```
TITLE zip.asm - per compito 02/04/2007

COMMENT *

Si vogliono comprimere N=64 caratteri di ingresso
appartenenti all'alfabeto {'A','B','C','D'} con
la seguente legge di codifica:

                            1 <- 'A'
                           01 <- 'B'
                          001 <- 'C'
                         0001 <- 'D'

I byte vanno compressi uno alla volta, ponendo i dati
codificati nel buffer di memoria ENCBUFF. Il programma,
dopo aver acquisito da I/O il byte da comprimere
ed averlo posto in AL, scrive il risultato della codifica
in DX. Quando DH e' tutto pieno, il suo contenuto viene
salvato in ENCBUFF, mentre eventuali bit residui di codifica
rimangono in DL.

*

;-----------------------------------------------------------------
DATA_PORT EQU   1Ah
STATUS_PORT EQU 1Bh


;-----------------------------------------------------------------
STACK SEGMENT STACK 'STACK'    ; definizione del segmento di stack
      DB      16 DUP('STACK')  ; lo stack e' riempito con la stringa 'stack'
STACK ENDS


;-----------------------------------------------------------------
DATA SEGMENT PUBLIC 'DATA'           ; definizione segmento dati

    Nuncompressed equ 64
    Ncompressed EQU  Nuncompressed/2                    ; caso peggiore (tutte 'D')
    ENCBUFF DB Ncompressed dup(?)
    Ncaratteri equ 4
    caratteri db 'A','B','C','D'
    codici db 10000000B,01000000B,00100000B,00010000B     ; cambiare qui per cambiare legge di codifica
    lunghezze db 1,2,3,4
    Nvuoti db 8                                         ; # di bit di codifica restanti per riempire del tutto DH
    residuo db 0                                        ; cio' che rimane in DL dopo lo shift del codice in DH

DATA ENDS


;-----------------------------------------------------------------
```

```
CSEG SEGMENT PUBLIC 'CODE'

MAIN PROC FAR
      ASSUME cs:cseg,DS:data;

        MOV AX,DATA             ; necessari 2 trasferimenti per assegnare a DS
        MOV DS,AX               ; l'indirizzo del segmento dati in modo esplicito

        mov bx,0                ; # CARATTERI DI INGRESSO GIA' CODIFICATI
        mov di,0                ; indice del buffer ENCBUFF con i caratteri codificati

ciclo_principale:

ACQUISIZIONE_carattere:
            in AL,STATUS_PORT           ; LETTURA DELLO STATO: NUOVO DATO DISPONIBILE SE LSB=1
            TEST AL,1
            JNE ACQUISIZIONE_CARATTERE
            IN AL,DATA_PORT             ; LETTURA DI UN CARATTERE DAL DISPOSITIVO DI INGRESSO

codifica_singolo_carattere:

            MOV SI,0
confronta_carattere:
            MOV AH,caratteri[SI]
            CMP AL,AH
            JE carattere_trovato
            inc si
            cmp si,Ncaratteri
            jl confronta_carattere

carattere_trovato:                  ; qui si usa il valore del registro SI
                                    ; corrispondente al carattere trovato per puntare al
                                    ; giusto codice e alla giusta lunghezza di codifica
            mov ch,lunghezze[SI]
            mov dl,codici[SI]
            mov cl,ch               ; questo valore di cl e' usato se lunghezze[SI] <= Nvuoti
            mov residuo,0           ; questo valore di residuo e' usato se lunghezze[SI] <= Nvuoti
            cmp ch,Nvuoti
            jle scrivi_in_dh
codice_troppo_lungo:
            mov cl,Nvuoti
            sub ch,Nvuoti           ; un residuo di lunghezze[SI]-Nvuoti bit resta in DL
                                    ; e va scritto in DH in un secondo momento,
                                    ; dopo la sua scrittura in memoria
            mov residuo,ch
scrivi_in_dh:
            shl dx,cl
            sub Nvuoti,cl

            cmp Nvuoti,0
            jne gestisci_residuo
```

```
memorizza_dh:
            mov ENCBUFF[DI],dh
            inc DI
            mov Nvuoti,8

gestisci_residuo:
            mov cl,residuo          ; scrittura dell'eventuale residuo in dh:
            shl dx,cl               ; e' 0 nel caso lunghezze[SI] <= Nvuoti
            sub Nvuoti,cl


            INC BX
            CMP BX,Nuncompressed    ; condizione di terminazione ciclo principale
            JE check_ultimo_byte
            JMP ciclo_principale

check_ultimo_byte:
             cmp Nvuoti,8           ; se Nvuoti=8 non c'e' piu' nulla da salvare in memoria
             je finish
             mov cl,Nvuoti
             shl dx,cl
             MOV ENCBUFF[DI],Dh     ; scrittura degli ultimi bit codificati

FINISH:

        MOV AH,4CH              ; si restituisce il controllo al DOS
        INT 21h

MAIN ENDP

CSEG ENDS

END MAIN                    ; il programma comincia all'indirizzo di MAIN
```