1) Está resuelto en la teoría.

3)
```
            ORG 1000H
   LETRA    DB  41H
; 41h es el caracter ascii de la letra A  mayúscula
;
            ORG 2000H
            MOV AH, 27 ; cantidad de letras
            MOV BX, OFFSET LETRA ; Dirección donde está cargada la letra
            MOV AL, 1; Cantidad de caracteres a mostrar en pantalla
    SIGO:   INT 7 ; Función que muestra en pantalla una cadena de caracteres
            ADD BYTE PTR [BX], 20H; Las minúsculas están corridas  en 20h
            INT 7
            SUB BYTE PTR [BX], 20H ; Vuelvo a las mayúsculas
            ADD BYTE PTR [BX], 1 ;  Próxima letra
            DEC AH
            JNZ SIGO
            HLT
            END
```

4) Está resuelto en teoría

5) El número ingresado por teclado es almacenado en memoria como el ASCII que representa al número, ej ASCII 0 = 30H, el ASCII 1 = 31H.....ASCII 9=39H

```
               ORG 1000H
        NUM    DB  ?
        MSJ1   DB  "CARACTER NO VALIDO"
        FIN1   DB  ?
        MSJ2   DB  "CARACTER VALIDO"
        FIN2   DB  ?
;
;
            ORG 3000H     ;     Subrutina que determina si el caracter ingresado es un número
 ES_NUM:  MOV AH, [BX]  ; Cargo en AH el caracter ingresado
            SUB AH, 30H
            CMP AH, 10 ; Si es un número en AH hay un valor entre 0 y 9.
            JNC NO_ES
            MOV AH, 0FFH
            JMP FIN
   NO_ES:   MOV AH, 00H
      FIN:  RET
;
;
;
```

```
                ORG 2000H
                MOV BX, OFFSET NUM ; Dirección donde se almacena el carácter ingresado por
teclado
                INT 6
                CALL ES_NUM; Llamado a subrutina que verifica si es un número
                CMP AH, 00H ; Si devuelve 0 en AH no es un número
                JZ NO_ES_NUM
                MOV BX, OFFSET MSJ2
                MOV AL, OFFSET FIN2 - OFFSET MSJ2
                INT 7
                JMP AFUERA
NO_ES_NUM:      MOV BX, OFFSET MSJ1
                MOV AL, OFFSET FIN1 - OFFSET MSJ1
                INT 7
    AFUERA:     HLT
                END
```

6) Suponemos que el carácter ingresado es un número. Para verificarlo habría que usar la rutina del ej5.

```
                ORG 1000H
          MSJ   DB "INGRESE UN NUMERO"
         FINM   DB ?
          NUM   DB ?
        CERO0   DB "CERO"
         FIN0   DB ?
         UNO1   DB "UNO"
         FIN1   DB ?
         DOS2   DB "DOS"
         FIN2   DB ?
        TRES3   DB "TRES"
         FIN3   DB ?
    CUATRO4     DB "CUATRO"
         FIN4   DB ?
       CINCO5   DB "CINCO"
         FIN5   DB ?
        SEIS6   DB "SEIS"
         FIN6   DB ?
       SIETE7   DB "SIETE"
         FIN7   DB ?
      OCHO8     DB "OCHO"
         FIN8   DB ?
     NUEVE9     DB "NUEVE"
         FIN9   DB ?
;
;
                ORG 2000H
    SIGO1:      MOV AH, 0; Cuenta en AH las veces que ingresó 0 en forma consecutiva
```

```
SIGO:     MOV BX, OFFSET MSJ
          MOV AL, OFFSET FINM - OFFSET MSJ
          INT 7
          MOV BX, OFFSET NUM
          INT 6
          MOV CL, NUM
          CMP CL, 30H
          JNZ UNO_1
          MOV BX, OFFSET CERO0
          MOV AL, OFFSET FIN0 - OFFSET CERO0
          INT 7
          ADD AH, 1 ; Suma una vez que ingresó en 0
          CMP AH, 1  ; Si no es 0, entonces es la segunda vez que ingresa un 0 y el programa
          JZ SIGO       ;termina. Va a SIGO si es el primer 0
          JMP AFUERA  ; Por acá pasa si es el segundo 0.
UNO_1:  CMP CL, 31H
          JNZ DOS_2
          MOV BX, OFFSET UNO1
          MOV AL, OFFSET FIN1 - OFFSET UNO1
          INT 7
          JMP SIGO1; En SIGO1 pone a 0 la cuenta consecutiva de ceros
DOS_2:  CMP CL, 32H
          JNZ TRES_3
          MOV BX, OFFSET DOS2
          MOV AL, OFFSET FIN2 - OFFSET DOS2
          INT 7
          JMP SIGO1
TRES_3:  CMP CL, 33H
          JNZ CUATRO_4
          MOV BX, OFFSET TRES3
          MOV AL, OFFSET FIN3 - OFFSET TRES3
          INT 7
          JMP SIGO1
CUATRO_4:  CMP CL, 34H
          JNZ CINCO_5
          MOV BX, OFFSET CUATRO4
          MOV AL, OFFSET FIN4 - OFFSET CUATRO4
          INT 7
          JMP SIGO1
CINCO_5:  CMP CL, 35H
          JNZ SEIS_6
          MOV BX, OFFSET CINCO5
          MOV AL, OFFSET FIN5 - OFFSET CINCO5
          INT 7
          JMP SIGO1
SEIS_6:  CMP CL, 36H
          JNZ SIETE_7
          MOV BX, OFFSET SEIS6
```

```
            MOV AL, OFFSET FIN6 - OFFSET SEIS6
            INT 7
            JMP SIGO1
SIETE_7:    CMP CL, 37H
            JNZ OCHO_8
            MOV BX, OFFSET SIETE7
            MOV AL, OFFSET FIN7 - OFFSET SIETE7
            INT 7
            JMP SIGO1
OCHO_8:     CMP CL, 38H
            JNZ NUEVE_9
            MOV BX, OFFSET OCHO8
            MOV AL, OFFSET FIN8 - OFFSET OCHO8
            INT 7
            JMP SIGO1
NUEVE_9:    MOV BX, OFFSET NUEVE9
            MOV AL, OFFSET FIN9 - OFFSET NUEVE9
            INT 7
            JMP SIGO1
AFUERA:     HLT
            END
```

