Libraries & Interfaces – The GRAPHICS Library

• **Principle:**

→ Prototype defs. of fcts. at beginning of a program

→ Actual C code of fcts. at the end

• **Program:**

```
  ⋮
  I  ALL PROTOTYPE DEFS.
  ⋮
  main ()
  ⋮
  I  C CODE OF FUNCTIONS
```

GENERAL

▪ CLIENT/USER	▪ IF	▪ LIBRARY
→ User lib. functions "DrawLine"	→ Specifies prototypes of all fcts.	→ C code of all fcts. of a library

"**IF** = Glue between client and library"

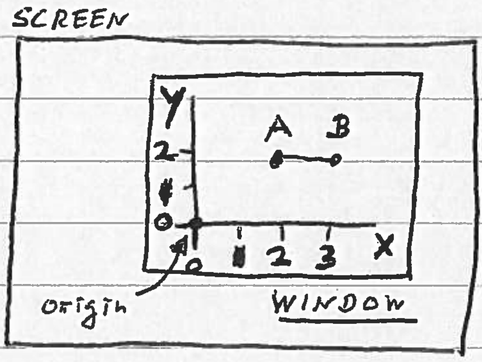
• **Ex:** <math.h> , "graphics.h", <stdio.h> , "genlib.h"

→ **Client must know**
- → NAMES of functions
- → NO. & TYPES of ARGUMENTS
- → TYPE of RESULT

graphics.h : PROTOTYPE DEFS. of graphics fcts.
graphics.c : C CODE of all graphics fcts.

• NOTE: Can look up all PROTOTYPE defs. of all fcts. in math.h ...

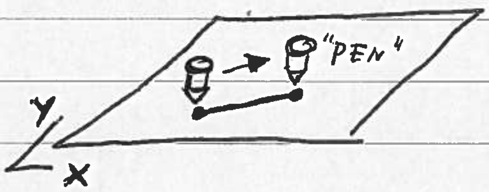
■ graphics.h



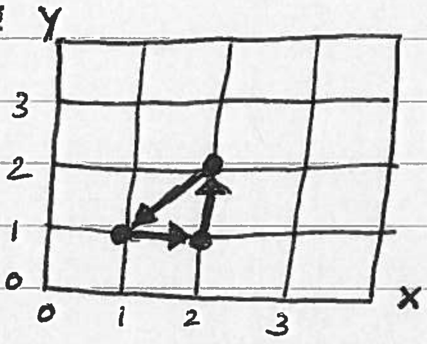
A = (2,2)
B = (3, 2)

"Absolute coords. of points A, B"

HERE: PEN PLOTTER MODEL



• EX:



COMMANDS:

```
MovePen (1,1);     /* no line drawn */
DrawLine (1,0);    /* difference vector */
                   /* to draw: (1,0) */
DrawLine (0,1);
DrawLine (-1,-1);
```

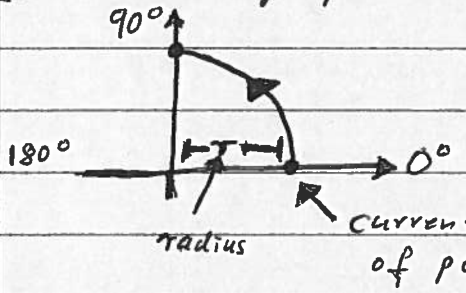
➡ 'MovePen (x,y)' uses absolute coords.
'DrawLine (dx,dy)' uses relative coords.

■ Use in a program:

```
1) InitGraphics();    /* initialize graphics */
2) MovePen (x,y);     /* move 'pen' to loc. (x,y) */
3) DrawLine (dx,dy);  /* draw a line from curr. */
                      /* location (x,y) to loc. */
                      /*        (x+dx, y+dy) */
```

... more graphics...

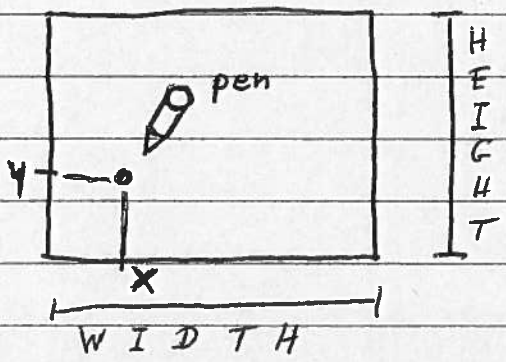
• **Ex:** ▪ Drawing part of circle / circular arc:



"DrawArc (r, start, sweep);"

here: start = 0°
sweep = 90°

▪ Information concerning window / pen configuration:



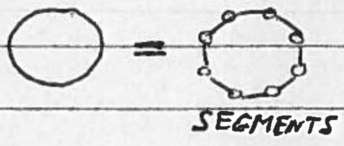
"GetWindowWidth ();"
/* window width */

"GetWindowHeight ();"
/* window height */

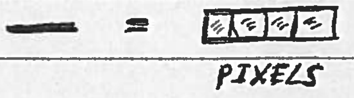
"GetCurrentX ();"
/* abs. x-coord. of pen */

"GetCurrentY ();" ...

