Assembly program 06

;Accept a character and show a message: Capital letter, Small letter, Digit, Somthing else

```
.MODEL  small
.STACK  100h
.DATA
 msg1      db 13,10,'Enter a character : $'
 msg2      db 13,10,'Capital letter $'
 msg3      db 13,10,'Small letter $'
 msg4      db 13,10,'Digit $'
 msg5      db 13,10,'Something else $'
 msg6      db 13,10,'Hit any key to exit $'
 char       db 0

.CODE
            mov AX, @data
            mov DS, AX

            lea DX,msg1                ;Show msg1
            mov AH,09h
            int 21h

            mov AH, 01h                ;Read a character
            int 21h
            mov char,AL

            cmp char,'0'               ;If character less then 0 :  Somthing else
             jb  other

            cmp char,'9'               ;If character is between 0 -9 : Digit
             jle digit

             cmp char,'A'              ;If    '9' < char < 'A' : Other
              jb  other

             cmp char,'Z'               ;Show Capital letter
              jle upper

             cmp char,'a'              ;If    'Z' < char < 'a' : Other
              jb  other

             cmp char,'z'              ;show Small letter
             jle lower

            jmp other                   ;Show other

lower:    lea DX,msg3              ;Show small letter
          mov AH,09h
          int 21h
          jmp exit
```

upper: lea DX,msg2 ;Show capital letter	37
mov AH,09h	38
int 21h	39
jmp exit	40
digit: lea DX,msg4 ;Show Digit	41
mov AH,09h	42
int 21h	43
jmp exit	44
other: lea DX,msg5 ;Show Somthing else	45
mov AH,09h	46
int 21h	47
jmp exit	48
exit: lea DX,msg6 ;Show msg6 on screen	49
mov AH,09h	50
int 21h	51
mov AH, 01h ;Read a character	52
int 21h	53
mov AH, 4Ch ;End program	54
int 21h	55
END	56

Assembly program 07

```
;Accept  two  digits and show its product
.MODEL   small
.STACK   100h
.DATA
 msg1       db 13,10,'Enter two numbers 0 -9 : $'
 msg2       db 13,10,'Invalid Digit $'
 msg3       db '  *   =       $'
 msg4       db 13,10,'Hit any key to exit $'
 dig1       db 0
 dig2       db 0
 char1      db 0
 char2      db 0
 crlf       db 13,10,'$'

.CODE
            mov AX, @data
            mov DS, AX

            lea DX,msg1              ;Show msg1
            mov AH,09h
            int 21h

            mov AH, 01h              ;Read 1st digit
            int 21h
            mov dig1,AL

            cmp dig1,'0'             ;If character is less then 0 - invalid
            jb  invalid

            cmp dig1,'9'             ;If character is greater then 9 - invalid
            ja  invalid

            mov AH,02h               ;Put a space after 1st dig
            mov DL,' '
            int 21h

            mov AH, 01h              ;Read 2nd digit
             int 21h
             mov dig2,AL

           cmp dig2,'0'              ;If character is less then 0 - invalid
           jb  invalid

           cmp dig2,'9'              ;If character is greater then 9 - invalid
           ja  invalid

           sub dig1,'0'              ;Change digit from ascii to binary
           sub dig2,'0'
```

mov AL,dig1 ;Multiply dig1 by dig2, result in AX	38
mov BL,dig2	39
mul BL	40
mov AH,0	41
mov BL,10 ;Divide result (in AX) by 10, result in AL,	42
div BL	43
mov char1,AL ;Put 1st digit of result in char1	44
add char1,'0'	45
mov char2,AH ;Put 2nd digit of result in char2	46
add char2,'0'	47
lea DX,crlf ;Skip new line	48
mov AH,09h	49
int 21h	50
lea BX,msg3 ;Put address of msg3 in BX	51
mov AL,dig1 ;Put dig1 in AL	52
add AL,'0' ;Change dig1 from binary to ascii	53
mov [BX],AL ;Put dig1 in msg3	54
mov AL,dig2 ;Put dig2 in AL	55
add AL,'0' ;Change dig2 from binary to ascii	56
mov [BX+4],AL ;Put dig2 in msg3	57
mov AL,char1 ;Put 1st digit of result in msg3	59
mov [BX+8],AL	60
mov AL,char2 ;Put 2nd digit of result in msg3	61
mov [BX+9],AL	62
lea DX,msg3 ;Show output line msg3	63
mov AH,09h	64
int 21h	65
jmp exit ;Output line was shown, exit program	66
invalid: lea DX,msg2 ;Invalid digit	67
mov AH,09h	68
int 21h	69
exit: lea DX,msg4 ;Show msg3 on screen	70
mov AH,09h	71
int 21h	72
mov AH, 01h ;Read a character	73
int 21h	74
mov AH, 4Ch ;End program	75
int 21h	76
END	77

