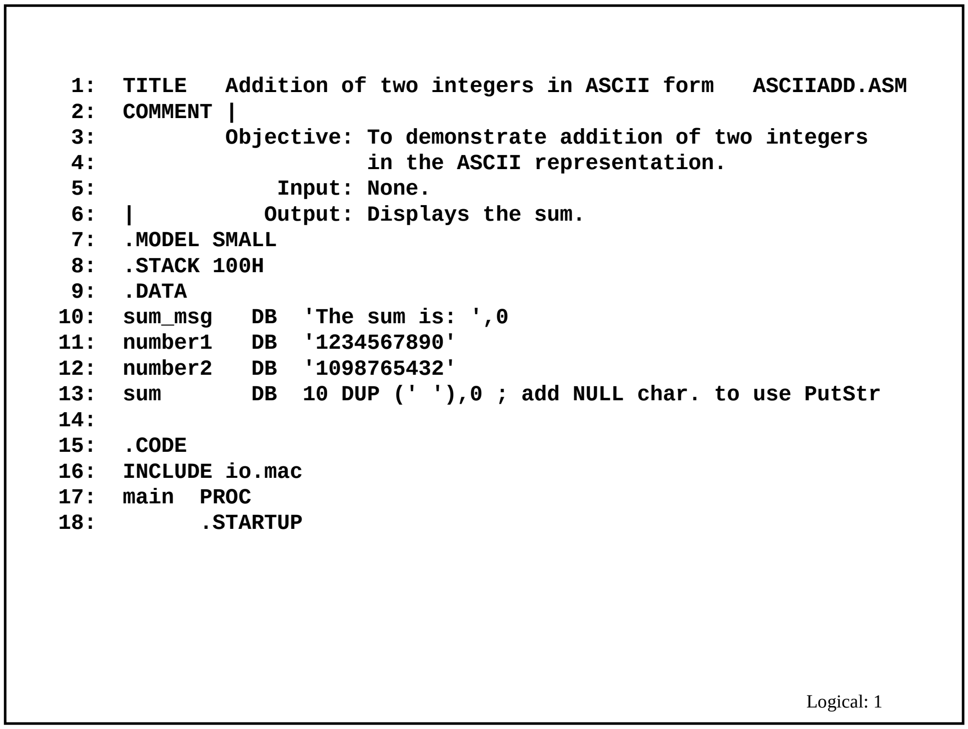
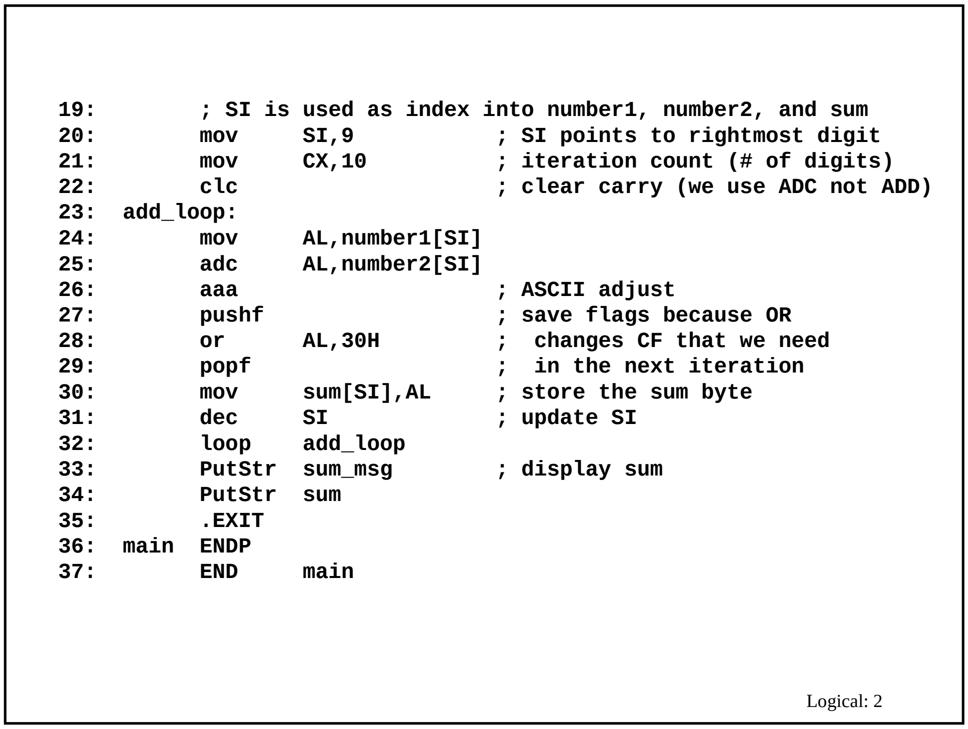


