PROGRAM: 1

```
data segment
     m1 db 0ah,0dh,"Enter the number:$"
     m2 db 0ah,0dh,"the number is even$"
     m3 db 0ah,0dh,"the number is odd$"
data ends

code segment

assume cs:code,ds:data
start:
      mov ax,data
      mov ds,ax
      lea dx,m1
      mov ah,09h
      int 21h
      mov ah,01h
      int 21h
      add al,48
      rcr al,1
      jc odd
      lea dx,m2
      mov ah,09h
      int 21h
      jmp stop
      odd:lea dx,m3
      mov ah,09h
      int 21h
      stop:mov ah,4ch
      int 21h
code ends
end start
```

OUTPUT:

ENTER THE NUMBER: **5**
THE NUMBER IS ODD

ENTER THE NUMBER: **2**
THE NUMBER IS EVEN

PROGRAM: 2

```
data segment
  msg1 db 0ah,0dh,"first no$"
  msg2 db 0ah,0dh,"second no$"
  msg3 db 0ah,0dh,"result:$"
  n1 db 07h dup(?)
  n2 db 07h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm

readDigit macro
mov ah,01h
int 21h
sub al,30h
endm

printDigit macro
add dl,30h
mov ah,02h
int 21h
endm

code segment

assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov si,offset n1
mov di,offset n2
display msg1
mov cl,04h

first:
readDigit
mov [si],al
inc si
dec cl
jnz first
display msg2
mov cl,04h
```

```
second:
readDigit
mov [di],al
inc di
dec cl
jnz second
clc
mov cl,04h
addition:
dec di
dec si
mov al,[si]
mov bl,[di]
adc al,bl
mov ah,00h
aaa
mov [di],al
dec cl
jnz addition
display msg3
mov cl,04h
print:
mov dl,[di]
printDigit
inc di
dec cl
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

FIRST NO: **1111**
SECOND NO: **2560**

RESULT: **3671**

PROGRAM: 3

```
data segment
  msg1 db 0ah,0dh,"first no:$"
  msg2 db 0ah,0dh,"seconf no:$"
  msg3 db 0ah,0dh,"result:$"
  n1 db 07h dup(?)
  n2 db 07h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm

readDigit macro
mov ah,01h
int 21h
sub al,30h
endm

printDigit macro
add dl,30h
mov ah,02h
int 21h
endm

code segment

assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov si,offset n1
mov di,offset n2
display msg1
mov cx,04h

first:
readDigit
mov [si],al
inc si
dec cx
jnz first
display msg2
mov cx,04h
second:
readDigit
mov [di],al
```

```
inc di
dec cx
jnz second
clc
mov cx,04h
subtraction:
dec di
dec si
mov al,[si]
mov bl,[di]
sbb al,bl
mov ah,00h
aas
mov [di],al
dec cx
jnz subtraction
display msg3
mov cx,04h
print:
mov dl,[di]
printDigit
inc di
dec cx
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

FIRST NO: **3890**
SECOND NO: **1243**

RESULT: **2647**

PROGRAM: 4

```
data segment
  msg1 db 0ah,0dh,"first no$"
  msg2 db 0ah,0dh,"second no$"
  msg3 db 0ah,0dh,"result:$"
  n1 db 09h dup(?)
  n2 db 09h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm

readDigit macro
mov ah,01h
int 21h
sub al,30h
endm

printDigit macro
add dl,30h
mov ah,02h
int 21h
endm

code segment

assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov si,offset n1
mov di,offset n2
display msg1
mov cx,08h

first:
readDigit
mov [si],al
inc si
dec cx
jnz first
display msg2
mov cx,08h
second:
readDigit
mov [di],al
```

```
inc di
dec cx
jnz second
clc
mov cx,08h
addition:
dec di
dec si
mov al,[si]
mov bl,[di]
adc al,bl
mov ah,00h
aaa
mov [di],al
dec cx
jnz addition
display msg3
mov cx,08h
print:
mov dl,[di]
printDigit
inc di
dec cx
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

FIRST NUMBER: **34562310**
SECOND NUMBER: **14567890**

RESULT: **49130200**

PROGRAM: 5

```
data segment
  msg1 db 0ah,0dh,"first no$"
  msg2 db 0ah,0dh,"second no$"
  msg3 db 0ah,0dh,"result:$"
  n1 db 09h dup(?)
  n2 db 09h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm

readDigit macro
mov ah,01h
int 21h
sub al,30h
endm

printDigit macro
add dl,30h
mov ah,02h
int 21h
endm

code segment

assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov si,offset n1
mov di,offset n2
display msg1
mov cx,08h

first:
readDigit
mov [si],al
inc si
dec cx
jnz first
display msg2
mov cx,08h
second:
readDigit
mov [di],al
```

