;Y=100-60 sin((pi/180)\*x)

;Bit 7 6 5 4 3 2 1 0

;Data R R R G G G B B

.model small

.stack 100

.data

msg db 10,13,'this is cosine wave$'

one\_eighty dw 180

scale dw 30

hundred dw 100

rint dw 0

x dw 0

.code

.386

main:

mov ax,@data ; Initialize DS (needed for .exe-program)

mov ds, ax

mov ax, 0A000h ; Segment to video memory

mov es, ax

mov ax, 13h

int 10h ; switch to 320x200 mode

mov cx, 0

l1:

push cx ; store CX

call get\_cosine

mov bx, cx

call vector\_to\_memory

mov di, ax

mov al, 03h ; color

mov [es:di], al ; put pixel

pop cx ; restore CX

inc cx ; CX = CX + 1

cmp cx, 320 ; right boarder reached?

jne l1 ; no, next degree

mov ah,09h ; display message

lea dx,msg

int 21h

xor ah, ah

int 16h ; keyboard (wait for key)

mov ax, 3

int 10h ; go to text mode

mov ax, 4C00h

int 21h ; return to DOS, exit code 0

get\_cosine: ; Args: CX = angle (degree!)

mov word[x],cx

fldpi ; ST(0)=Pi, ST(1)=CX

fimul word[x] ; ST(0)=Pi\*CX

fidiv word[one\_eighty] ; ST(0)=(Pi\*CX)/180(degree to rad)

fsin ; ST(0)=sine (rad)

fimul word[scale] ; ST(0)=sine\*scale

fild hundred

fsub st,st(1) ;ST(0)=100-60\*SIN((Pi\*CX)/180)

fist word [rint] ; store integer with rounding

mov ax, word[rint] ; AX = Y

ret ; Return: AX = Y (signed!)

vector\_to\_memory: ; Calculate offset=Y\*320+X

;Args: BX = X, AX = Y

push dx ; mul changes dx too

mov cx, 320 ; video mode width

mul cx ; DX:AX = AX \* CX

add ax, bx ; left indentation

pop dx

ret ; Return: AX = offset in memory

end main

;\*\*\*\*\*\*\*\*\*\*\*\*\*Output\*\*\*\*\*\*\*\*\*\*\*