Assembly program 08

```
;Loop - Accept a letter a-z and show all sequence from this letter to z

.MODEL    small
.STACK    100h
.DATA
 msg1       db 13,10,'Enter a letter a-z : $'
 msg2       db 13,10,'Invalid input $'
 msg3       db 13,10,'Hit any key to exit $'
 crlf       db 13,10,'$'
 char       db 0

.CODE
            mov AX, @data
            mov DS, AX

getchar:    lea DX,msg1             ;Show msg1
            mov AH,09h
            int 21h

            mov AH, 01h             ;Read a letter
            int 21h
            mov char,AL

            cmp char,'a'            ;If character is less then 'a' - invalid
             jb  invalid

             cmp char,'z'           ;If character is greater then 'z' - invalid
             ja  invalid

             lea DX,crlf            ;Skip to new line
             mov AH,09h
             int 21h

nextchar:   cmp char,'z'            ;Check if last char was shown
             ja  exit               ;Exit program

             mov AH,02h             ;Put a space after 1st dig
             mov DL,char
             int 21h

             mov AH,02h             ;Put a space after 1st dig
             mov DL,' '
             int 21h

            inc char                  ;Put next char
            jmp nextchar

invalid:   lea DX,msg2               ;Show msg invalid input
```

	mov AH,09h		36
	int 21h		37
	lea DX,crlf	;Skip to new line	38
	mov AH,09h		39
	int 21h		40
	jmp getchar	;Invalid input, try again	41
exit:	lea DX,msg3	;Show msg3 on screen	42
	mov AH,09h		43
	int 21h		44
	mov AH, 01h	;Read a character	45
	int 21h		46
	mov AH, 4Ch	;End program	47
	int 21h		48
END			49

Assembly program 09

```
;Accept string of 20 chracters and show it on screen

.MODEL  small
.STACK  100h
.DATA
msg1       db 13, 10, 'Hit any key to exit', 13, 10, '$'
msg2       db 13, 10, 'Enter a  string, # to exit', 13, 10, '$'
msg3       db 13, 10, 'The string entered was : $'
outstr     db 21 dup(0)
.CODE
           mov AX, @DATA
           mov DS, AX

nextstr:   lea  DX, msg2            ;Display msg "Enter a 5 chars
           mov  AH, 09h
           int  21h

           lea  BX,outstr           ;Point to output string outstr
           mov  CL, 1               ;Initialize counter

          mov  AH, 01h
next:     int  21h                   ;Accept a character from user

          cmp  AL,'#'                ;Finish loop and stop program if
           je   exit

          mov  [BX], AL             ;Put character accepted in the
          inc  BX                   ;Point to next position in output
          inc  CL                   ;Increase counter
          cmp  CL,20                 ;Check if all 20 characters
          jna  next                  ;If not jump to get the next

          mov  [BX], '$'              ;Put the dollar sign at the end of
          lea  DX, msg3              ;Display "The string entered was
          mov  AH, 09h               ;Display msg "The string
          int  21h

          lea  DX, outstr            ;Display the string entered
         mov  AH, 09h
          int  21h
          jmp  nextstr              ;Jump to get the next string

 exit:   lea  DX, msg1             ;Display msg "Hit any key to
         mov  AH, 09h
         int  21h

         mov AH, 01h              ;Accept any key
          int 21h

          mov AH, 4ch            ;Return control to the operating
          int 21h
END
```

