**Data Structures**

Circle

**Contents:**

* Define a CIRCLE using typedef with center, radius and area.
* Dynamically Allocate memory for n circles.
* Generate *n* circles by randomly generating a center (*x*, *y*) in [8.0, 20.0] and

and radius in [2.0, 6.0].

* print the circles (center and radius of each circle).
* Writing whole program.

**Defining a CIRCLE using typedef with center, radius and area:**

* First we will create a structure(point) to represent two coordinates X,Y.

//vertex structure  
**typedef struct**{  
 **float** x;  
 **float** y;  
}point;

* Create another structure(CIRCLE) using the previous structure(point).

//structure for representing a circle

**typedef struct** {  
 **float** radius;  
 **float** area;  
 point p;  
}CIRCLE;

* Here p is a point which contains X and Y coordinates.

**Allocating memory for n circles.**

* malloc() for allocating memory dynamically.
* Create a pointer to structure for a circle and store the dynamically allocated

Memory in that pointer.

CIRCLE \*c;  
c=(CIRCLE\*)malloc(n\***sizeof**(CIRCLE));

**Generate *n* circles by randomly generating a center (*x*, *y*) in [8.0, 20.0] and**

**and radius in [2.0, 6.0].**

* Make the time as seed for generating random number.
* Generate random no between [8.0,20.0](center),[2.0,6.0](radius).
* Assign the generated random no’s to the structure variables.

CIRCLE \*genCircles(**int** n)  
{  
 **int** i;  
 srand(time(NULL));  
 CIRCLE \*c;  
 c=(CIRCLE\*)malloc(n\***sizeof**(CIRCLE));  
 **for**(i=0;i<n;i++)  
 {  
 c[i].p.x=rand()%13+8;  
 c[i].p.y=rand()%13+8;  
 c[i].radius=rand()%5+2;  
 }  
 **return** c;  
}

* Here the return type is pointer to structure(CIRCLE).

**printing the circles (center and radius of each circle):**

//printing center,radius,vertices  
**void** printCircles(CIRCLE \*c, **int** n)  
{  
 **int** i;  
 **float** pi=3.14;  
 **for**(i=0;i<n;i++)  
 {  
 printf("(%f,%f) radius=%f\n",c[i].p.x,c[i].p.y,c[i].radius);  
 }  
}

**Complete Program:**

#include<stdio.h>  
#include<time.h>  
#include<stdlib.h>  
  
**typedef struct** {  
 **float** x,y;  
}point;  
**typedef struct** {  
 **float** radius;  
 **float** area;  
 point p;  
}CIRCLE;  
  
CIRCLE \*genCircles(**int** n){  
 **int** i;  
 srand(time(NULL));  
 CIRCLE \*c;  
 c=(CIRCLE\*)malloc(n\***sizeof**(CIRCLE));  
 **for**(i=0;i<n;i++) {  
 c[i].p.x=rand()%13+8;  
 c[i].p.y=rand()%13+8;  
 c[i].radius=rand()%5+2;  
 }  
 **return** c;  
}  
//printing center,radius,vertices  
**void** printCircles(CIRCLE \*c, **int** n) {  
 **int** i;  
 **float** pi=3.14;  
 **for**(i=0;i<n;i++){  
 printf("(%f,%f) radius=%f area=%f\n",c[i].p.x,c[i].p.y,c[i].radius,pi\*c[i].radius\*c[i].radius);  
 }  
}  
**int** main(**int** argc,**char**\*argv[]) {  
 **int** n=3;  
 CIRCLE \*c;  
 c=genCircles(n);  
 printCircles(c,n);  
 **return** 0;  
}

(17.000000,16.000000) radius=5.000000 area=78.500003

(20.000000,8.000000) radius=2.000000 area=12.560000

(17.000000,20.000000) radius=5.000000 area=78.500003