7) Supongamos que la suma de los dos números es ≤ 9. El número ingresado por teclado es almacenado en memoria como el ASCII que representa al número, ej ASCII 0 = 30H, el ASCII 1 = 31H.....ASCII 9=39H, es decir si los números ingresados son 2 (ASCII 32H) y 3 (ASCII 33H) la suma resultará 32H+33H=65H. Este resultado tiene sumado dos veces 30H, entonces hay que restar 30H para obtener el resultado correcto 35H, que es el ASCII de 5. Limitamos el resultado al caso ≤ 9 para que tenga un solo dígito, si el resultado tuviese 2 dígitos. por ej 7+8=15, habría que guardarlo en 2 posiciones de memoria una para el 1 (31H) y otra para el 5 (35H).

```
            ORG 1000H
MSJ1        DB  "INGRESE EL PRIMER NUMERO:"
FIN1        DB  ?
MSJ2        DB  "INGRESE EL SEGUNDO NUMERO:"
FIN2        DB  ?

            ORG 1500H
NUM1  DB  ?
NUM2  DB  ?
RES   DB  ?


            ORG 2000H
            MOV BX,OFFSET MSJ1  ;    BX:=1000H apunta a MSJ1
```

```
                MOV AL,OFFSET FIN1-OFFSET MSJ1;    AL=25 cantidad de carac. en
MSJ1
                INT 7          ;  Muestra en pantalla MSJ1
                MOV BX,OFFSET NUM1 ;  BX=1500H apunta a NUM1 donde se guardará el
número ingresado
                INT 6  ;               Espera dato ingresado por el teclado
                MOV AL,1  ;            Cantidad de números ingresados
                INT 7   ;                Muestra en pantalla el número ingresado por teclado
                MOV BX,OFFSET MSJ2 ; BX=101AH apunta a MSJ2
                MOV AL,OFFSET FIN2-OFFSET MSJ2 ; AL=26=1AH cantidad de carac. en
MSJ2
                INT 7 ;                Muestra en pantalla MSJ2
                MOV BX,OFFSET NUM2 ;  BX=1502H apunta a NUM2 donde guardará el
número ingresado
                INT 6       ;          Espera dato ingresado por teclado
               MOV AL,1
                INT 7   ;               Muestra en pantalla el número ingresado por teclado
               MOV AL,NUM1  ;   Carga en AL el primer número ingresado
               ADD AL,NUM2  ; Suma los dos números ingresados por teclado
               SUB AL, 30H ; Resta al resultado 30H para formar el ASCII
               MOV RES,AL ; Almacena en RES el resultado
               MOV BX,OFFSET RES ; BX=1502H apunta a RES
               MOV AL,1  ; Cantidad de números del resultado
               INT 7 ; Muestra el resultado en pantalla.
               INT 0
               END
```

