-------binary representation and stores the individual binary digits in reverse order in an array named array . Once the conversion is complete, use a loop to copy the converted digits from the stack to the array-----

```
org 0x100]

jmp start


array : 0,0,0,0,0
num1 : 0xABCD
conversion:
push bp
push ax
push bx
push cx
push dx



mov ax , [bp+4]
mov bx , 10 ; the binary conversion
mov cx , 0

nextdigit:
mov dx , 0
div bx
add dl , 0x30
push dx
inc cx
cmp ax , 0
jnz nextdigit

mov di , 0

start:
mov cx , 5
mov bp , 0
mov ax , num1
mov bx , array

looper :
pop dx
mov dx , [array+bp]
add bp , 2
dec cx
cmp cx , 0
jne looper

mov ax , 0x4c00
int 0x21c
pop dx
mov array , dx
add 2
```

Clear the screen before drawing each rectangle.
Implement a subroutine called ’makingrectangle’ that takes the four coordinates as arguments and draws a rectangle using the ’*’ character and a specified color. The rectangle should be filled with the specified c olor.
The ’makingrectangle’ subroutine should handle the color and coordinate calculations correctly to draw th e rectangle.
After drawing each rectangle, the program should wait for user input before proceeding to draw the next r ectangle

```
[org 0x100]

jmp start

top : dw 20
bottom : dw 30
left : dw 40
right : dw 70

top2 : dw 22
bottom2 : dw 28
left2 : dw 42
right2 : dw 68

top3 : dw 24
bottom3 : dw 26
left3 : dw 44
right3 : dw 66

start :
call clrscr
push word [top]
push word [bottom]
push word [left]
push word [right]

call makingrectangle

end :
mov ax , 0x4c00
int 0x21

clrscr :
mov ax , 0xb800
mov es , ax


xor di , di
mov ax , 0x0720
moc cx , 2000
```

```
cld
rep stosw

ret


makingrectangle:

push bp
mov bp ,  sp
push ax

mov al , 80
mul byte [bp+10]
add ax , [bp+6]
shl ax , 1
mov di , ax

push di

mov ah , 0x04 ; red color

mov cx , [bp+4]
sub cx , [bp+6]

push cx


mov al, '*'

loop1:
rep stosw
pop bx
pop di
push bx
dec bx
shl bx , 1
add di , 160

mov cx , [bp+8]
sub cx , [bp+10]

sub cx , 2

mov al , '*'

loop2:
mov si ,di
mov word [es:si] , ax
add si , bx
mov word [es:si] , ax
sub si ,bx
add di , 160
loop loop2
```

```
pop cx
mov al , '*'

loop3: rep stosw
return:
pop ax
pop bp
ret 8

call clrscr2
push word [top2]
push word [bottom2]
push word [left2]
push word [right2]

call makingrectangle2

end :
mov ax , 0x4c00
int 0x21

clrscr2 :
mov ax , 0xb800
mov es , ax


xor di , di
mov ax , 0x0720
moc cx , 2000

cld
rep stosw

ret


makingrectangle2:

push bp
mov bp ,  sp
push ax

mov al , 80
mul byte [bp+10]
add ax , [bp+6]
shl ax , 1
mov di , ax

push di

mov ah , 0x03 ;  color

mov cx , [bp+4]
sub cx , [bp+6]
```

```
push cx


mov al, '*'

loop12:
rep stosw
pop bx
pop di
push bx
dec bx
shl bx , 1
add di , 160

mov cx , [bp+8]
sub cx , [bp+10]

sub cx , 2

mov al , '*'

loop22:
mov si ,di
mov word [es:si] , ax
add si , bx
mov word [es:si] , ax
sub si ,bx
add di , 160
loop2 loop22

pop cx
mov al , '*'

loop32: rep stosw
return:
pop ax
pop bp
ret 8

call clrscr3
push word [top3]
push word [bottom3]
push word [left3]
push word [right3]

call makingrectangle3

end :
mov ax , 0x4c00
int 0x21

clrscr3 :
mov ax , 0xb800
mov es , ax
```

```
xor di , di
mov ax , 0x0720
moc cx , 2000

cld
rep stosw

ret


makingrectangle3:

push bp
mov bp ,  sp
push ax

mov al , 80
mul byte [bp+10]
add ax , [bp+6]
shl ax , 1
mov di , ax

push di

mov ah , 0x07 ;  color

mov cx , [bp+4]
sub cx , [bp+6]

push cx


mov al, '*'

loop13:
rep stosw
pop bx
pop di
push bx
dec bx
shl bx , 1
add di , 160

mov cx , [bp+8]
sub cx , [bp+10]

sub cx , 2

mov al , '*'

loop23:
mov si ,di
mov word [es:si] , ax
```

```
add si , bx
mov word [es:si] , ax
sub si ,bx
add di , 160
loop3 loop23

pop cx
mov al , '*'

loop33: rep stosw
return:
pop ax
pop bp
ret 8
```

"Write an x86 assembly program that prints a decimal number at the top left corner of the screen. The pro gram should implement the following functionalities:

Implement a subroutine called 'printnum' that takes a decimal number as an argument and prints it at the t op left corner of the screen using video memory.
The 'printnum' subroutine should handle the binary to decimal conversion and printing logic correctly.
After printing the number, the program should clear the screen using a subroutine called 'clrscr'.
The decimal number to be printed is 4529.

```
[org 0x100]

jmp start

; printing on top left corner
printnum:
push bp
move bp , sp
push es
push ax
push bx
push cx
push dx
push di

mov ax , 0xb800
mov es , ax
mov ax , [bp+4]
mov bx , 2 ; the binary conversion
mov cx , 0

nextdigit:
mov dx , 0
div bx
add dl , 0x30
push dx
inc cx
cmp ax , 0
jnz nextdigit
```

```
mov di , 0

nextpos:
pop dx
mov dh , 0x07
mov [es:di] , dx
add di , 2
loop nextpos
pop di
pop dx
pop cx
pop bx
pop ax
pop es
pop bp
ret 2

start:
call clrscr ; calling clr screen func
mov ax , 4529
push ax
call printnum

mov ax , 0x4c00
int 0x21
```