UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Katedra za elektroniku

Predmet: Racunarska elektronika

Projekat: GUI Kalkulator

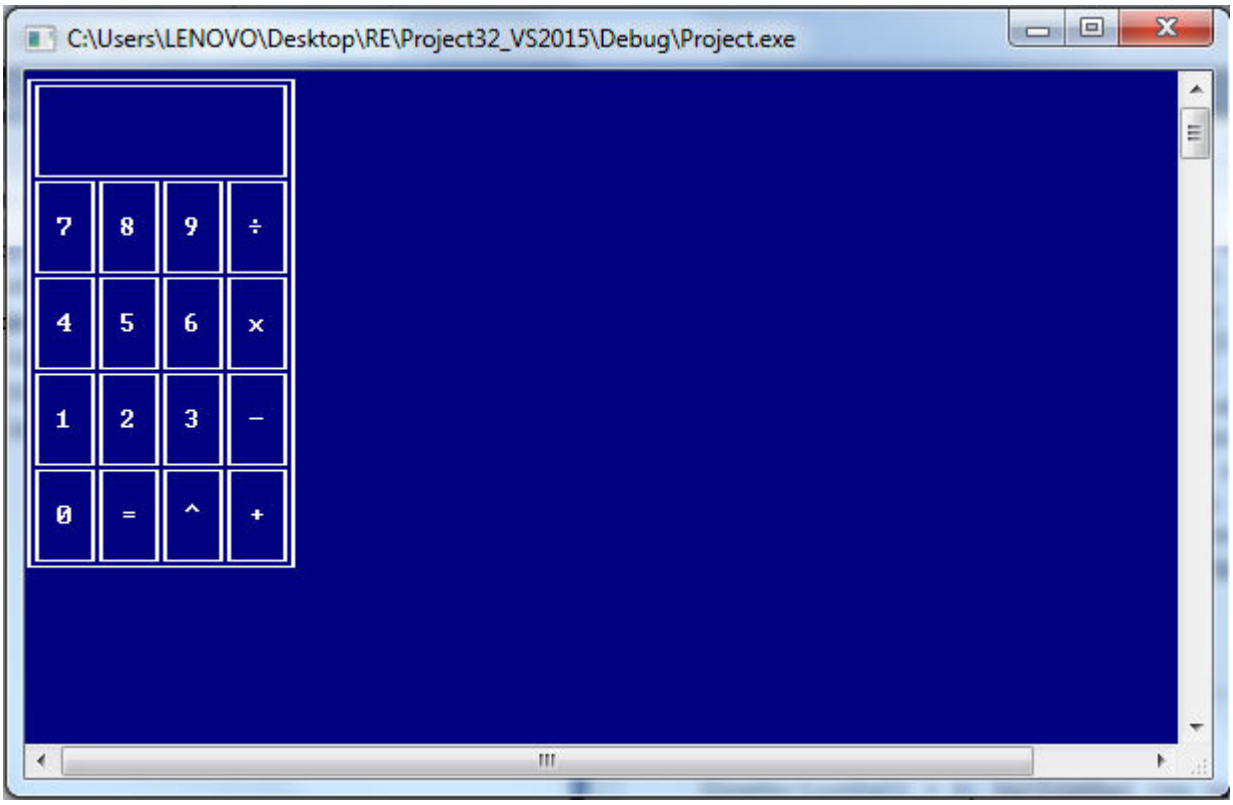
Projekat radili:

Ime	Prezime	broj indeksa
Petar	Pavlovic	188/2015
Luka	Adamovic	526/2015

Tekst zadatka:
Ideja je da se jednostavne računske operacije kao što su sabiranje, oduzimanje, množenje, deljenje i stepenovanje predstave preko lepo uređenog grafičkog interfejsa. Operacijama se pristupa unosom operanada i znakova operacije preko tastature (+, -, *, / i ^). Potrebno je formirati dugmiće na kojima će biti predstavljeni brojevi i operacije i displej na kome će se ispisivati rezultat u zavisnosti od unetih parametara.

Objasnjenje rada programa:

Na samom pocetku korisnika docekuje graficki prikaz kalkulatora sa svim dostupnim brojevima i svim dostupnim operacijama.



Graficki prikaz kalkulatora

Nakon unosa prvog operanda, pritiskom na taster *Enter* prelazimo na biranje operacije. Posle izbora operacije unosimo drugi operand, i pritiskom na taster *Enter* na displeju se ispisuje rezultat.

