MP ASSIGNMENT 1

Group Members:

1. **Ali Hamza Malik CMS: 291480**
2. **Ali Aqdas CMS: 285050**

SOURCE CODE:

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;Assembly code for Clock Graphics                        ;
;Written by ALI HAMZA MALIK and ALI AQDAS                 ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
.model small
.stack 100h
PlotLine macro xA,yA,xB,yB,colorofline
   LOCAL dxGTdy,dxNGTdy,xaGTxb
   LOCAL xaNGTxb,yaGTyb,yaNGTyb,xaDONExb,yaDONEyb,dxDONEdy
   ;temporary storing
   mov al,colorofline
   mov linecolor,al
   mov ax,xa
   mov xatmp,ax
   mov ax,xb
   mov xbtmp,ax
   mov ax,ya
   mov yatmp,ax
   mov ax,yb
   mov ybtmp,ax

   ;computing delX and delY
   mov ax,yB
   sub ax,yA
   mov bx,xB
   sub bx,xA
   ;taking absolute values and storing
   call absval
   mov tmpdelY,ax
   mov ax,bx
   call absval
   mov tmpdelX,ax
   mov ax,tmpdelx
   mov bx,tmpdely
```

```
cmp ax,bx
jg dxGTdy
jmp dxNGTdy
;abs[dy]<abs[dx].......
dxGTdy:
mov ax,xA
mov bx,xB
cmp ax,bx
jg xaGTxb
jmp xaNGTxb


;XA>XB................
xaGTxb:
   ;mov ax,xb
   ;mov bx,yb
   ;mov cx,xa
   ;mov dx,ya
   ;PlotLineLow x1,y1,x02,y02
   PlotLineLow xbtmp,ybtmp,xatmp,yatmp
;exit to inner endif
jmp xaDONExb

;XA NOT > XB
xaNGTxb:
;PlotLineLow x02,y02,x1,y1
PlotLineLow xatmp,yatmp,xbtmp,ybtmp

xaDONExb: ;endif

jmp dxDONEdy ;exit to end
;........................

;abs[dy] NOT <abs[dx]..........
dxNGTdy:

mov cx,yB;
mov dx,yA;


cmp dx,cx
jg yaGTyb
jmp yaNGTyb

yaGTyb:
;PlotLineHigh x1,y1,x02,y02
PlotLineHigh xbtmp,ybtmp,xatmp,yatmp
;exit of inner endif
```

```
    jmp yaDONEyb

    yaNGTyb:
    ;PlotLineHigh x02,y02,x1,y1
    PlotLineHigh xatmp,yatmp,xbtmp,ybtmp

    yaDONEyb:

    dxDONEdy:

endm

drawcircle macro xc,yc,r,color
  local circleloop,cond,exitloop

  mov ax,xc
  mov x0,ax

  mov ax,yc
  mov y0,ax

  mov ax,r
  mov circleradius,ax

    ;         xcircle = 0;
    ;         ycircle = circleradius;
    ;         circleerror = -circleradius;
    mov ax,circleradius
    mov ycircle,ax
    neg ax
    mov circleerror,ax

    mov al,color
    mov circolor,al
circleloop:
call circlepoints
;inc circolor
call inccirclepoints
;while (xcircle <= ycircle)
    mov ax,xcircle
    mov bx,ycircle
    cmp ax,bx
    jg exitloop
    jmp circleloop   ;jump to circleloop

    exitloop:
endm
```

```
PlotLineLow macro xintL,yintL,xfinL,yfinL
   LOCAL condL,overcondL,errGzeroL,elseL,outL,yloopL
;PlotLineHigh proc
;requires values in y1 y02 x1 x02

;mov ax,y1  ;moving y1  to ax
;sub ax,y02  ;subtracting y02 from y1
;mov bx,x1  ;moving x1 to ax
;sub bx,x02  ;subtracting x02 from x1
;.......................................
mov ax,yfinL  ;moving y1  to ax
sub ax,yintL  ;subtracting y02 from y1
mov bx,xfinL  ;moving x1 to ax
sub bx,xintL  ;subtracting x02 from x1
;.......................................
mov delx,bx ; moving dx to delx
mov dely,ax ; moving dy to dely

mov dx,1
mov yi,dx

;..................
;if (dy)<0

cmp ax,0
jl condL
jmp overcondL
condL:
mov dx,-1
mov yi,dx
neg ax
mov dely,ax
overcondL:

;end if
;..................
;D=2dy-dx
mov ax,dely
add ax,dely
sub ax,delx
mov error,ax
;..................

