# TASK 1:

org 0x0100

mov cx, 9 ; Height

mov bx, 9 ; Width

mov sp, 0x1000 ; Initialize stack pointer

push bx ; Width (Push parameters onto the stack)

push cx ; Height

call clear\_screen

call draw\_rectangle

add sp, 4 ; Clean up the stack

mov ah, 4Ch ; Terminate program

int 21h

clear\_screen:

mov ax, 0xb800 ; load video base in ax

mov es, ax ; point es to video base

mov di, 0 ; point di to top left column

nextchar: mov word [es:di], 0x0720 ; clear next char on screen

add di, 2 ; move to next screen location

cmp di, 4000 ; has the whole screen cleared

jne nextchar ; if no clear next position

ret

draw\_rectangle:

push bp

mov bp, sp

mov cx, [bp + 4] ; Height (Access parameters using BP)

mov bx, [bp + 6] ; Width

L1:

push cx ; Preserve the outer loop counter

mov cx, bx ; Set the inner loop counter

L2:

mov ax, cx ; y position (Calculate the location using the formula)

imul ax, 50

add ax, bx ; x position

imul ax, 3

mov ah, 2 ; Function code for printing character

mov dl, ' ' ; Character to be displayed

int 21h ; Print the character

mov ah, 2 ; Function code for printing character

mov dl, '\*' ; Character to be displayed

int 21h ; Print the character

loop L2 ; Continue printing if there are more characters in this row

mov ah, 9 ; Function code for printing string

mov dx, data ; Address of the newline and terminator string

int 21h ; Print newline and terminator

pop cx ; Restore the outer loop counter

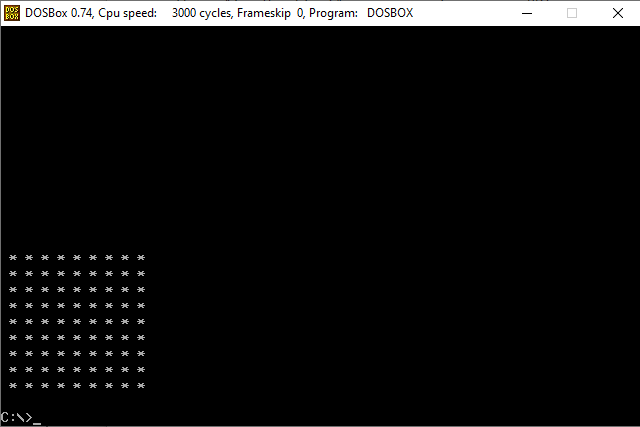
loop L1 ; Continue printing if there are more rows

pop bp

ret

section .data

data db 20, '$' ; Newline and string terminator for printing

OUTPUT:  


# TASK 2:

org 0x0100

mov cx, 9 ; Height

mov bx, 0 ; Width

mov sp, 0x1000 ; Initialize the stack pointer

push bx ; Width (Push parameters onto the stack)

push cx ; Height

call clear\_screen

call draw\_triangle

add sp, 4 ; Clean up the stack

mov ah, 4Ch ; Terminate program

int 21h

clear\_screen:

mov ax, 0xb800 ; load the video base in ax

mov es, ax ; point es to the video base

mov di, 0 ; point di to the top left column

nextchar: mov word [es:di], 0x0720 ; clear the next character on the screen

add di, 2 ; move to the next screen location

cmp di, 4000 ; has the whole screen cleared

jne nextchar ; if not, clear the next position

ret

draw\_triangle:

push bp

mov bp, sp

mov cx, [bp + 4] ; Height (Access parameters using BP)

mov bx, [bp + 6] ; Width

L1:

push cx ; Preserve the outer loop counter

inc bx ; increment to get triangle

mov cx, bx ; Set the inner loop counter

L2:

mov ax, cx ; y position (Calculate the location using the formula)

imul ax, 50

add ax, bx ; x position

imul ax, 3

mov ah, 2 ; Function code for printing character

mov dl, '\*' ; Character to be displayed

int 21h ; Print the character

loop L2 ; Continue printing if there are more characters in this row

mov ah, 9 ; Function code for printing string

mov dx, data ; Address of the newline and terminator string

int 21h ; Print newline and terminator

pop cx ; Restore the outer loop counter

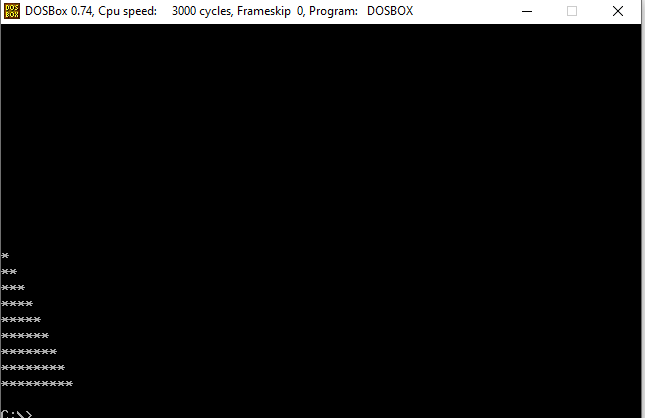
loop L1 ; Continue printing if there are more rows

pop bp

ret

section .data

data db 10, '$' ; Newline and string terminator for printing

OUTPUT:  


TASK 3:  
org 0x0100

mov cx, 5 ; Height

mov bx, 3 ; Width

mov sp, 0x1000 ; Initialize stack pointer

push bx ; Width (Push parameters onto the stack)

push cx ; Height

call clear\_screen

call draw\_rectangle

add sp, 4 ; Clean up the stack

mov ah, 4Ch ; Terminate program

int 21h

clear\_screen:

mov ax, 0xb800 ; load video base in ax

mov es, ax ; point es to video base

mov di, 0 ; point di to top left column

nextchar: mov word [es:di], 0x0720 ; clear next char on screen

add di, 2 ; move to next screen location

cmp di, 4000 ; has the whole screen cleared

jne nextchar ; if no clear next position

ret

draw\_rectangle:

push bp

mov bp, sp

mov cx, [bp + 4] ; Height (Access parameters using BP)

mov bx, [bp + 6] ; Width

L1:

push cx ; Preserve the outer loop counter

mov cx, bx ; Set the inner loop counter

mov ah, 2 ; Function code for printing character

mov dl, ' ' ; Character to be displayed

int 21h ; Print the character

L2:

mov ax, cx ; y position (Calculate the location using the formula)

imul ax, 50

add ax, bx ; x position

imul ax, 3

mov ah, 2 ; Function code for printing character

mov dl, '\*' ; Character to be displayed

int 21h ; Print the character

loop L2 ; Continue printing if there are more characters in this row

mov ah, 9 ; Function code for printing string

mov dx, data ; Address of the newline and terminator string

int 21h ; Print newline and terminator

pop cx ; Restore the outer loop counter

loop L1 ; Continue printing if there are more rows

pop bp

ret

section .data

data db 10, '$' ; Newline and string terminator for printing