Potom, program prelazi u mod cekanja, gde se od korisnika ocekuje da pritisne bilo koji taster da bi restartovao displej i presao u pocetno stanje.

Prilikom unosa operanda ukoliko korisnik nije zadovoljan moze ga obrisati pre pritiska *Enter* tastera i to klasicno, pritiskom na taster *Backspace.* Ukoliko je korisnik slucajno pritisnuo taster *Enter*, a nije zadovoljan izabranim operandom, u pocetno stanje moze uci pritiskom na taster *R*.

Korisnik, ukoliko vise ne zeli da koristi kalkulator, potrebno je da nakon pocetnog stanja dva puta pritisne taster *Enter*, cime aktivira izlaz i zavrsava koriscenje programa.

Objasnjenje koda:

Program je napisan koristeci *Irvine32.inc* biblioteku i Irvinove preporuke za modele koji se koriste za rad u *VS2015*.

Na pocetku programa vrsi se definisanje i inicijalizacija konstanti koje predstavljaju kordinate za iscrtavanje interfejsa kalkulatora, kao i dimenzije prozora.

U **.data** sekciji definisani su svi stringovi, sve poruke koje korisnik dobija nakon nedozvoljenih operacija. Jos se u **.data** nalaze i promenljive koje program koristi.

U **.code** sekciji se najpre menja boja pozadine, a zatim se prelazi na iscrtavanje grafickog interfejsa kalkulatora. Uz pomoc fukcije **WriteChar**, heksadecimalnih vrednosti preuzetih iz ASCII tabele i gore vec zadatih kordinata, program iscrtava interfejs i upisuje potrebne brojeve i operacije.

Nakon iscrtavanja grafickog interfejsa program ceka na korisnika da unese prvi operand, uz pomoc funkcije **ReadInt** smesta zeljeni operand u akomulator, i pozivom funkcije **WriteInt** ispisuje ga na displej, bio on pozitivan ili negativan. Zatim se ceka na unos operacije. Po njenom unosu funkcija **ReadChar** cita operaciju, nakon cega program prelazi u ispisivanje zadate operacije na displej i skace na odredjeni deo koda zaduzen za tu operaciju. Tu se dalje vrsi ucitavanje drugog operanda, izvrsavanje operacije i ispisivanje rezultata na displej.

Ukoliko je rezultat negativan, program prelazi na poseban deo koda koji sluzi za ispisivanje negativnih brojeva. To se radi zbog nemogucnosti ispisivanja negativnih brojeva u asemblerskom jeziku. Naime, na pocetku se "negativan" rezultat negira, dodaje *char* '-' i koristi funkcija **WriteDec**, koja ispisuje vrednos rezultata bez znaka ispred.