• **NOTE:** → Do I need a circle command?
No!



→ Do I need a line command?
No!



```
/*
 * graphics.h
 */
#ifndef _graphics_h
#define _graphics_h
/*
 * InitGraphics: DESCRIPTION...
 */
void InitGraphics (void);
...
#endif
```

IF:
graphics.h = example of typical definition of an interface

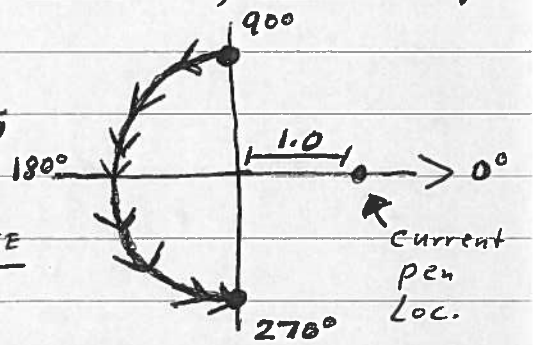
! "graphics.h provides information needed by client
• to properly utilize existing graphics functions."

•EX: Typical graphics.h prototype defs.:

```
void  MovePen (double x, double y);
void  DrawLine (double dx, double dy);
void  DrawArc (double r, double start, double sweep);
```

DrawArc (1.0, 90.0, 180.0);

positive values for COUNTER-CLOCKWISE rotation

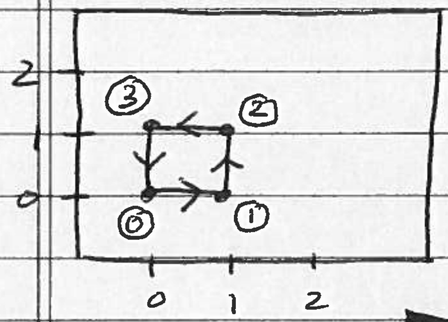


```
double  GetWindowWidth (void);
double      "    Height (void);
double  GetCurrentX (void);
double      "    Y (void);
```

and many more ...

Example Drawings

- /* Draw unit square */

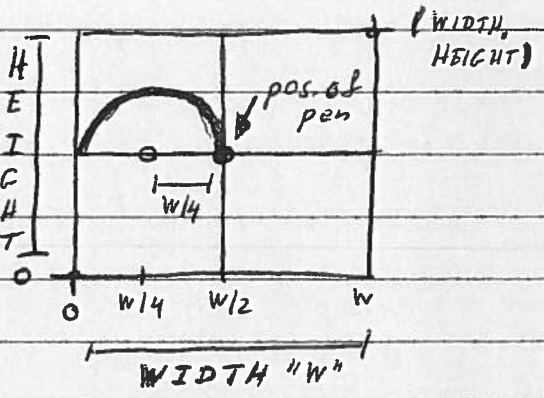


```
#include <stdio.h>
#include "genlib.h>
#include "graphics.h>
main()
{
 InitGraphics();
 MovePen(0.0, 0.0);
 DrawLine(1.0, 0.0);
    "    (0.0, 1.0);
    "    (-1.0, 0.0);
    "    (0.0, -1.0);
}
```

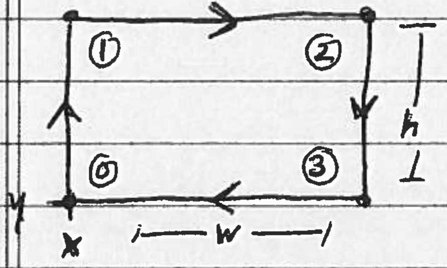
- ...

```
MovePen(GetWindowWidth()/2,
        GetWindowHeight()/2);

DrawArc(GetWindowWidth()/4,
        0.0, 180.0);
...
```



