computeCircle

A structure called circle is defined below. The structure consists of the radius of the circle and the (x,y) coordinates of its centre.

```
struct circle {
    double radius;
    double x;
    double y;
};
```

(a) Implement the function intersect() that returns 1 if two circles intersect, and 0 otherwise. Two circles intersect when the distance between their centres is less than or equal to the sum of their radii. The function prototype is given below:

```
int intersect(struct circle c1, struct circle c2);
```

(b) Implement the function contain() that returns 1 if *c1* contains *c2*, i.e. circle *c2* is found inside circle *c1*. Otherwise, the function returns 0. Circle *c1* contains circle *c2* when the radius of *c1* is larger than or equal to the sum of the radius of *c2* and the distance between the centres of *c1* and *c2*. The function prototype is given below:

```
int contain(struct circle *c1, struct circle *c2);
```

A sample program template is given below to test the functions:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define INIT_VALUE -1000
struct circle {
  double radius;
  double x;
  double y;
};
int intersect(struct circle, struct circle);
int contain(struct circle *, struct circle *);
int main()
{
  struct circle c1, c2;
  int choice, result = INIT_VALUE;

  printf("Select one of the following options: \n");
  printf("1: intersect()\n");
  printf("2: contain()\n");
  printf("3: exit()\n");
  do {
    result=-1;
```

```
        printf("Enter your choice: \n");
        scanf("%d", &choice);
        switch (choice) {
          case 1:
            printf("Enter circle 1 (radius x y): \n");
            scanf("%lf %lf %lf", &c1.radius, &c1.x, &c1.y);
            printf("Enter circle 2 (radius x y): \n");
            scanf("%lf %lf %lf", &c2.radius, &c2.x, &c2.y);
            result = intersect(c1, c2);
            if (result == 1)
              printf("intersect(): intersect\n");
            else if (result == 0)
              printf("intersect(): not intersect\n");
            else
              printf("intersect(): error\n");
            break;
          case 2:
            printf("Enter circle 1 (radius x y): \n");
            scanf("%lf %lf %lf", &c1.radius, &c1.x, &c1.y);
            printf("Enter circle 2 (radius x y): \n");
            scanf("%lf %lf %lf", &c2.radius, &c2.x, &c2.y);
            result = contain(&c1, &c2);
            if (result == 1)
              printf("contain(): contain\n");
            else if (result == 0)
              printf("contain(): not contain\n");
            else
              printf("contain(): error\n");
            break;
        }
      } while (choice < 3);
      return 0;
    }
    int intersect(struct circle c1, struct circle c2)
    {
        /* Write your code here */
    }
    int contain(struct circle *c1, struct circle *c2)
    {
        /* Write your code here */
    }
```

Some sample input and output sessions are given below:

(1) Test Case 1:
Select one of the following options:
1: intersect()()
2: contain()

3: exit()
Enter your choice:
1
Enter circle 1 (radius x y):
10 5 5
Enter circle 2 (radius x y):
5 1 1
intersect(): intersect
Enter your choice:
3

(2) Test Case 2:
Select one of the following options:
1: intersect()()
2: contain()
3: exit()
Enter your choice:
2
Enter circle 1 (radius x y):
10 5 5
Enter circle 2 (radius x y):
1 1 1
contain(): contain
Enter your choice:
3

(3) Test Case 3:
Select one of the following options:
1: intersect()()
2: contain()
3: exit()
Enter your choice:
1
Enter circle 1 (radius x y):
1 5 5
Enter circle 2 (radius x y):
1 10 10
intersect(): not intersect
Enter your choice:
3

(4) Test Case 4:
Select one of the following options:
1: intersect()()
2: contain()
3: exit()
Enter your choice:
2

Enter circle 1 (radius x y):
1 5 5
Enter circle 2 (radius x y):
1 10 10
contain(): not contain
Enter your choice:
3