組合語言 期中考　　109 年 11 月 12 日

寫下列題目中的組合語言程式可以使用的指令有：MOV, ADD, SUB, ADC, SBB, NEG, INC, DEC, JMP, JB, JNB, JA, JNA, JG, JNG, JL, JNL, JC, JO, JP, JS, JNC, JNS, JNO, JNS, JNP, JZ, JNZ, JCXZ, LOOP, LOOPZ, LOOPNZ, CALL, RET, PUSH, POP, AND, OR, NOT, XOR, TEST, CMP, STC, CLC 或課本第 6 章(包含)前，在課堂中講解過卻遺漏的其它指令。

可能用到的 ASCII 碼：ENTER 是 0DH，換行是 0AH，空格是 20H，數字'0'是 30H，英文字母'A'是 41H，'Z'是 5AH，'a'是 61H，'z'是 7AH。

若覺得題目有錯誤，可說明錯誤處，並自行加以修改，而後解題。若題目無錯誤，而自行修改，當作是答錯。

1. 在資料段(data segment)中定義的資料如下所示:：

```
hexdgt byte 10 dup (?)
hexno byte ?
textout byte 11 dup (?)
```

其中，由 hexdgt 起，每個位置已經放有一位數正確的十六進制數值，hexno 則存放著數值的個數(一定大於 1)。我們想將這些十六進數值轉換成文字字串，存放在 textout 起的位置中。例如，hexdgt 起的內容依次是 1、2、3、4、0AH、0BH、6、7、8 和 9，hexno 的內容是 5，表示十六進制數值 1234A。你的程式執行後，textout 存放的字串應該是"1234AH"。請寫出將這數值轉成文字的組合語言程式部分。請注意: 字串的結束應有一個數碼 0。若第一個數字是'A'到'F'，應在前面加一個'0'。

2. 資料段(data segment)中定義了一個字串資料 str1，如下所示：

```
Str1 byte 100 dup (?)
```

程式執行時，由位置 str1 起已放入一個以數值 0 結束的字串。請寫出一段組合語言程式，計算字串 str1 中數字、大寫和小寫字母的個數。大寫字母的個數放在暫存器 AL 中，小寫字母個數放在暫存器 AH 中，數字個數放在暫存器 BL 中。

3. 資料的定義如下所示:：

```
NUM1 DWORD 2 DUP (?)
NUM2 DWORD 2 DUP (?)
NUM3 DWORD 3 dup (?)
```

其中，NUM1 和 NUM2 用以分別存放兩個 64 位元的數值，每個數值以兩個 32 位元方式存放，低階部分放在低階位置。請寫一 32 位元機器的組合語言程式將這兩個 64 位元數值相加，將結果存放在 NUM3 起的位置中，也是低階部分放在低階位置。你應將低階 32 位元相加，再做高階 32 位元的加法。低階部分相加可能有進位，須加到高階部分。高階部分也可能產生進位。例如，NUM1 起存放的資料依序是 00001234H、56780000H，NUM2 起存放的資料依序是 0ABCD0000H、0EF123456H。意思是存放了 64 位元資料 5678000000001234H 和 0EF123456ABCD0000H。程式執行後，NUM3 起存放的資料依序應該是 ABCD1234、458A3456

和 1。

4. 請將下列 C++語言程式轉成組合語言(只要能達到相同結果即可，中間變數和過程可以省略)。

```
int b[] = {1,2,3,4,10,20,30,40,50,60,70,80};
int num = 10;
unsigned int i;
for(i = 0; i < num; i++)
 b[i] = b[i]+1;
```

5. 請說明下列程式片斷執行到標記 L2 前，執行每一個指令後，暫存器(EAX、EBX、ECX、ESI 或 EDI)或記憶體位置的內容有甚麼改變。假設在執行第一個指令前，暫存器 EAX、EBX、ECX、ESI 和 EDI 的內容都是 0。VAR1 的地址是 406000H。請你也計算出 VAR2 和 VAR3 的地址。你的答案可以是個數值，如 12H；也可以是個未知值(地址或其它定義常數)，如 VAR1+4 (即 406004H)。記憶體內容可寫成記憶體位置 VAR3 存 32 位元的 12345678H，或記憶體位置 VAR3+1 存 8 位元的 12H。

```
  .DATA
  VAR1  BYTE 1, 2, 3, 4, 5, 6, 7, 8
  VAR2  WORD 9, 10, 11, 12
  VAR3  DWORD 2
  .CODE
    MOVZX  EAX, VAR1
    MOV BX, VAR2+2
    MOV ECX, VAR3
    MOV ESI, OFFSET VAR1
    MOV EDI, OFFSET VAR2
 L1:
    INC EBX
    DEC AL
    DEC WORD PTR[ESI]
    ADD WORD PTR [EDI+2], 10
    ADD EDI, VAR3
    LOOP L1
 L2:
```