
بسم الله الرحمن الرحيم

UNIVERSITY OF GUJRAT

A WORLD CLASS UNIVERSITY

UNIVERSITY OF GUJRAT

UOG

Topic:
Decimal Input and Output

Group member:

M NABEEL	(14093122-014)
TAYYAB SAEED	(14093122-015)
NASEER AHMAD	(14093122-016)
AQEEL HAIDER	(14093122-017)
MIR HAMZA MAJEED	(14093122-018)
HAMZA TAJ	(14093122-018)

Decimal Input and Output

- Computer represent every thing in binary.
- But it is convenient for user to represent input and output in decimal.
- If we input 21543 character string then it must to converted internally.
- Conversely on output the binary contents of R/M must be converted to decimal equivalent before being printed.

Decimal input

- Convert a string of ASCII digits to the binary representation of decimal equivalent
- For input we repeatedly multiply AX by 10

Algorithm (First version):

Total=0

Read an ASCII

REPEAT
convert character to number

Total=total*10+value

Read a character

Until character is carriage return

Cont..

Example: input of 123

Total =0

Read '1'

Convert '1' to 1

Total=10*0 +1=1

Read '2'

Convert '2' to 2

Total=10*1 +2=12

Read '3'

Convert '3' to 3

Total=10*12 +3=123

Cont..

Range: -32768 to 32767

Optional sign followed by string of digits & carriage return

Outside '0' to '9'

jumps to new line and ask for input again

Cont..

Algorithm(second version):

total=0

Negative=false

Read a character

Case character of

- '-': negative=true
  read a character
- '+': read a character

End_case

Repeat

If character is not between '0' to '9'

Then

Go to beginning

Cont..

Else

Convert character to binary value

Total=10*total+value

End_if

Read a character

Until character is carriage return

If negative =true

Then

total=-total

Cont..

Program(source code):

```
INDEC PROC
;READ NUMBER IN RANGE -32768 TO 32767
PUSH BX
PUSH CX
PUSH DX
@BEGIN:
;total =0
XOR BX,BX          ;BX hold total
;negative =false
```

Cont..

```
XOR CX,CX          ;CX hold sign
;read char
MOV AH,1
INT 21H
;case char of
CMP AL,'-'          ;minus sign
JE @MINUS           ;yes,set sign
CMP AL,'+'          ;plus sign
JE @PLUS            ;yes,get another char
```

Cont..

```
JMP @REPEAT2              ;start processing char
@MINUS: MOV CX,1
@PLUS: INT 21H
;end case
@REPEAT2:
;if char. is between '0' and '9'
CMP AL,'0'                ;char >='0'?
JNGE @NOT_DIGIT           ;illegal char.
```

Cont..

```
CMP AL,'9'                    ;char<='9' ?
JNLE @NOT_DIGIT
;then convert char to digit
AND AX,000FH
PUSH AX                       ;save number
;total =total *10 +digit
MOV AX,10
MUL BX
POP BX                        ;retrieve number
ADD BX,AX                     ;total =total *10 +digit
```

Cont..

```
;read char
MOV AH,1
INT 21H
CMP AL,0DH              ;CR
JNE @REPEAT2            ;no keep going
;until CR
MOV AX,BX               ;store number in AX
;if negative
OR CX,CX                ;negative number
```

Cont..

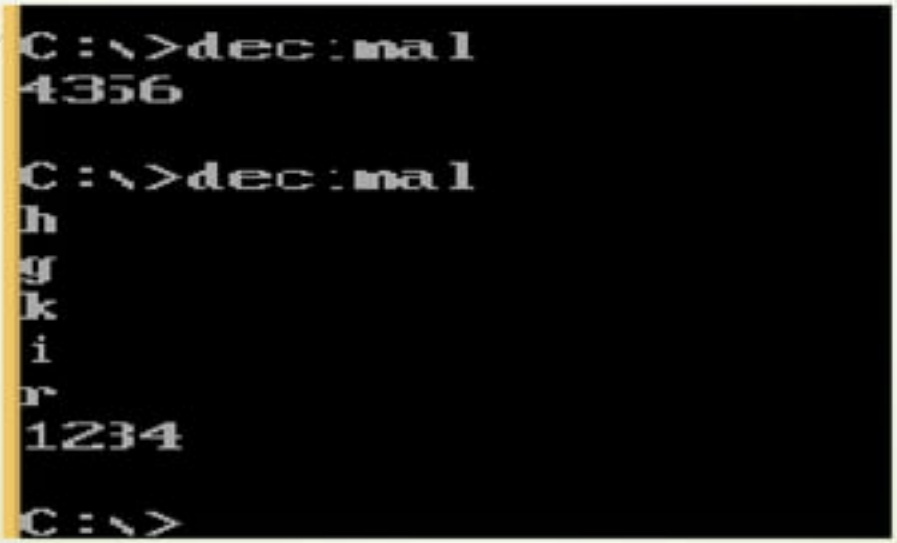
```
Jz @EXIT                    ;no,exit
;then
NEG AX                      ;yes,negate
;end if
@EXIT:                      ;retrieve registers
 POP DX
 POP CX
POP BX
RET
```

Cont..

```
;here if illegal char entered
@NOT_DIGIT:
MOV AH,2
MOV DL,0DH
INT 21H
MOV DL,0AH
INT 21H
JMP @BEGIN
INDEC ENDP
```

Cont..