U nastavku je dat ceo kod GUI kalkulatora.

```
; ------------ - GUI Kalkulator---------------- -

; Projekat iz Racunarske elektronike napisan koristeci Irvine32.inc
biblioteku
; Studenti: Petar Pavlovic 188 / 2015 i Luka Adamovic 526 / 2015
; Elektrotehnicki Fakultet u Beogradu
; AddTwo.asm - Glavni asemblerski fajl
; Uputstvo za koriscenje kalkulatora:
; Prvi korak : Uneti zeljenji pozitivan ili negativan broj, zatim kliknuti
Enter
; Drugi korak : Uneti neku od ponudjenih operacija
; Treci korak : Uneti drugi operand, zatim kliknuti Enter
; Cetvrti korak : Diviti se rezultatu : 3
; Peti korak : Ukoliko zelite nazad na ponovno racunanje kliknuti Enter, pa
se onda vratiti Prvi korak,
; a ukoliko zelite da izadjete iz programa jos dva puta kliknuti Enter

INCLUDE Irvine32.inc

; ------------ - Define the grid------------------
lineTop = 0; top row number
lineLeft = 0; left row number
lineHorizontal1 = 4; horizontal row number
lineHorizontal2 = 8
lineHorizontal3 = 12
lineHorizontal4 = 16
lineBottom = 20; bottom row number
lineRight = 16; right row number
lineVertical1 = 4; vertical row number
lineVertical2 = 8
lineVertical3 = 12

; ----------------Corners position-------------- -

; ---------------- - Corners------------------------
x11 = 0
y11 = 0
x15 = 16
y15 = 0
x61 = 0
y61 = 20
x65 = 16
y65 = 20

; ------------------ - Side corners----------------------
x21 = 0
y21 = 4
x22 = 4
y22 = 4
x23 = 8
y23 = 4
x24 = 12
y24 = 4
x25 = 16
y25 = 4
```

```
x31 = 0
y31 = 8
x35 = 16
y35 = 8
x41 = 0
y41 = 12
x45 = 16
y45 = 12
x51 = 0
y51 = 16
x55 = 16
y55 = 16
x62 = 4
y62 = 20
x63 = 8
y63 = 20
x64 = 12
y64 = 20

; -------------- - Central corners------------------
x32 = 4
y32 = 8
x33 = 8
y33 = 8
x34 = 12
y34 = 8
x42 = 4
y42 = 12
x43 = 8
y43 = 12
x44 = 12
y44 = 12
x52 = 4
y52 = 16
x53 = 8
y53 = 16
x54 = 12
y54 = 16

; ------------------ - Char position------------------ -
x0 = 2
y0 = 18
xeq = 6
yeq = 18
xpow = 10
ypow = 18
xad = 14
yad = 18
x1 = 2
y1 = 14
x2 = 6
y2 = 14
x3 = 10
y3 = 14
xsub = 14
ysub = 14
x4 = 2
```

```
y4 = 10
x5 = 6
y5 = 10
x6 = 10
y6 = 10
xmul = 14
ymul = 10
x7 = 2
y7 = 6
x8 = 6
y8 = 6
x9 = 10
y9 = 6
xdiv = 14
ydiv = 6
xfirst = 2
yfirst = 2

; ------------------Define the window size------------------ -
xmin = 0; left edge
xmax = 17; right edge
ymin = 0; top
ymax = 21; bottom


.data       ;Strings for writing and variables that are used in program
num1 word ?
num2 word ?

button byte ?
pomneg byte "-", 0
pominf byte "inf :(", 0
pomnul byte "undefined :p", 0
pomerase byte "              ", 0

.code       ;Code of program starts here

main PROC

; PROGRAM STARTS HERE
; ----------------------------------------------------------------

;-----------------------Color of the background-----------------
mov eax, white + (blue * 16)
call SetTextColor
call Clrscr

; ---- - ---------------Hides the cursor-------------------------
.data
cursorInfo CONSOLE_CURSOR_INFO <>
outHandle  DWORD ?
.code
INVOKE GetStdHandle, STD_OUTPUT_HANDLE
mov  outHandle, eax
INVOKE GetConsoleCursorInfo, outHandle, ADDR cursorInfo
mov  cursorInfo.bVisible, 0
INVOKE SetConsoleCursorInfo, outHandle, ADDR cursorInfo
```

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

; ------Draw the Vertical Line 1---------------------------------------
; from(0, 0) --to (0, 20)

mov  dl, lineLeft
mov  dh, lineTop
mov  ecx, lineBottom - lineTop + 1
mov  al, 0BAh

DrawLineV1 :
call Gotoxy
call WriteChar
inc  dh
loop DrawLineV1
; ------------------------------------------------------------ -

; ------Draw the Vertical Line 2 ---------------------------------------
; from(4, 4) --to(4, 20)

mov  dl, lineVertical1
mov  dh, lineHorizontal1
mov  ecx, lineBottom - lineHorizontal1 + 1
mov  al, 0BAh

DrawLineV2 :
call Gotoxy
call WriteChar
inc  dh
loop DrawLineV2
; ------------------------------------------------------------ -

; ------Draw the Vertical Line 3 ---------------------------------------
; from(8, 4) --to(8, 20)

mov  dl, lineVertical2
mov  dh, lineHorizontal1
mov  ecx, lineBottom - lineHorizontal1 + 1
mov  al, 0BAh

DrawLineV3 :
call Gotoxy
call WriteChar
inc  dh
loop DrawLineV3
; ------------------------------------------------------------ -

; ------Draw the Vertical Line 4 ---------------------------------------
; from(12, 4) --to(12, 20)

mov  dl, lineVertical3
mov  dh, lineHorizontal1
mov  ecx, lineBottom - lineHorizontal1 + 1
mov  al, 0BAh

DrawLineV4 :
call Gotoxy
```

```
call WriteChar
inc  dh
loop DrawLineV4
; ------------------------------------------------------------ -