Assembly program 10	
;Accept 20 characters and show it in inverse order	1
.MODEL small	2
.STACK 100h	3
.DATA	4
array db 21dup(0)	5
msg1 db 13, 10, 'Enter 20 characters', 13, 10, '$'	6
msg2 db 13, 10, 'Display array in inverse order:', 13, 10, '$'	7
msg3 db 13, 10, 'Hit any key to exit', 13, 10, '$'	8
crlf db 13, 10, '$'	9
mone dw 0	10
temp db 0	11
.CODE	12
mov AX,@DATA	13
mov DS,AX	14
	15
mov mone,1	16
lea DX,msg1 ;"Enter 20 characters"	17
mov AH,09h	18
int 21h	19
	20
getNextChar: cmp mone,20	21
ja showInverse	22
	23
mov AH,01h ; Get a character	24
int 21h	25
	26
lea BX,array ; Insert character in array	27
add BX,mone	28
mov [BX],AL	29
	30
inc mone ; Increase counter	31
	32
jmp getNextChar ; Get next character	33
	34
showInverse: mov mone,20	35
lea DX,msg2 ; "Display array in inverse order:"	36
mov AH,09h	37
int 21h	38
	39
showNextChar: cmp mone,1	40
jb exit	41
	42
lea BX,array ; Show a character	43
add BX,mone	44
mov DL,[BX]	45
mov AH,02h	46
int 21h	47

```
                sub mone,1              ; decrease counter

                jmp showNextChar        ; Get next character

exit:           lea DX,msg3             ;"Hit any key to exit"
                mov AH,09h
                int 21h

                mov AH,01h              ;Get a character
                int 21h

                mov AH,4Ch              ;Return control to operating system
                int 21h
END
```

Assembly program 11	
;Call routine	1
;Accept 2 characters and show them, loop ends when % is accepted in first char	2
.MODEL small	3
.STACK 100h	4
.DATA	5
msg1 db 13,10, 'Enter 1st char, % to quit : $'	6
msg2 db 13,10, 'Enter 2nd char : $'	7
msg3 db 13,10, 'Hit any key to exit $'	8
crlf db 13,10, '$'	9
chr1 db 0	10
chr2 db 0	11
.CODE	12
mov AX,@data	13
mov DS,AX	14
mainLoop: call getchar1 ; get 1st character	15
cmp chr1,'%' ; check if % was accepted	16
je exit	17
call getchar2 ; get 2nd character	18
call show	19
jmp mainLoop	20
exit: lea DX,msg3 ;"Hit any key to quit"	21
mov AH,09h	22
int 21h	23
mov AH,01h ;accept an any character	24
int 21h	25
mov AH,4ch ;return to operating system	26
int 21h	27
getchar1: lea DX,msg1 ;"Enter 1st char :"	28
mov AH,09h	29
int 21h	30
mov AH,01h ; accept char1	31
int 21h	32
mov chr1,AL	33
ret	34
getchar2: lea DX,msg2 ;"Enter 2nd char :"	35
mov AH,09h	36
int 21h	37
mov AH,01h ;accept char2	38
int 21h	39
mov chr2,AL	40
ret	41

show:	lea DX,crlf	;skip to new line	42
	mov AH,09h		43
	int 21h		44
	mov DL,chr1	;display char1	45
	mov AH,02h		46
	int 21h		47
	mov DL,'-'	;display '-'	48
	mov AH,02h		49
	int 21h		50
	mov DL,chr2	;display char2	51
	mov AH,02h		52
	int 21h		53
	ret		54
END			55

Assembly program 12	
;call routine and move parameter by stack	1
	2
;Showing the triangle	3
.MODEL small	4
.STACK 100h	5
.DATA	6
msg1 db 13, 10, 'Hit any key to exit $'	7
crlf db 13, 10, '$'	8
len db 0	9
mone dw 0	10
.CODE	11
mov AX, @data	12
mov DS, AX	13
mov mone,10 ;set 1st line to 10 asterics	14
nextLine: mov BX, mone	15
push BX ;push parameter value to stack	16
call prtLine ;call print line routne	17
mov BX, mone	18
dec BX ;decriment num of asterics	19
mov mone, BX	20
jnz nextLine ;jump to print next line	21
lea DX, msg1 ;triangle was printed, exit program	22
mov AH, 09h	23
int 21h	24
mov AH, 01h	25
int 21h	26
mov AH, 4ch	27
int 21h	28
prtLine: pop AX ;pop return address	29
pop BX ;pop routine parameter - line length	30
push AX ;push return address	31
mov len,BL ;get line length	32
lea DX, crlf ;start new line	33
mov AH, 09h	34
int 21h	35
mov DL, '*'	36
mov AH, 02h	37
prtChar: int 21h ;display '*'	38
dec BL	39
jnz prtChar	40
ret	41
END	42