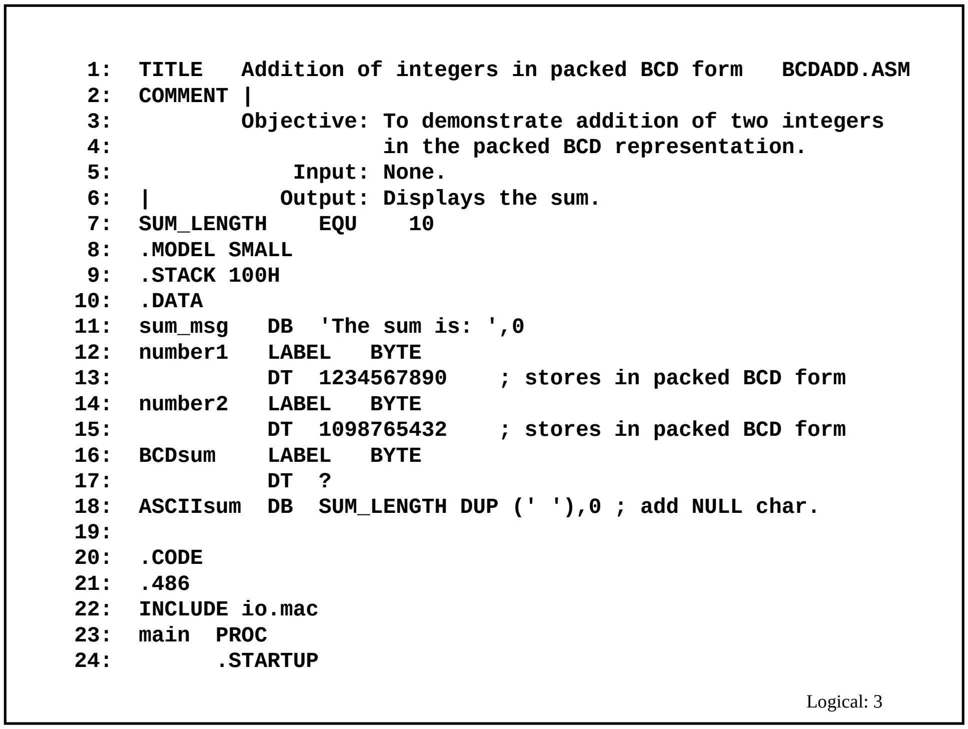
```
TITLE   Addition of two integers in ASCII form   ASCIIADD.ASM
COMMENT |
        Objective: To demonstrate addition of two integers
                   in the ASCII representation.
            Input: None.
|          Output: Displays the sum.
.MODEL SMALL
.STACK 100H
.DATA
sum_msg   DB  'The sum is: ',0
number1   DB  '1234567890'
number2   DB  '1098765432'
sum       DB  10 DUP (' '),0 ; add NULL char. to use PutStr

.CODE
INCLUDE io.mac
main  PROC
      .STARTUP
```