; ------Draw the Vertical Line 5 ---------------------------------------
; from(16, 0) --to(16, 20)

mov  dl, lineRight
mov  dh, lineTop
mov  ecx, lineBottom - lineTop + 1
mov  al, 0BAh

DrawLineV5 :
call Gotoxy
call WriteChar
inc  dh
loop DrawLineV5
; ------------------------------------------------------------ -


; ------Draw the Horizontal Line 1 ---------------------------------------
; from(0, 0) --to(16, 0)

mov  dh, lineTop
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
mov  al, 0CDh

DrawLineH1 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH1
; ------------------------------------------------------------ -

; ------Draw the Horizontal Line 2 ---------------------------------------
; from(0, 4) --to(16, 4)

mov  dh, lineHorizontal1
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
mov  al, 0CDh

DrawLineH2 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH2
; ------------------------------------------------------------ -

; ------Draw the Horizontal Line 3 ---------------------------------------
; from(0, 8) --to(16, 8)

mov  dh, lineHorizontal2
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
```

```
mov  al, 0CDh

DrawLineH3 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH3
; ------------------------------------------------------------ -

; ------Draw the Horizontal Line 4 ---------------------------------------
; from(0, 12) --to(16, 12)

mov  dh, lineHorizontal3
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
mov  al, 0CDh

DrawLineH4 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH4
; ------------------------------------------------------------ -

; ------Draw the Horizontal Line 5 ---------------------------------------
; from(0, 16) --to(16, 16)

mov  dh, lineHorizontal4
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
mov  al, 0CDh

DrawLineH5 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH5
; ------------------------------------------------------------ -

; ------Draw the Horizontal Line 6 ---------------------------------------
; from(0, 20) --to(16, 20)

mov  dh, lineBottom
mov  dl, lineLeft
mov  ecx, lineRight - lineLeft + 1
mov  al, 0CDh

DrawLineH6 :
call Gotoxy
call WriteChar
inc  dl
loop DrawLineH6
; ------------------------------------------------------------ -

; ------Draw corners-----------------------------------------
mov al, 0C9h
mov dl, x11
```

```
mov dh, y11

call Gotoxy
call WriteChar

mov al, 0BBh
mov dl, x15
mov dh, y15

call Gotoxy
call WriteChar

mov al, 0BCh
mov dl, x65
mov dh, y65

call Gotoxy
call WriteChar

mov al, 0C8h
mov dl, x61
mov dh, y61

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw right curve----------------------------------------
mov al, 0CCh
mov dl, x21
mov dh, y21

call Gotoxy
call WriteChar

mov dl, x31
mov dh, y31

call Gotoxy
call WriteChar

mov dl, x41
mov dh, y41

call Gotoxy
call WriteChar

mov dl, x51
mov dh, y51

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw left curve----------------------------------------
mov al, 0B9h
mov dl, x25
```

```
mov dh, y25

call Gotoxy
call WriteChar

mov dl, x35
mov dh, y35

call Gotoxy
call WriteChar

mov dl, x45
mov dh, y45

call Gotoxy
call WriteChar

mov dl, x55
mov dh, y55

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw down curve----------------------------------------
mov al, 0CBh
mov dl, x22
mov dh, y22

call Gotoxy
call WriteChar

mov dl, x23
mov dh, y23

call Gotoxy
call WriteChar

mov dl, x24
mov dh, y24

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw up curve----------------------------------------
mov al, 0CAh
mov dl, x62
mov dh, y62

call Gotoxy
call WriteChar

mov dl, x63
mov dh, y63

call Gotoxy
```

```
call WriteChar

mov dl, x64
mov dh, y64

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw center----------------------------------------
mov al, 0CEh
mov dl, x32
mov dh, y32

call Gotoxy
call WriteChar

mov dl, x33
mov dh, y33

call Gotoxy
call WriteChar

mov dl, x34
mov dh, y34

call Gotoxy
call WriteChar

mov dl, x42
mov dh, y42

call Gotoxy
call WriteChar

mov dl, x43
mov dh, y43

call Gotoxy
call WriteChar

mov dl, x44
mov dh, y44

call Gotoxy
call WriteChar

mov dl, x52
mov dh, y52

call Gotoxy
call WriteChar

mov dl, x53
mov dh, y53

call Gotoxy
```

```
call WriteChar

mov dl, x54
mov dh, y54

call Gotoxy
call WriteChar
; ----------------------------------------

; ------Draw chars----------------------------------------
mov al, 030h
mov dl, x0
mov dh, y0