Output:



Input Overflow

- AX:FFFFh
- In decimal:65535
- Range:-32768 to 32767
- Anything out of range called input overflow
- For example:
- Input:32769
- Total=327690

Cont..

Algorithm:

total=0

Negative=false

Read a character

Case character of

- '-': negative=true
  read a character
- '+': read a character

Cont..

End_case

Repeat

If character is not between '0' to '9'

Then

Go to beginning

Else

Convert character to binary value

Total=10*total

Cont..

If overflow

Then

go to beginning

Else

Total =total*10 +value

If overflow

Then

go to beginning

Cont..

End_if

End_if

End_if

Read a character

Until character is carriage return

If negative =true

Then

total=-total

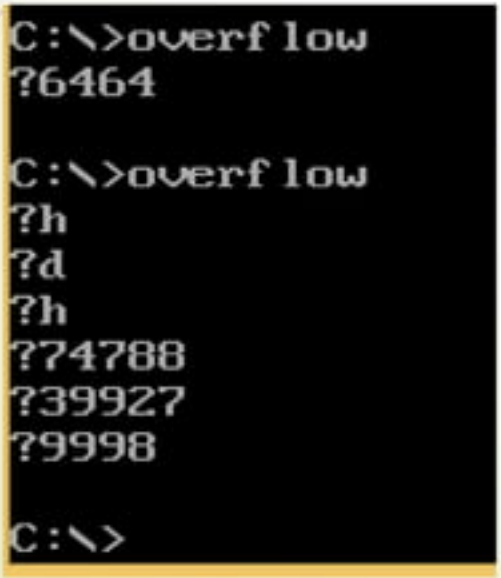
Cont..

Code:

```
;total =total *10 +digit
MOV AX,10
MUL BX
CMP DX,0
JNE @NOT_DIGIT
POP BX
ADD BX,AX
JC @NOT_DIGIT
```

Cont..

Output:



Decimal Output

Algorithm for Decimal Output:

- If AX < 0 /*AX holds output value */
- THEN
- Print a minus sign
- Replace AX by its twos complement
- End_IF
- Get the digits in AX's decimal representation
- Convert these digits into characters and print them

Cont..

To see what line 6 entitles, suppose the contents of AX, expressed in decimal is 24168. To get the digits in decimal representation , we can proceed as follows,

- Divide 24618 by 10, Quotient= 2461, remainder=8
- Divide 2461 by 10, Quotient= 246, remainder=1
- Divide 246 by 10 , Quotient=24, remainder=6
- Divide 24 by 10, Quotient=2, remainder=4
- Divide 2 by 10, Quotient=0, remainder=2

Cont..

LINE 6:

Cout =0 /*will count decimal digit */

REPEAT

divide quotient by 10

Push remainder on the stack

Count= count +1

UNTILL

Quotient=0

Cont..

LINE 7:

FOR count times DO

Pop a digit from the stack

Convert it to a character

Output the character

END_FOR

Cont..

Program Listing PMG9_1.ASM

```
.MODEL    SMALL
.STACK    100H
.CODE
OUTDEC   PROC
;prints AX as a signed decimal integer
;input: AX
;output: none
PUSH    AX                    ;save registers
PUSH    BX
PUSH    CX
PUSH    DX
```

Cont..

```
;if AX < 0
OR    AX,AX       ;AX < 0?
JGE   @END_IF1    ;NO  >0
;then
PUSH   AX         ; save number
MOV   DL,'-'      ;get '-'
MOV   AH,2        ;print character function
INT    21H        ;print '-'
POP    AX         ;get Ax back
NEG   AX          ;AX= -AX
@END_IF1:
```

Cont..

```
;get decimal digits
XOR     CX,CX       ;CX counts digits
MOV    BX,10D       ;BX has divisor
@REPEAT1:
XOR      DX,DX      ;prepare high word of dividend
DIV      BX         ;AX=quotient, DX=remainder
PUSH    DX          ;save remainder on stack
INC      CX         ;count = count +1
;until
OR       AX,AX      ;quotient = 0?
JNE     @REPEAT     ;no, keep going
```

Cont..

```
;convert digits to character and print
MOV       AH,2          ;print character function
;for count time do
@PRINT_LOOP
POP       DX            ;digit in DL
OR        DL,30H        ;convert to character
INT       21H           ;print digit
;end_for
POP       DX            ; restore registers
POP       CX
POP       BX
POP       AX
OUTDEC    ENDP
```


THANK YOU