Logical: 1



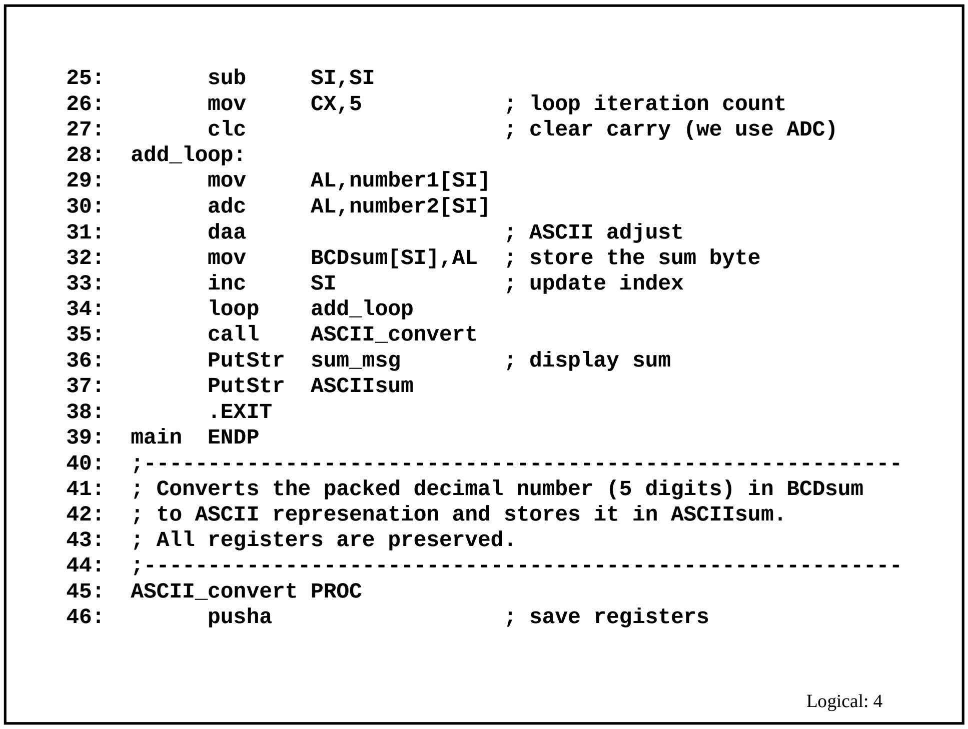
```
      ; SI is used as index into number1, number2, and sum
      mov     SI,9           ; SI points to rightmost digit
      mov     CX,10          ; iteration count (# of digits)
      clc                    ; clear carry (we use ADC not ADD)
add_loop:
      mov     AL,number1[SI]
      adc     AL,number2[SI]
      aaa                    ; ASCII adjust
      pushf                  ; save flags because OR
      or      AL,30H         ;  changes CF that we need
      popf                   ;  in the next iteration
      mov     sum[SI],AL     ; store the sum byte
      dec     SI             ; update SI
      loop    add_loop
      PutStr  sum_msg        ; display sum
      PutStr  sum
      .EXIT
main  ENDP
      END     main
```