call Gotoxy
call WriteChar

mov al, 03Dh
mov dl, xeq
mov dh, yeq

call Gotoxy
call WriteChar

mov al, 05Eh
mov dl, xpow
mov dh, ypow

call Gotoxy
call WriteChar

mov al, 02Bh
mov dl, xad
mov dh, yad

call Gotoxy
call WriteChar

mov al, 031h
mov dl, x1
mov dh, y1

call Gotoxy
call WriteChar

mov al, 032h
mov dl, x2
mov dh, y2

call Gotoxy
call WriteChar

mov al, 033h
mov dl, x3
mov dh, y3

call Gotoxy
```

```
call WriteChar

mov al, 02Dh
mov dl, xsub
mov dh, ysub

call Gotoxy
call WriteChar

mov al, 034h
mov dl, x4
mov dh, y4

call Gotoxy
call WriteChar

mov al, 035h
mov dl, x5
mov dh, y5

call Gotoxy
call WriteChar

mov al, 036h
mov dl, x6
mov dh, y6

call Gotoxy
call WriteChar

mov al, 078h
mov dl, xmul
mov dh, ymul

call Gotoxy
call WriteChar

mov al, 037h
mov dl, x7
mov dh, y7

call Gotoxy
call WriteChar

mov al, 038h
mov dl, x8
mov dh, y8

call Gotoxy
call WriteChar

mov al, 039h
mov dl, x9
mov dh, y9

call Gotoxy
call WriteChar
```

```
mov al, 0F6h
mov dl, xdiv
mov dh, ydiv

call Gotoxy
call WriteChar

mov dl, xfirst
mov dh, yfirst
call Gotoxy

jmp start
; -------------------Drawing ends here-----------------------

;--------------------Press any key to continue...------------
loopWait:
      mov eax, 10 ;delay for msg processing
      call Delay
      call ReadKey
      JZ loopWait

start :

      ;----------------------Positioning for writing----------
      mov dl, xfirst
      mov dh, yfirst

      call Gotoxy

      mov edx, offset pomerase
      call writestring

      mov dl, xfirst
      mov dh, yfirst

      call Gotoxy


; ----------Read and write first No------------------ -
      mov edx, offset num1                ; Read first no
      call readint
      mov num1, ax

      mov dl, xfirst
      mov dh, yfirst

      call Gotoxy
      call WriteInt

; ------------------------------------------------------ -

; ---------- - Read and write operation------------------
      mov edx, offset button
      call readchar
      mov button, al
```

```
        call WriteChar

        cmp button, '+'
        JE addition

        cmp button, '-'
        JE subtraction

        cmp button, '*'
        JE multiplication

        cmp button, '/'
        JE division

        cmp button, '^'
        JE power

        cmp button, 'r'
        JE start
        JNE stop

;----------------------Addision----------------------
addition :

        mov edx, offset num2; read 2nd no
        call readint
        mov num2, ax

        mov ax, num1
        add ax, num2

        bt ax, 31
        JC printNegAdd    ;If the number is negative jump to printNegAdd

        mov dl, xfirst
        mov dh, yfirst

        call Gotoxy

        mov edx, offset pomerase
        call writestring

        mov dl, xfirst
        mov dh, yfirst

        call Gotoxy

        call writeint

        JMP loopWait

        printNegAdd:

              neg ax

              mov dl, xfirst
              mov dh, yfirst
```

```
            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomneg
            call writestring
            call writedec

            JMP loopWait
;----------------------Substraction---------------------
subtraction :

      mov edx, offset num2; read 2nd no
      call readint
      mov num2, ax

      mov ax, num1

      bt ax, 31
      JC printNegSubAdd             ; If the first number is negative jump to
printNegSubAdd

      sub ax, num2

      JC prntNegSub                       ;If the result is negative jump to
printNegSub

      mov dl, xfirst
      mov dh, yfirst

      call Gotoxy

      mov edx, offset pomerase
      call writestring

      mov dl, xfirst
      mov dh, yfirst

      call Gotoxy

      call writeint

      JMP loopWait

      prntNegSub:
            mov ax, num2
            sub ax, num1

            mov dl, xfirst
            mov dh, yfirst
```

```
            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomneg
            call writestring
            call writedec