;mov si,x02
mov si,yintL ;............................

mov y02temp,si ;duplication y02
```

```
;mov dx,y02
mov dx,xintL ;.............................
mov x02temp,dx ;duplication x02
mov dx,0
mov cx,delx
;for x from x02 to x1

yloopL:
mov drawloop,cx
mov dx,y02temp
mov cx,x02temp
;call pixel ;plot pixel
mov al,linecolor
plotpixel cx,dx,al
inc cx
mov x02temp,cx

;............
;if D>0
cmp error,0
jg errGzeroL
jmp elseL
;if error>0
;-----------------
errGzeroL:
;y=y+yi
add dx,yi
mov y02temp,dx
;............
;D=D+2(dy-dx)
mov bx,dely
sub bx,delx
mov dymindx,bx
add bx,dymindx
add bx,error
mov error,bx
jmp outL
;-----------------
elseL:
mov bx,dely
add bx,dely
add bx,error
mov error,bx
jmp outL

outL:

mov cx,drawloop
```

```
loop yloopL

;plotlinehigh endp
endm

PlotLineHigh macro xint,yint,xfin,yfin
   LOCAL condH,overcondH,errGzeroH,elseH,outH,yloopH
;PlotLineHigh proc
;requires values in y1 y02 x1 x02

;mov ax,y1  ;moving y1  to ax
;sub ax,y02  ;subtracting y02 from y1
;mov bx,x1  ;moving x1 to ax
;sub bx,x02  ;subtracting x02 from x1
;.......................................
mov ax,yfin  ;moving y1  to ax
sub ax,yint  ;subtracting y02 from y1
mov bx,xfin  ;moving x1 to ax
sub bx,xint  ;subtracting x02 from x1
;.......................................
mov delx,bx ; moving dx to delx
mov dely,ax ; moving dy to dely

mov dx,1
mov xi,dx

;..................
;if (dx)<0

cmp bx,0
jl condH
jmp overcondH
condH:
mov dx,-1
mov xi,dx
neg bx
mov delx,bx
overcondH:

;end if
;..................
;D=2dx-dy
mov ax,delx
add ax,delx
sub ax,dely
mov error,ax
;..................
```

```
;mov si,x02
mov si,xint ;............................

mov x02temp,si ;duplication x02
;mov dx,y02
mov dx,yint ;.............................
mov y02temp,dx ;duplication y02
mov cx,dely
;for y from y02 to y1

yloopH:
mov drawloop,cx
mov dx,y02temp
mov cx,x02temp
;call pixel ;plot pixel
mov al,linecolor
plotpixel cx,dx,al
inc dx
mov y02temp,dx

;............
;if D>0
cmp error,0
jg errGzeroH
jmp elseH
;if error>0
;-----------------
errGzeroH:
;x=x+xi
add cx,xi
mov x02temp,cx
;............
;D=D+2(dx-dy)
mov bx,delx
sub bx,dely
mov dxmindy,bx
add bx,dxmindy
add bx,error
mov error,bx
jmp outH
;-----------------
elseH:
mov bx,delx
add bx,delx
add bx,error
mov error,bx
jmp outH
```

```
outH:

mov cx,drawloop
loop yloopH

;plotlinehigh endp
endm


plotpixel macro xi, yi, color
;AH=0Ch AL = Color, CX = x, DX = y
   mov al,color
   mov cx,xi
   mov dx,yi

   mov ah, 0ch
   int 10h

endm

startvideomode macro mode,color
   mov ax, 0a000h
   mov es, ax

   mov ah, 0
   mov al, mode
   int 10h



;Set background/border color
;AH=0Bh, BH = 00h   BL = Background/Border color (border only in text modes)
   mov ah,0Bh   ;set config
   mov bh,00h
   mov bl,color   ;choose color as background color
   int 10h
endm

printchar macro x,y,char,color

mov  dl, x   ;Column
mov  dh, y  ;Row
;mov  bh, 0    ;Display page
mov  ah, 02h  ;SetCursorPosition
int  10h