Logical: 2



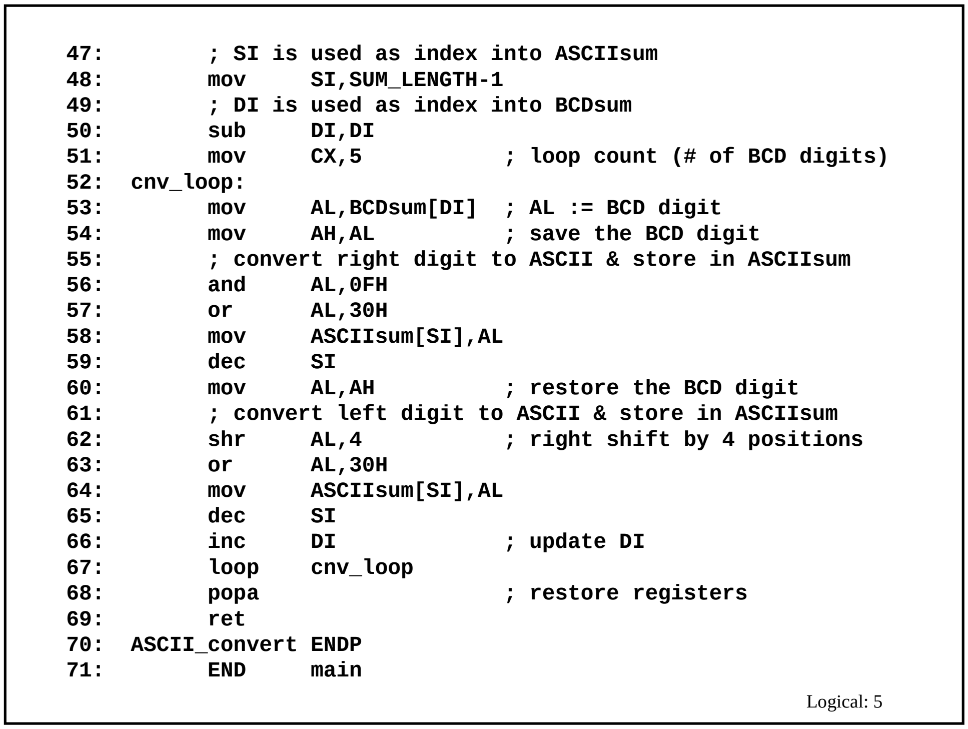
```
TITLE   Addition of integers in packed BCD form   BCDADD.ASM
COMMENT |
        Objective: To demonstrate addition of two integers
                   in the packed BCD representation.
            Input: None.
|          Output: Displays the sum.
SUM_LENGTH    EQU    10
.MODEL SMALL
.STACK 100H
.DATA
sum_msg   DB  'The sum is: ',0
number1   LABEL   BYTE
          DT  1234567890    ; stores in packed BCD form
number2   LABEL   BYTE
          DT  1098765432    ; stores in packed BCD form
BCDsum    LABEL   BYTE
          DT  ?
ASCIIsum  DB  SUM_LENGTH DUP (' '),0 ; add NULL char.

.CODE
.486
INCLUDE io.mac
main  PROC
      .STARTUP
```

Logical: 3



```
      sub     SI,SI
      mov     CX,5           ; loop iteration count
      clc                    ; clear carry (we use ADC)
add_loop:
      mov     AL,number1[SI]
      adc     AL,number2[SI]
      daa                    ; ASCII adjust
      mov     BCDsum[SI],AL  ; store the sum byte
      inc     SI             ; update index
      loop    add_loop
      call    ASCII_convert
      PutStr  sum_msg        ; display sum
      PutStr  ASCIIsum
      .EXIT
main  ENDP
;------------------------------------------------------------
; Converts the packed decimal number (5 digits) in BCDsum
; to ASCII represenation and stores it in ASCIIsum.
; All registers are preserved.
;------------------------------------------------------------
ASCII_convert PROC
      pusha                  ; save registers
```

Logical: 4



```
      ; SI is used as index into ASCIIsum
      mov     SI,SUM_LENGTH-1
      ; DI is used as index into BCDsum
      sub     DI,DI
      mov     CX,5           ; loop count (# of BCD digits)
cnv_loop:
      mov     AL,BCDsum[DI]  ; AL := BCD digit
      mov     AH,AL          ; save the BCD digit
      ; convert right digit to ASCII & store in ASCIIsum
      and     AL,0FH
      or      AL,30H
      mov     ASCIIsum[SI],AL
      dec     SI
      mov     AL,AH          ; restore the BCD digit
      ; convert left digit to ASCII & store in ASCIIsum
      shr     AL,4           ; right shift by 4 positions
      or      AL,30H
      mov     ASCIIsum[SI],AL
      dec     SI
      inc     DI             ; update DI
      loop    cnv_loop
      popa                   ; restore registers
      ret
ASCII_convert ENDP
      END     main
```

Logical: 5