1. Considere o seguinte programa, destinado a ser executado no simulador do PEPE. Ao executar, este programa faz alguns acessos à memória (<u>para além das buscas de instruções, que se ignoram</u>). A tabela indica em que endereço o primeiro acesso é feito (uma escrita).

```
                PLACE   0000H
0000H                   MOV     SP, 1800H
0002H                   MOV     R1, 231FH
0004H                   CALL    X
0006H           fim:    JMP     fim
0008H           X:      MOV     R2, 1234H
000AH                   CALL    Y
000CH                   RET
000EH           Y:      MOV     [R2], R1
0010H                   RET
```

a) Preencha os espaços no programa (os do lado esquerdo são os endereços em que as instruções ficam).

b) Acabe de preencher a tabela com os acessos à memória feitos pelo programa, indicando em que endereço o acesso é feito, se é leitura (L) ou escrita (E) e o respectivo valor (lido ou escrito).

Endereço	L ou E	Valor
17FEH	E	**0006H**
17FCH	**E**	**000CH**
1234H	**E**	**231FH**
17FCH	**L**	**000CH**
17FEH	**L**	**0006H**

2. Ao analisar um programa de assembly, o assemblador do PEPE produz uma tabela interna com os símbolos (constantes simbólicas) e respectivos valores. Considere o seguinte programa e arme-se em assemblador, preenchendo os valores (em hexadecimal) dos símbolos usados.

```
PLACE   1000H
X:      WORD    2000H
Y       EQU     1000H
W:      STRING  "abcdef"
Z:      WORD    1006H
PLACE   0
A:      MOV     SP, 1000H
        MOV     R1, 1000H
B:      SUB     R1, 2
C:      JNZ     B
fim:    JMP     fim
```

Símbolo	Valor (hexadecimal)
X	**1000H**
Y	**1000H**
W	**1002H**
Z	**1008H**
A	**0000H**
B	**0004H**
C	**0006H**
fim	**0008H**

3. Considere o seguinte programa. Indique qual o valor dos registos quando o programa termina.

```
PLACE   1000H
X:      WORD    2BBFH
Y:      WORD    2B4FH
        WORD    54AFH
        WORD    27AFH
PLACE   0
A:      MOV     SP, 2000H
        MOV     R1, X
B:      MOV     R2, 2
        CALL    AAA
fim:    JMP     fim
AAA:    SHL     R2, 1
        MOV     R3, [R1+R2]
        RET
```

R1	**1000H**
R2	**0004H**
R3	**54AFH**
SP	**2000H**