mov  al, char
mov  bl, color  ;Color is red
;mov  bh, 0    ;Display page
```

```
mov  ah, 0Eh  ;Teletype
int  10h

endm


sinr macro radius
 call sin
 mov cx,radius
 mul cx
 mov cx,10004
 div cx
endm
cosr macro radius
 call cos
 mov cx,radius
 mul cx
 mov cx,10004
 div cx
endm


quadrantloop macro theta,rad,xcen,ycen
local QD2,QD3,QD4,quadext

;QD1:
mov ax,theta
cmp ax,15
JA  QD2
  mov cx,6
  mul cx
  sinr rad
  mov cx,ycen
  sub cx,ax
  mov y,cx
  mov ax,theta
  mov cx,6
  mul cx
  cosr rad
  add ax,xcen
  mov x,ax
jmp quadext
QD2:
mov ax,theta
cmp ax,30
JA  QD3
   mov cx,6
  mul cx
  sinr rad


 mov cx,ycen
  sub cx,ax
  mov y,cx

  mov ax,theta

    mov cx,6
  mul cx
  cosr rad

   mov cx,xcen
  sub cx,ax

  mov x,cx
jmp quadext
QD3:
mov ax,theta
cmp ax,45
JA  QD4
  mov cx,6
  mul cx
  sinr rad
  add ax,ycen
  mov y,ax
  mov ax,theta
  mov cx,6
  mul cx
  cosr rad
  mov cx,xcen
  sub cx,ax
  mov x,cx
jmp quadext
QD4:
  mov ax,theta
  mov cx,6
  mul cx
  sinr rad
  add ax,ycen
  mov y,ax
  mov ax,theta
  mov cx,6
  mul cx
  cosr rad
  add ax,xcen
  mov x,ax
```

```
quadext:
endm


drawincrementedmarkings macro increment,markingcolor,startradius,endradius
local outerloop,LP,endloop1,cd1,endloop2,cd2
;quadrantloop macrocurrentsecond,rad,xcen,ycen,x,y
mov cx,startradius  ;;71
mov varradius,cx

outerloop:

mov cx,0
mov currentsecond,cx

LP:

quadrantloop currentsecond,varradius,159,103
plotpixel x,y,markingcolor
mov cx,currentsecond
add cx,increment
mov currentsecond,cx


CMP cx, 60
JLE cd1    ; If it is less than or equal to 60, then jump to LP

jmp endloop1
cd1:
jmp LP
endloop1:




inc varradius
mov cx,varradius
CMP cx,endradius  ; 74
JLE cd2    ; If it is less than or equal to endradius, then jump to outerloop

jmp endloop2
cd2:
jmp outerloop
endloop2:


endm
```

```
.data
;#region
;;line var
xAtmp dw ?
xBtmp dw ?
yAtmp dw ?
yBtmp dw ?
hrs db ?
mins db ?
secs db ?
hrsdig db 2 dup(?)
minsdig db 2 dup(?)
secsdig db 2 dup(?)
linecolor db 0
x02 dw 10
x02temp dw ?
x1 dw 20
y02 dw 40
y02temp dw ?
y1 dw 20
xi dw ?
yi dw ?
delx dw ?
dely dw ?
dxmindy dw ?
dymindx dw ?
error dw ?
drawLoop dw ?

tmpdelX dw ?
tmpdelY dw ?
;;;;;;;;;;circle var
count dw 0d
circleradius dw 0d
xcircle dw 0d
ycircle dw 0d
x0 dw 0d
y0 dw 0d
varradius dw 0
temp1 dw 0d
temp2 dw 0d
circleerror dw 0
bgcolor dw 0h
circolor db 0h
;;;;;math
xsin dw 0
xsin2 dw 0
signvar db 0
```

```
 clockxcenter dw 159
 clockycenter dw 103
 clockradius  dw 95

 secondhandradius  dw 65
 minutehandradius  dw 65
 hourhandradius  dw 40

 currentsecond dw 0
 currentminute dw 0
 currenthours dw 0

 x dw 0   ;;;variable x used to traverse circle
 y dw 0   ;;;variable y used to traverse circle