הוראות אסמבלי

	פקודה	C	S	Z	הערות
	הוראות אריתמטיות				
1	MOV opnd1, opnd2				משים את ערכו של opnd2 בתוך opnd1
2	ADD opnd1, opnd2	√	√	√	מוסיף ל opnd1 את ערכו של opnd2
3	SUB opnd1, opnd2	√	√	√	מחסר מ opnd1 את ערכו של opnd2
4	INC opnd1		√	√	מוסיף 1 לערכו של opnd1
5	DEC opnd1		√	√	מחסר 1 מערכו של opnd1
6	CMP opnd1, opnd2	√	√	√	משווה את ערכיהם של שני האופרנדים (בפועל הוא מבצע את פעולה 3 מבלי לשנות את ערכי האופרנדים)
7 בית	MOV AL, 30 MOV BL, 4 MUL BL				מבצע פעולת כפל: 30 x 4 = 120 המכפלה AX = 120
7 מילה	MOV AX, 125 MOV BX, 200 MUL BX				מבצע פעולת כפל: 125 x 200 = 120 המכפלה 25000 = (DX AX)
8 בית	MOV AX, 205 MOV BL, 30 DIV BL				מבצע פעולת חילוק: (25) 6 = 30 : 205 המנה AL = 6, השארית AH = 25
8 מילה	MOV DX,0 MOV AX,65012 MOV CX,5000 DIV CX				מבצע פעולת חילוק של שני מספרים בני 16 ביטים המנה AX = (DX AX) / operand = 13 השארית DX = 12
					הוראות קפיצה
9	JMP label				מבצע קפיצה ללא תנאי
10	JZ / JE label				מבצע קפיצה אם דגל האפס דלוק
11	JNZ / JNE label				מבצע קפיצה אם דגל האפס מכובה
12	JS label				מבצע קפיצה אם דגל הסימן דלוק
13	JNS label				מבצע קפיצה אם דגל הסימן מכובה
14	JC label				מבצע קפיצה אם דגל הנשא דלוק
15	JNC label				מבצע קפיצה אם דגל הנשא מכובה
16	JA / JNBE label				מבצע קפיצה אם לאחר ההשוואה
17	JAE / JNB label				מבצע קפיצה אם לאחר ההשוואה

18	JB / JNAE label				מבצע קפיצה אם לאחר ההשוואה
19	JBE / JNA label				מבצע קפיצה אם לאחר ההשוואה
	הוראות לוגיות				
20	NOT opnd1				הופך את ערכי הסיביות של opnd1
21	AND opnd1, opnd2	0	√	√	מפעיל את האופרטור AND על פי טבלת האמת. התוצאה ב opnd1
22	OR opnd1, opnd2	0	√	√	מפעיל את האופרטור OR על פי טבלת האמת. התוצאה ב opnd1
23	XOR opnd1, opnd2	0	√	√	מפעיל את האופרטור XOR על פי טבלת האמת. התוצאה ב opnd1
	הוראות סיבוב				
24	SHL opnd1, num	√	√	√	הזזת הסיביות של opnd1 שמאלה num פעמים
25	SHR opnd1, num	√	√	√	הזזת הסיביות של opnd1 ימינה num פעמים
26	ROL opnd1, num	√			הזזת הסיביות של opnd1 במעגל שמאלה num פעמים
27	ROR opnd1, num	√			הזזת הסיביות של opnd1 במעגל ימינה num פעמים
28	RCL opnd1, num	√			הזזת הסיביות של opnd1 במעגל שמאלה דרך הנשא num פעמים
29	RCR opnd1, num	√			הזזת הסיביות של opnd1 במעגל ימינה דרך הנשא num פעמים