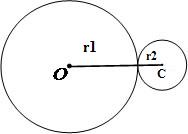
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Questão 2)** Usando os TADS Ponto e Círculo, construa uma main que:

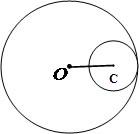
* 1. crie um vetor para armazenar 6 círculos
  2. preencha o vetor com 6 círculos. O valor de raio, x e y do centro devem ser gerados randomicamente
  3. exiba o vetor de círculos.
  4. crie um novo circulo cFora
  5. Para cada um dos círculos do vetor, exiba se cFora é:

1. Circunferência tangente.  
  
a) Tangentes externas  
Duas circunferências são tangentes internas quando possuem somente um ponto em comum e uma exterior à outra. A condição para que isso ocorra é que a distância entre os centros das duas circunferências seja equivalente à soma das medidas de seus raios.



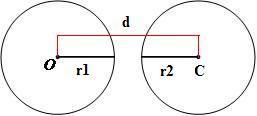
dOC = r1 + r2

b) Tangente interna  
Duas circunferências são tangentes internas quando possuem apenas um ponto em comum e uma esteja no interior da outra. A condição para que isso ocorra é que a distância entre os dois centros seja igual à diferença entre os dois raios.



dOC = r1 - r2

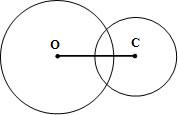
2. Circunferência externa.  
Duas circunferências são consideradas externas quando não possuem pontos em comum. A condição para que isso ocorra é que a distância entre os centros das circunferências deve ser maior que a soma das medidas de seus raios.



dOC > r1 + r2

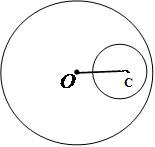
3. Circunferências secante.  
Duas circunferências são consideradas secantes quando possuem dois pontos em comum. A condição para que isso aconteça é que a distância entre os centros das circunferências deve ser menor que a soma das medidas de seus raios.

Não pare agora... Tem mais depois da publicidade ;)



dCO < r1 + r2

4. Circunferência interna.  
Duas circunferências são consideradas internas quando não possuem pontos em comum e uma está localizada no interior da outra. A condição para que isso ocorra é que a distância entre os centros das circunferências deve ser equivalente à diferença entre as medidas de seus raios.



dOC < r1 - r2

* 1. Por Marcelo Rigonatto  
     Especialista em Estatística e Modelagem Matemática

I

**TAD CÍRCULO**

/\* Tipo exportado \*/

#include "PONTO.h"

**typedef** struct circulo Circulo**;**

/\* Funções exportadas \*/

/\* Função cria - Aloca e retorna um círculo com centro (x,y) e raio r \*/

Circulo**\*** circ\_cria**(**float x**,** float y**,** float r**);**

/\* Função libera - Libera a memória de um círculo previamente criado \*/

void circ\_libera**(**Circulo**\*** c**);**

/\* Função area - Retorna o valor da área do círculo \*/

float circ\_area**(**Circulo**\*** c**);**

/\* Função interior - Verifica se um dado ponto p está dentro do círculo \*/

int circ\_interior**(**Circulo**\*** c**,** Ponto**\*** p**);**

/\* Função getCirculo - Retorna uma string lo no formato (c:centro,r:raio)\*/

char**\*** circ\_getCirculo**(**Circulo**\*** p**);**

/\* Função exibe -Escreve na tela o círculo no formato (c:centro,r:raio)\*/

void circ\_exibe**(**Circulo**\*** p**);**

/\* Função compara - Retorna:

valor<0 (negativo) se o 1º círculo é menor que o 2º,

valor == 0, se são iguais

valor>0 se o 1º círculo é maior que o 2º

Um círculo é menor que outro se sua área for menor que a do outro\*/

int circ\_compara**(**Circulo**\*** c1**,** Circulo**\*** c2**);**

/\* Função clona – cria um novo cículo igual ao recebido\*/

Circulo**\*** circ\_clona**(**Circulo**\*** c1**);**

/\* get e set do raio e do centro\*/

float circ\_getRaio**(**Circulo**\*** c**);**

Ponto**\*** circ\_getCentro**(**Circulo**\*** c**);**

void circ\_setRaio**(**Circulo**\*** const**,**float r**);**

void circ\_setCentro**(**Circulo**\*** c**,**Ponto **\***p**);**

**TAD PONTO**

/\* Tipo exportado \*/

**typedef** struct ponto Ponto**;**

/\* Funções exportadas \*/

/\* Função cria - Aloca e retorna um ponto com coordenadas (x,y) \*/

Ponto**\*** pto\_cria**(**float x**,** float y**);**

/\* Função libera - Libera a memória de um ponto previamente criado \*/

void pto\_libera**(**Ponto**\*** p**);**

/\* Função acessa - Retorna os valores das coordenadas de um ponto \*/

void pto\_acessa**(**Ponto**\*** p**,** float**\*** x**,** float**\*** y**);**

/\* Função atribui - Atribui novos valores às coordenadas de um ponto \*/

void pto\_atribui**(**Ponto**\*** p**,** float x**,** float y**);**

/\* Função distancia - Retorna a distância entre dois pontos \*/

float pto\_distancia**(**Ponto**\*** p1**,** Ponto**\*** p2**);**

/\* Função getPonto - Retorna uma string com os valores do ponto formato(x,y)\*/

char**\*** pto\_getPonto**(**Ponto**\*** p**);**

/\* Função exibe -Escreve na tela o ponto no formato (xxx.xx,yyy.yy)\*/

/\* Função exibe -Escreve na tela o ponto no formato(xxx.xx,yyy.yy)\*/

void pto\_exibe**(**Ponto**\*** p**);**

int pto\_menorque**(**Ponto**\*** pt1**,**Ponto**\*** pt2**);**

/\* Função menorque -recebe dois pontos e retorna 1 se o pt1 está mais pr´ximo da origem, 0 cc

**Questão 3) 3.0**

Inclua no seu TAD Lista encadeada circular (Tarefa 7) a funcionalidade

copia\_sublista: esta função recebe uma lista (l1 ) e , uma informação e uma quantidade de nós. Caso a informação esteja na lista l1, constrói uma nova ista com os n nós subsequentes de l1

Questão 4)

O setor de compras de uma empresa resolveu implementar a seguinte estratégia para atender os

pedidos de compras de materiais de escritório de suas duas filiais sem melindrar nenhuma das

filiais:

Cada filial, faz sua própria relação de itens a comprar. Esta relação está organizada por

prioridade de necessidade, isto é, os itens mais necessários estão em primeiro lugar. Um item

é definido pelo código do material e da quantidade necessária.

O setor de compras une as duas relações, alternando um pedido da filial 1 e um pedido da filial

2. Esta estrutura deve estar organizada pelo código do ítem a comprar.

a) Qual estrutura de dado para representar os pedidos das filiais? Por que? Descreva esta

estrutura em C

b) Qual estrutura de dado para representar o processo? Por que?