```
inc di
dec cx
jnz second
clc
mov cx,08h
subtraction:
dec di
dec si
mov al,[si]
mov bl,[di]
sbb al,bl
mov ah,00h
aas
mov [di],al
dec cx
jnz subtraction
display msg3
mov cx,08h
print:
mov dl,[di]
printDigit
inc di
dec cx
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

FIRST NUMBER: **65784302**
SECOND NUMBER: **32156789**

RESULT: **33627513**

PROGRAM: 6

```
data segment
   m1 db 0ah,0dh,"enter the string:$"
   m2 db 0ah,0dh,"enter the key:$"
   result1 db 0ah,0dh,"key found:$"
   result2 db 0ah,0dh,"key not found:$"
   array db 09h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm
readcharacter macro
mov ah,01h
int 21h
endm

code segment

assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov si,offset array
display m1
mov cl,00h
stringScan:
readcharacter
cmp al,0dh
jz ended
mov [si],al
inc cl
inc si
jmp stringScan
ended:
display m2
readcharacter
mov bl,al
mov ch,00h
check:
dec si
cmp bl,[si]
jz found
dec cl
jnz check
jmp notfound
found:
```

```
display result1
jmp finish
notfound:
display result2
finish:
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

ENTER THE STRING: **2 3 5 6 7**
ENTER THE KEY: **8**

KEY NOT FOUND

ENTER THE STRING: **2 3 5 6 7**
ENTER THE KEY: **3**

KEY FOUND

PROGRAM: 7

```
data segment
   m1 db 0ah,0dh,"enter no of elements:$"
   m2 db 0ah,0dh,"enter the number:$"
   result db 0ah,0dh,"sorted array:$"
   n db ?
   array db 09h dup(?)
data ends

display macro msg
lea dx,msg
mov ah,09h
int 21h
endm

readdigit macro
mov ah,01h
int 21h
sub al,30h
endm

printdigit macro
add dl,30h
mov ah,02h
int 21h
endm

code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
display m1
readdigit
mov n,al
mov cl,n
display m2
mov si,offset array

read:
readdigit
mov [si],al
inc si
dec cl
jnz read
mov cl,n
loop1:
mov ch,n
mov si,offset array
```

```
loop2: mov dl,[si]
cmp dl,[si+1]

jnc swap
jmp swapped
swap:
mov dl,[si]
xchg dl,[si+1]
mov [si],dl
swapped:
inc si
dec ch
jnz loop2
dec cl
jnz loop1
display result
mov si,offset array
mov cl,n
inc si

print:
mov dl,[si]
printdigit
inc si
dec cl
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

ENTER NO OF ELEMENTS : **4**
ENTER THE NUMBER: **5 2 3 1**

SORTED ARRAY: **1 2 3 5**

ENTER NO OF ELEMENTS: **8**
ENTER THE NUMBER : **6 8 7 4 3 2 1 0**

SORTED ARRAY: **0 1 2 3 4 6 7 8**

PROGRAM: 8

```
assume cs:code,ds:data

data segment
msg1 db 0ah,0dh,"enter string:$"
msg2 db 0ah,0dh,"reverse:$"
str1 db 20 dup("?")
rev db 20 dup("?")
data ends

display macro msg
mov ah,09h
lea dx,msg
int 21h
endm
code segment
start: mov ax,data
       mov ds,ax
       display msg1
       mov cx,0000h
       lea si,str1
loop1: mov ah,01h
        int 21h
        mov [si],al
        cmp al,0dh
        je loop2
        inc si
        inc cx
        jmp loop1
loop2: mov bl,"$"
        mov [si],bl
        mov di,si
        lea si,rev
        dec di
loop3: mov bl,[di]
        mov [si],bl
        dec di
        inc si
        dec cx
        jnz loop3
        mov bl,"$"
        mov [si],bl
       display msg2
       display rev
       mov ah,4ch
       int 21h
       code ends
       end start
```

OUTPUT:

ENTER STRING: **CAR**

REVERSE : **RAC**

PROGRAM: 9

```
data segment
  msg1 db 0ah,0dh,"enter the string:$"
  msg2 db 0ah,0dh,"string is palindrome:$"
  msg3 db 0ah,0dh,"string is not palindrome:$"
  str1 db 50 dup("?")
data ends

display macro msg
mov ah,09h
lea dx,msg
int 21h
endm
code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
display msg1
mov cx,0000h
lea si,str1
loop1:
      mov ah,01h
      int 21h
      mov [si],al
      cmp al,0dh
      je loop2
      inc si
      inc cx
      jmp loop1
loop2:
      dec si
      lea di,str1
loop3:
       mov bl,[di]
       mov al,[si]
       cmp al,bl
       jnz loop4
       inc di
       dec si
       dec cx
       jnz loop3
       display msg2
       jmp loop5
loop4:
      display msg3
loop5:
       mov ah,4ch
```

```
int 21h
code ends
end start
```

OUTPUT:

ENTER THE STRING : **AMMA**

STRING IS PALINDROME

ENTER THE STRING : **CAR**

STRING IS NOT PALINDROME