored in this file as a set of two digit decimal nos. in a single line - for e.g. 73712341561234.
Find the smallest in the set of 2-digit decimal number and display the number on the screen in the next line.
File should not be created in ALP it is already available in current directory. The count of the 2 digit decimal data available in the file is 10.

```
.model tiny
.486
.data
datc    db      14
datca   db      ?
dat1    db      14 dup(0)
fil1    db      14 dup(0)
        db      0
datf    db      20 dup(0)
datb    db      10 dup(0)
dis1    db      'Enter the name of the file: $'
dis2    db      0ah,0dh
dis3    dw      '$$','$$'
.code
.startup
        lea     dx,dis1
        mov     ah,09h
        int     21h
        lea     dx,datc
        mov     ah,0ah
        int     21h
        movzx   cx,datca
        lea     si,dat1
        lea     di,fil1
 rep    movsb
        mov     ah,3dh
        lea     dx,fil1
        mov     al,02h
        int     21h
        mov     bx,ax
        mov     ah,3fh
        mov     cx,20
        lea     dx,datf
        int     21h
        mov     ah,3eh
        int     21h
        lea     si,datf
        lea     di,datb
        mov     cx,20
x1:     lodsw
        and     ax,0f0fh
        rol     al,4
        or      al,ah
        stosb
        dec     cx
        dec     cx
        jnz     x1
        mov     cx,10
        dec     cx
        lea     si,datb
        lodsb
x3:     cmp     al,[si]
        jbe     x2
        mov     al,[si]
x2:     inc     si
        loop    x3
        mov     ah,al
        rol     ah,4
        and     ax,0f0fh
        or      ax,3030h
        xchg    ah,al
        mov     dis3,ax
        lea     dx,dis2
        mov     ah,09h
        int     21h
.exit
end
```

4) Write an ALP that will do the following:

1. Display “Enter File Name”
2. Enter File Name as a user input.
3. Examine a set of strings in a file. The length of each string is four characters.
4. The number of strings in the file is 5.
5. The ALP should scan each string and see if it is equal to ‘ ‘ (four blank spaces). If yes, all strings that follow this string of four blank spaces should be reversed until the next string of four blank spaces is encountered.

All strings are on the same line and are stored without spaces except in case of 4 blank character string.
E.g. File content will be as follows
Mathhave yourmine dead
The result in the same file same line must be
Mathhave ruoyenim dead
You have to use a single instruction to reverse the string.

```
.model tiny
.486
.data
dis1    db      "Enter File name: $"
kcnt1   db      13
acnt    db      ?
fil1    db      14 dup(0)
fil2    db      14 dup(0)
dat1    db      40 dup(0)
cnt1    db      10
hand1   dw      ?
.code
.startup
        lea     dx,dis1
        mov     ah,09h
        int     21h
        mov     ah,0ah
        lea     dx,kcnt1
        int     21h
        lea     si,fil1
        lea     di,fil2
        movzx   cx,acnt
 rep    movsb
        mov     ah,3dh
        mov     al,02h
        lea     dx,fil2
        int     21h
        mov     hand1,ax
        mov     ah,3fh
        mov     bx,hand1
        lea     dx,dat1
        mov     cx,40
        int     21h
        mov     ah,3eh
        int     21h
        lea     di,dat1
        movzx   cx,cnt1
        mov     eax,'    '
 repne  scasd
        mov     si,di
x1:     lodsd
        cmp     eax,'    '
        jz      x2
        bswap   eax
        stosd
        loop    x1
x2:     mov     ah,3dh
        mov     al,02h
        lea     dx,fil2
        int     21h
        mov     hand1,ax
        mov     ah,40h
        mov     bx,hand1
        lea     dx,dat1
        mov     cx,40
        int     21h
        mov     ah,3eh
        int     21h

.exit
end
```

5) Write an ALP that will scan a database of grades stored in a file marks.txt – the grades are to be stored as follows in file: Last three digits of the id followed by grade. Only coarse grading is used so valid grades are – A, B, C D, E. For e.g. for 5 students the storage in memory will be as follows:
'238', 'A', '211', 'B', '247', 'C', '110', 'E', '111', 'B'. There are no spaces/newlines etc. The count of students is 10. Your ALP must do the following:
(a) Display –" Enter the grades for which you need ids : "
(b) Take in the single character grade in the same line with Echo.
(c) Open the file marks.txt and transfer its contents into location ***dat1.***
(d) Find the id of students and number of students who have scored the grade entered by the user and store the ids alone starting from location ***res1***, the count of students who have got the grade should be stored in location ***grd1***.
(e) Display the id of the students on the next line in the following manner :
For e.g. if grade entered by user is B. the display should be
211
111
(f) Store the same values you display in the same format in the file ***'grades.txt'***
marks.txt is already available in current directory(you can create it in windows and store data in relevant format)grades.txt has to be created and written in by the ALP.

```
.model tiny
.data
dis1    db     "Enter the grades for which we you need ids: $"
inp1    db     ?
fil1    db     'marks.txt',0
fil2    db     'grades.txt',0
han1    dw     ?
han2    dw     ?
dat1    db     50 dup(?)
cnt1    dw     0ah
dis2    db     0ah, 0dh
res1    db     41 dup('$')
grd1    db     0
.code
.startup
      lea     dx,dis1
      mov     ah,09h
      int     21h
      mov     ah,01h
      int     21h
      mov     inp1,al
      mov     ah,3dh
      lea     dx,fil1
      mov     al,02h
      int     21h
      mov     han1,ax
      mov     ah,3fh
      mov     bx,han1
      mov     cx,45
```

```
      lea     dx,dat1
      int     21h
      mov     ah,3eh
      int     21h
      lea     di,res1
      lea     si,dat1
x3:    mov     cx,3
      mov     al,[SI+3]
      cmp     al,inp1
      jne     x1
      inc     grd1
   rep movsb
      mov     dx,0a0dh
      mov    [di],dx
      inc     di
      inc     di
      jmp     x2
x1:    add     si,3
x2:    inc     si
      dec     cnt1
      jnz     x3
      lea     dx,dis2
      mov     ah,09h
      int     21h
      mov     ah,3ch
      lea     dx,fil2
      mov     cl,20h
      int     21h
      mov     han1,ax
      mov     ah,3dh
      lea     dx,fil2
      mov     al,02h
      int     21h
      mov     bx,ax
      mov     al,grd1
      mov     cl,5
      mul     cl
      mov     cx,ax
      lea     dx,res1
      mov     ah,40h
      int     21h

.exit
end
```