;#endregion


.code
main proc

mov ax,@data ; initialize DS
mov ds,ax ; new ;;;;irvine16  wont work in 16bit without these lines
mov ax,stack  ;;just incase it works
mov ss,ax
startvideomode 13h,00h
call clockgraphics
mov ax, 4C00h               ; Exit(0)
int 21h
main endp


clockgraphics:
;#region
drawincrementedmarkings 1d,15d,74d,74d
drawincrementedmarkings 5d,4d,66d,74d
call drawclockbody

mainsecloop:

call gettime

;movzx ax,secs
mov ax,0
mov al,secs
```

```
call rescale60
mov bx,60 ;;;;making it rotate clockwise
sub bx,ax
mov currentsecond,bx


;movzx ax,mins
mov ax,0
mov al,mins
call rescale60
;mov  currentminute,ax
mov bx,60
sub bx,ax
mov currentminute,bx


;movzx ax,hrs
mov ax,0
mov al,hrs
call rescale24_12
mov cx,5
mul cx
call rescale60
mov bx,60
sub bx,ax
mov currenthours,bx


quadrantloop currentminute,minutehandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,14

quadrantloop currenthours,hourhandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,4

quadrantloop currentsecond,secondhandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,10

call customdelay

quadrantloop currentminute,minutehandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,0

quadrantloop currenthours,hourhandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,0

quadrantloop currentsecond,secondhandradius,clockxcenter,clockycenter
PlotLine clockxcenter,clockycenter,x,y,0
```

```
jmp mainsecloop

ret
;#endregion

drawclockbody:
;#region
call nulreg
drawcircle clockxcenter,clockycenter,95d,0Ch
plotpixel clockxcenter,clockycenter,2
;;159,103



printchar 19,2,'1',0Ch
printchar 20,2,'2',0Ch

printchar 20,23,'6',0Ch

printchar 30,12,'3',0Ch
printchar 9,12,'9',0Ch

printchar 25,3,'1',0Ch
printchar 29,7,'2',0Ch

printchar 29,17,'4',0Ch
printchar 26,21,'5',0Ch

printchar 13,21,'7',0Ch
printchar 10,17,'8',0Ch


printchar 14,3,'1',0Ch
printchar 15,3,'1',0Ch

printchar 10,7,'1',0Ch
printchar 11,7,'0',0Ch





call nulreg
mov dx, clockradius
sub dx,20
drawcircle clockxcenter,clockycenter,dx,1

ret
```

```
;#endregion


absval:
;#region
;................................
;takes absolute values of ax
;returns absval in ax
;................................
cmp ax,0
jge staysame
neg ax
staysame:
ret
;#endregion

gettime:
;#region
   ; Hours is in CH
   ; Minutes is in CL
   ; Seconds is in DH
   mov ah,2ch
   int 21h
   mov hrs,ch
   mov mins,cl
   mov secs,dh
   ;AAM to adjust two digit hours
   mov ah,0
   mov al,ch
   aam
   lea di,hrsdig
   mov [di],ah
   mov [di+1],al

   ;AAM to adjust two digit mins
   mov ah,0
   mov al,cl
   aam
   lea di,minsdig
   mov [di],ah
   mov [di+1],al

   ;AAM to adjust two digit secs
   mov ah,0
   mov al,dh
   aam
   lea di,secsdig
   mov [di],ah
```

```
    mov [di+1],al
ret
;#endregion


disptime:
;#region
    call gettime

    mov ah,2
    lea si,hrs

    mov dl,[si]
    add dl,30h
    int 21h
    mov dl,[si+1]
    add dl,30h
    int 21h

    mov dl,':'
    int 21h

    lea si,mins

    mov dl,[si]
    add dl,30h
    int 21h
    mov dl,[si+1]
    add dl,30h
    int 21h

    mov dl,':'
    int 21h

    lea si,secs

    mov dl,[si]
    add dl,30h
    int 21h
    mov dl,[si+1]
    add dl,30h
    int 21h

    ret
  ;#endregion

circlepoints:
;#region
```

```
mov ax,x0
add ax,xcircle
mov temp1,ax
mov ax,y0
add ax,ycircle
mov temp2,ax
        plotpixel temp1, temp2, circolor


mov ax,x0
add ax,ycircle
mov temp1,ax

mov ax,y0
add ax,xcircle
mov temp2,ax
        plotpixel temp1, temp2, circolor


mov ax,x0
sub ax,ycircle
mov temp1,ax

mov ax,y0
add ax,xcircle
mov temp2,ax
        plotpixel temp1, temp2, circolor




mov ax,x0
sub ax,xcircle
mov temp1,ax
mov ax,y0
add ax,ycircle
mov temp2,ax
        plotpixel temp1, temp2, circolor