            JMP loopWait

      printNegSubAdd:

            neg num1
            mov ax, num1
            add ax, num2

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomneg
            call writestring
            call writedec

            JMP loopWait
;--------------------Multiplication--------------------

multiplication :

      mov edx, offset num2; read 2nd no
      call readint
      mov num2, ax

      mov ax, num1
      bt ax, 31
      JC prntNegMul                 ;If the first number is negative jump to
printNegMul
      mov bx, num2
      mul bx

      mov dl, xfirst
```

```
        mov dh, yfirst

        call Gotoxy

        mov edx, offset pomerase
        call writestring

        mov dl, xfirst
        mov dh, yfirst

        call Gotoxy

        call writeint

        JMP loopWait

        prntNegMul :

              neg num1
              mov ax, num1
              mov bx, num2
              mul bx

              mov dl, xfirst
              mov dh, yfirst

              call Gotoxy

              mov edx, offset pomerase
              call writestring

              mov dl, xfirst
              mov dh, yfirst

              call Gotoxy

              mov edx, offset pomneg
              call writestring
              call writedec

              JMP loopWait

;-------------------Division--------------------------
division :

        mov edx, offset num2                ;Read 2nd no
        call readint
        JZ printInfOrUnd                    ;If the second number is zero jump
to printInfOrUnd
        mov num2, ax

        mov ax, num1
        bt ax, 31
        jc prntNegDiv                              ; If the first number is
negative jump to printNegDiv
        mov bx, num2
        mov dx, 0
```

```
div bx

mov dl, xfirst
mov dh, yfirst

call Gotoxy

mov edx, offset pomerase
call writestring

mov dl, xfirst
mov dh, yfirst

call Gotoxy

call writeint

JMP loopWait

prntNegDiv :

	neg num1
	mov ax, num1
	mov bx, num2
	mov dx, 0
	div bx

	mov dl, xfirst
	mov dh, yfirst

	call Gotoxy

	mov edx, offset pomerase
	call writestring

	mov dl, xfirst
	mov dh, yfirst

	call Gotoxy

	mov edx, offset pomneg
	call writestring
	call writedec

	JMP loopWait

printInfOrUnd:

	mov ax, num1
	mov bx, 0FFFFh
	and ax, bx
	JNZ printInf

	mov dl, xfirst
	mov dh, yfirst

	call Gotoxy
```

```
            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomnul
            call writestring

            JMP loopWait

      printInf:
            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomInf
            call writestring

            JMP loopWait

;-----------------------Power----------------------
power :

      mov edx, offset num2                ; Read 2nd no
      call readint
      mov num2, ax
      mov di, num1
      mov si, num2
      mov ax, num1

      JZ printOne                               ; If the second number is
zero jump to printOne

      mov cx, num2

      mov bx, ax
      bt ax, 31

      JNC LMul

      neg num1
      mov ax, num1
      mov bx, ax
```

```
	LMul :
		dec cx
		cmp cx, 0
		JZ printPow
		mul bx
		JMP LMul

	printPow:
		bt di, 31
		jc PrintNeg                          ; If the first number is
negative jump to printNeg

	printPos:

		mov dl, xfirst
		mov dh, yfirst

		call Gotoxy

		mov edx, offset pomerase
		call writestring

		mov dl, xfirst
		mov dh, yfirst

		call Gotoxy

		call writeint

		JMP loopWait

	PrintNeg:

		bt si, 0
		JNC printPos                    ;If the second number is even jump
to printPos

		mov dl, xfirst
		mov dh, yfirst

		call Gotoxy

		mov edx, offset pomerase
		call writestring

		mov dl, xfirst
		mov dh, yfirst

		call Gotoxy

		mov edx, offset pomneg
		call writestring
		call writedec

		JMP loopWait

	printOne :
```

```
            mov bx, 0FFFFh
            and ax, bx
            JZ printError                       ;If the bouth numbers are
zero jump to printError
            mov ax, 1

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            call writeint

            JMP loopWait

      printError:

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomerase
            call writestring

            mov dl, xfirst
            mov dh, yfirst

            call Gotoxy

            mov edx, offset pomnul
            call writestring

            JMP loopWait

;----------------------Program ENDS here-----------------------
stop :
      exit
main endp
end main
```