Veamos ahora el caso que nos puede dar como resultado dos dígitos en vez de uno solo. Supongamos sumar dos números (en hexadecimal y en ascii 9 + 5) :

39h + 35h = 6Eh para saber que dígito decimal es 6Eh – 30h = 3Eh no corresponde a ningún ascii de dígito decimal (30h…..39h). Por lo tanto si el resultado es mayor que 3Ah, corresponde a 2 dígitos.

```
                ORG  1000H
MSJ1        DB  "INGRESE PRIMER NUMERO"
FIN1        DB   ?
MSJ2        DB  "INGRESE SEGUNDO NUMERO"
FIN2        DB   ?
NUM1        DB   ?
NUM2        DB   ?
RESUL       DB   30H ;
             DB   30H
;
;
                ORG 2000H
                MOV AX, 0000H
                MOV BX, OFFSET MSJ1
```

```
                    MOV AL, OFFSET FIN1 - OFFSET MSJ1
                    INT 7
                    MOV BX, OFFSET NUM1
                    INT 6
                    MOV BX, OFFSET MSJ2
                    MOV AL, OFFSET FIN2 - OFFSET MSJ2
                    INT 7
                    MOV BX, OFFSET NUM2
                    INT 6
                    MOV AL, NUM1
                    ADD AL, NUM2
                    SUB AL, 30H
                    CMP AL, 3AH
                    JC  RESUL_CORREC
                    ADD AL, 6
                    SUB AL, 10H
                    ADD AH, 31H
                    JMP ABAJO
RESUL_CORREC:   MOV AH, 30H
        ABAJO:  MOV RESUL, AH
                    MOV RESUL+1, AL
                    MOV BX, OFFSET RESUL
                    MOV AL, 2
                    INT 7
                    HLT
                    END
```

8) Está resuelto en archivo aparte

9) Vamos a mostrar primero el ej. con una clave de un solo caracter

```
            ORG 1000H
MSJ         DB   "INGRESE CLAVE"
FIN         DB   ?
CLAVE       DB    "M"
ING         DB   ?
CAR1        DB  "ACCESO PERMITIDO"
FIN1        DB    ?
CAR2        DB  "ACCESO DENEGADO"
FIN2        DB   ?

        ORG 2000H
        MOV BX, OFFSET MSJ
        MOV AL, OFFSET FIN-OFFSET MSJ
        INT 7
        MOV BX, OFFSET ING
        INT 6
        MOV AL,[BX]
```

```
        MOV BX, OFFSET CLAVE
        CMP AL,[BX]
        JNZ NOPO
        MOV BX, OFFSET CAR1
        MOV AL, OFFSET FIN1-OFFSET CAR1
        INT 7
        JMP FINAL
NOPO: MOV BX, OFFSET CAR2
        MOV AL, OFFSET FIN2-OFFSET CAR2
        INT 7
FINAL: INT 0
        END
```

Acá mostramos el ej. con una clave de 4 caracteres

```
ORG 1000H
MSJ       DB   "INGRESE CLAVE"
FIN        DB   ?
CLAVE   DB    "CASI"
ING        DB   ?
           DB  ?
           DB  ?
           DB  ?
CAR1     DB  "ACCESO PERMITIDO"
FIN1       DB    ?
CAR2     DB  "ACCESO DENEGADO"
FIN2       DB   ?

            ORG 2000H
            MOV BX, OFFSET MSJ
            MOV AL, OFFSET FIN-OFFSET MSJ
            INT 7
           MOV  CL,4
           MOV BX, OFFSET ING
SIGO:      INT 6
            INC  BX
            DEC  CL
           JNZ  SIGO
           MOV DX, OFFSET ING
           MOV CX, OFFSET CLAVE
           MOV AH, 4
MAS :       MOV BX, CX
           MOV AL, [BX]
           MOV BX, DX
           CMP AL, [BX]
```

```
            JNZ ACCDEN
             INC DX
             INC CX
             DEC AH
            JNZ MAS
            MOV BX, OFFSET CAR1
            MOV AL, OFFSET FIN1-OFFSET CAR1
             INT 7
            JMP FINAL
ACCDEN: MOV BX, OFFSET CAR2
             MOV AL, OFFSET FIN2-OFFSET CAR2
             INT 7
FINAL:       INT 0
             END
```

11) y 13) En archivos aparte.