Class Exercise 2

1. In the following code fragments character strings will be defined and **_PutStr** macro calls are given to display them. Determine what appears on the CRT screen when the DOS calls have all been completed. You should assume that the cursor is at the left side of the screen at the beginning of each **_PutStr**, and you should use an underscore **(_)** as in the previous class exercises where the cursor ends up.

```
Ex1Msg      DB      'Ex1 shows', 13, 10, 'two lines', 13, 10, '$'
            ...
            _PutStr Ex1Msg

Ex2Msg1     DB      'Does Ex2 show', 13, 10
Ex2Msg2     DB      'two lines too?', 13, 10, '$'
            ...
            _PutStr Ex2Msg1

Ex3Msg      DB      'What does Ex3', 13, 10, 10, 'do?$'
            ...
            _PutStr Ex3Msg

Ex4Msg      DB      'Ex4 is also', 13, 'strange$'
            ...
            _PutStr Ex4Msg

Ex5Msg      DB      'Ex5 is too', 8, 8, 8,  'not', 13, 10, '$'
            ...
            _PutStr Ex5Msg
```

Hint: When you display a character in a position that already contains one, the new character replaces the old one.

```
Ex6Msg      DB      'The 1000 $ question', 13, 10, '$'
            ...
            _PutStr Ex6Msg

Ex7Msg      DB      'The 1000 ', 36, ' question', 13, 10, '$'
            ...
            _PutStr Ex7Msg
```

Hint: 36 is the ASCII value of '$'

2. Suppose we have declared:

```
Message     DB      'Boo'
Char        DB      'k'
Ending      DB      '$', 13, 10
```

What is displayed by the following code below?

```
1.  _PutStr   Message

2.     mov    Char, 'm'
       _PutStr   Message

3.     mov    Char, '$'
       _PutStr   Message
```

3. Suppose you want to do a destructive backspace, that is you want to erase a character on the screen that you were backspacing into. In other words, if the on-screen image is

ABCX_

(_ is the cursor), you want it to end up looking like

ABC_

Which byte string should you display with the DOS 9h function to do the erase?

4. What is displayed by the following sets of DOS calls?

a) **_PutCh** **'H', "a", 'n', "s", 13, 10**

b) **_PutCh** **72, 97, 110, 115, 13, 10**

c) **_PutCh** **" " , " ", "C", 13, " ", "B", 13**
_PutCh **'A', 13, 10**

d) **_PutCh** **'x', 'y', 'C', 8, 8, 'B', 8, 8**
_PutCh **'A', 13, 10**

e) **_PutCh** **'\', 10, '\', 10, '\', 13, 10**

5. Write data declaration (DBs) and DOS calls to display the line:

The 1000 $ question shall be asked tomorrow

6. Write a program called **Today** using **PutDec** which displays today's date in the form:

Today is dd/m/yyyy

with the day, month and year filled in. The date is returned by the **_GetDate** macro:

```
_GetDate        from PCMAC.INC
; after the call       dh = month (1 to 12)
                        dl = day (1 to 31)
                        cx = year (e.g. 1975)
                        (al also contain the weekday (0 to 6))
```

Hints: You can extend a byte to word by zero filling the higher register. **_PutCh** and **_PutStr** destroys the contents of **ax** and **dx**. Don't call your program **date.asm** because **date.com** is a DOS shell command to set the system date of the PC.