mov ax,x0
sub ax,xcircle
mov temp1,ax
mov ax,y0
add ax,ycircle
mov temp2,ax
        plotpixel temp1, temp2, circolor
```

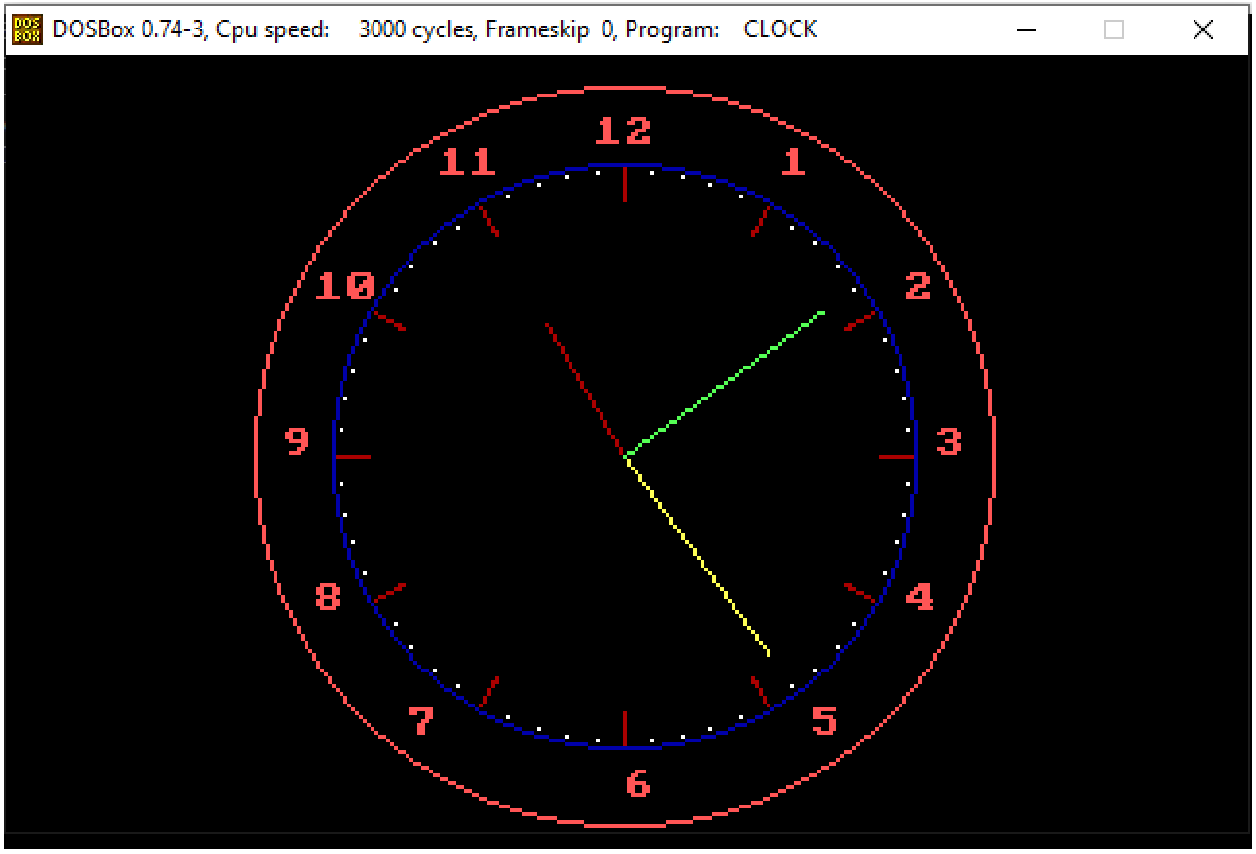
```
mov ax,x0
sub ax,xcircle
mov temp1,ax
mov ax,y0
sub ax,ycircle
mov temp2,ax
        plotpixel temp1, temp2, circolor



mov ax,x0
sub ax,ycircle
mov temp1,ax
mov ax,y0
sub ax,xcircle
mov temp2,ax
        plotpixel temp1, temp2, circolor




mov ax,x0
add ax,ycircle
mov temp1,ax
mov ax,y0
sub ax,xcircle
mov temp2,ax
        plotpixel temp1, temp2,circolor



mov ax,x0
add ax,xcircle
mov temp1,ax
mov ax,y0
sub ax,ycircle
mov temp2,ax
        plotpixel temp1, temp2, circolor

ret
;#endregion

inccirclepoints:
 ;#region
 ; circleerror+= 2*x=x+x
 mov ax,circleerror
 add ax,xcircle
 add ax,xcircle
 mov circleerror,ax