-



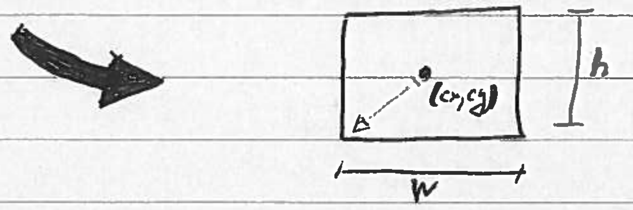
(x, y) is lower-left corner.

```
void DrawBox(dx, dy, dw, dh)
{                          /* d = double */
  MovePen(x, y);
  DrawLine(0, h);
    "     (w, 0);
    "     (0, -h);
    "     (-w, 0);
}
```

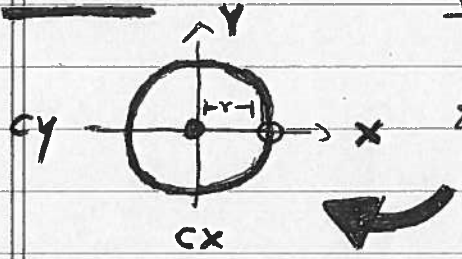
W4

- void DrawCentredBox (d cx, d cy, d w, d h)

```
{
  DrawBox ( cx - w/2, cy-h/2, w, h ) ;
}
```

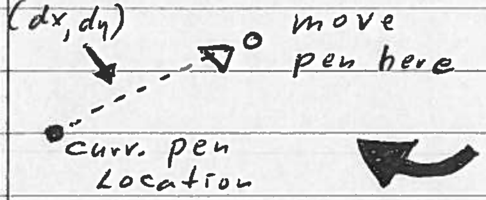


- Circle



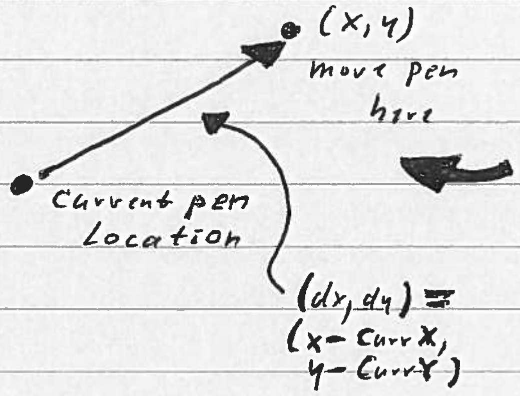
```
void DrawCentredCircle
        (d cx, d cy, d r )
{
  MovePen (cx+r , cy ) ;
  DrawArc ( r , 0.0, 360.0 ) ;
}
```

- "Invisible Line" – Move pen to new location without plotting



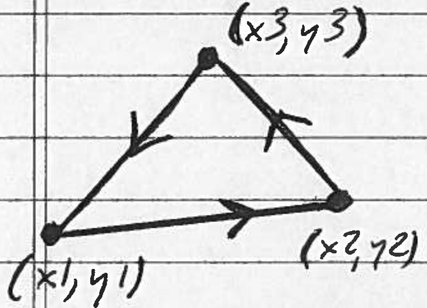
```
void AdjustPen (d dx, d dy)
{
  MovePen ( GetCurrentX() + dx,
            GetCurrentY() + dy ) ;
}
```

- "VisibleLine to Abs. Pos."



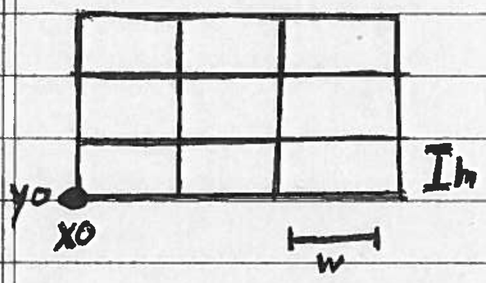
```
void DrawLineTo (d x, d y)
{
  DrawLine ( x - GetCurrentX(),
             y - GetCurrentY() ) ;
}
```

- **Triangle**



```
void DrawTriangle
     (d x1, d x2, d x3,
      d y1, d y2, d y3)
{
 MovePen (x1,y1);
 DrawLine (x2-x1, y2-y1);
   ...
 DrawLine (x1-x3, y1-y3);
}
```

- **(Cartesian) Grid**



```
void DrawGrid
     (d x0, d y0,
      d w, d h,
      int noX, int noY)
{int i, j;
 /* vertical lines */
 for (i=0; i<noX; i++)
 { MovePen (x0+i*w, y0);
   DrawLine(0.0, h*(noY-1));
 }
 /* horizontal lines */
  for (j=0; j<noY; j++)
  {
   MovePen (x0, y0+j*h);
   DrawLine (w*(noX-1), 0.0);
  }
```

Etc. Etc. Etc.