#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#define matrix\_size 100

//draw rectangle (x1,y1) length L wedth W

void rect (int x1, int y1,int L,int W, char arr[matrix\_size][matrix\_size]);

//draw line (x1,y1) to (x2,y2)

void line (int x1, int y1, int x2,int y2, char arr[matrix\_size][matrix\_size]);

//draw circle center( a,b ) and radius=r

void circle (int a, int b,int r, char arr[matrix\_size][matrix\_size]);

//drow Vertical line at x=x1 (x1,y1) to (x1,y2)

void vertical\_Line (int x1, int y1, int y2, char arr[matrix\_size][matrix\_size]);

int main()

{

char arr[matrix\_size][matrix\_size];

for(int y = 0; y < matrix\_size; y++)

for(int x = 0; x < matrix\_size; x++)

arr[y][x] = ' '; //fill the matrix with -

//draw rectangle (23,23) length 30 wedth 10

// rect (63, 60,30,10,arr);

//draw line (3,3) to (33,13)

line(35,35,60,10,arr); // B TO A

line(35,35,60,35,arr); // B TO C

line(30,40,85,40,arr); // REC LINE 1

line(30,55,85,55,arr); // REC LINE 3

line(15,40,30,55,arr); // \\ LEFT LINE

line(15,40,30,40,arr); // TOP LEFT LINE

line(85,55,100,40,arr); // // RIGHT LINE

line(85,40,100,40,arr); // TOP RIGHT LINE

//draw vertical Line at x=10

vertical\_Line(60,10,35,arr); // A TO C

vertical\_Line(60,35,40,arr); // LINE TO REC

vertical\_Line(30,40,55,arr); // REC LINE 4

vertical\_Line(85,40,55,arr); // REC LINE 2

//draw circle center( 17,20) radius=10

circle(40,47,4,arr); //LEFT

circle(58,47,5,arr); //CENTER

circle(75,47,4,arr); //RIGHT

//plot the matrix

for(int y = 0; y < matrix\_size; y++){

for(int x = 0; x < matrix\_size; x++)

printf("%c", arr[y][x]);

printf("\n");

}

return 0;

}

//draw rectangle (x1,y1) length L wedth W

void rect (int x1, int y1,int L,int W, char arr[matrix\_size][matrix\_size])

{

for(int y = 0; y < matrix\_size; y++)

{

for(int x = 0; x < matrix\_size; x++)

{

// see if we're within the range of x and y

if(x > x1 && x < x1+L && (y > y1 && y < y1+W))

{

arr[y][x] = '\*';

}

}

}

}

//draw circle center( a,b ) and radius=r

void circle (int a, int b,int r, char arr[matrix\_size][matrix\_size])

{

double EPSILON = 6;// need to be close to the radius value

for(int y = 0; y < matrix\_size; y++)

{

for(int x = 0; x < matrix\_size; x++)

{

// see if we're close to (x-a)\*\*2 + (y-b)\*\*2 == r\*\*2

if( fabs(pow((x-a),2)+pow((y-b),2)-pow(r,2)) < EPSILON)

{

arr[y][x] = '\*';

}

}

}

}

//draw line (x1,y1) to (x2,y2)

void line (int x1, int y1, int x2,int y2, char arr[matrix\_size][matrix\_size])

{

//0=y-m\*x-c

double EPSILON = 0.5, m, c;

m= (double) (y2-y1)/(x2-x1); // the slope of the line

c=m\*x1-y1; //the y-intercept of the line

for(int y = 0; y < matrix\_size; y++)

{

for(int x = 0; x < matrix\_size; x++)

{

// see if we're close to y = m\*x + c

if( x >= x1 && x <= x2 && fabs(y-m\*x+c) < EPSILON)

{

arr[y][x] = '\*';

}

}

}

}

//drow Vertical line at x=x1 (x1,y1) to (x1,y2)

void vertical\_Line (int x1, int y1, int y2, char arr[matrix\_size][matrix\_size])

{

for(int y = 0; y < matrix\_size; y++)

{

for(int x = 0; x < matrix\_size; x++)

{

// if x=x1 virtical Line

if( x == x1 && y<y2 && y>y1)

{

arr[y][x] = '\*';

}

}

}

}