 ;           if (circleerror >= 0)
    ;           {
    ;
    ;              y --;
    ;              circleerror-=2*y;
    ;
    ;           }
    ;
    ;           x++;
    mov ax,circleerror
    cmp ax,0
    jl cond

    dec ycircle
    mov ax,circleerror
    sub ax,ycircle
    sub ax,ycircle
    mov circleerror,ax
    cond:
    inc xcircle
    ret
    ;#endregion

textmode:
;#region
 mov ah,00 ; set display mode function.
  mov al,03 ; normal text mode 3
  int 10h   ; set it!
ret
;#endregion

customdelay:
;#region
           mov     cx, 3
    delRep: push    cx
           mov     cx, 0D090H
    delDec: dec     cx
           jnz     delDec
           pop     cx
           dec     cx
           jnz     delRep
ret
;#endregion
nulreg:
;#region
```

```
mov ax,0
mov bx,0
mov cx,0
mov dx,0

mov circleradius,ax
mov xcircle,ax
mov ycircle,ax
mov x0,ax
mov y0,ax
mov temp1,ax
mov temp2,ax
mov circleerror,ax
mov bgcolor,ax
mov circolor,al
ret
;#endregion

;cos in ax out ax range=[0,1]
cos:
;#region
add ax, 90
call sin
ret
;#endregion
;sin in ax out ax range=[0,1]
sin:
;#region
push      cx
push      dx
push      bx
sin360:
cmp       ax, 90
ja        dy90
sto0_90:
mov       si, 0
jmp       pp1
dy90:
cmp       ax, 180
jbe       z91to180
jmp       dy180
z91to180:
mov       cx, 180
sub       cx, ax
mov       ax, cx
mov       si, 0
jmp       pp1
z181to270:
```

```
sub      ax, 180
mov      si, 1
jmp      pp1
z271to360:
cmp      ax, 359
ja       zdy359
mov      cx, 360
sub      cx, ax
mov      ax, cx
mov      si, 1
jmp      pp1
zdy359:
sub      ax, 360
jmp      sin360

dy180:
cmp      ax, 270
jbe      z181to270
jmp      z271to360

pp1:
mov      cx, 175
xor      dx, dx
mul      cx
mov      xsin, ax
xor      dx, dx
mov      cx, ax
mul      cx
mov      cx, 10000
div      cx
mov      xsin2, ax
xor      dx, dx
mov      cx, 120
div      cx
mov      bx, 1677;1667
cmp      ax, bx
jae      goab
xor      signvar, 1
xchg     ax, bx
goab:
sub      ax, bx
mov      cx,xsin2
xor      dx, dx
mul      cx
mov      cx, 10000
div      cx             ;xx(xx/120-10000/6)
mov      cx, 10000
mov      dl, 0
```

```
cmp      dl, signvar
je       jia
sub      cx, ax
mov      ax, cx
jmp      kk1
jia:
add      ax, cx
kk1:
mov      cx,xsin
xor      dx, dx
mul      cx
mov      cx, 10000
div      cx
pop      bx
pop      dx
pop      cx
mov      signvar, 0
ret
;#endregion

rescale60: ;;;;In ax out ax
;#region
cmp ax, 15  ; Compares whether the counter has reached 10
jle range45_60    ; If it is less than or equal to 10, then jump to LP1

sub ax,15
jmp exitrescale60

range45_60:
add ax,45
exitrescale60:
ret
;#endregion

rescale24_12:;;;;;In ax out ax
;#region
cmp ax, 12  ; Compares whether the counter has reached 10
jle exitrescale24_12    ; If it is less than or equal to 10, then jump to LP1
sub ax,12
exitrescale24_12:
ret
;#endregion

end main
```

OUTPUTSCREENSHOT:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: CLOCK
12
1
2
3
4
5
6
7
8
9
10
11