1)A programação modular é um exemplo de paradigma orientado a objetos pois lida com classes e objetos.Certo ou Errado.Justifique.

R:Correto.Embora a programação modular não utilize nomes como classe,objeto e métodos, podemos fazer uma analogia entre os elementos dela com os da programação orientada a objeto. Imagine a seguinte situação: Temos o módulo peça que possui uma estrutura com o mesmo nome, a qual contém cor e id. Este também possui diversas funções de acesso. Agora, imagine que declaramos 6 ponteiros para peça em um módulo distinto da mesma aplicação. Podemos considerar esses seis ponteiros como sendo instâncias da classe peça(objetos), os campos cor e id como sendo os atributos da classe e as funções de acesso como sendo os métodos da classe. Portanto, sim, de uma certa perspectiva a programação modular pode ser considerada uma programação orientada a objetos

2)Apresente o esquema de algoritmo do quickSort apontando ao menos um hotspot preenchido com parâmetros do tipo ponteiro para função.

R:

3)Faça a argumentação de corretude completa do algoritmo abaixo:

INICIO

Ind <— 1

ENQUANTO IND <= LL FAÇA

INICIO

ATUAL <— ELE[IND]

AUX <— IND - 1

ENQUANTO AUX >= 1 E ELE[AUX] > ATUAL

INICIO

ELE[AUX+1] <— ELE[AUX]

AUX <— AUX + 1

FIM

ELE[AUX + 1] <— ATUAL

IND <— IND + 1

FIM

FIM

R:

Argumentação de Corretude

=>AE

INICIO

Ind <— 1

=>AI1

ENQUANTO IND <= LL FAÇA

INICIO

ATUAL <— ELE[IND]

AUX <— IND - 1

ENQUANTO AUX >= 1 E ELE[AUX] > ATUAL

INICIO

ELE[AUX+1] <— ELE[AUX]

AUX <— AUX + 1

FIM

ELE[AUX + 1] <— ATUAL

IND <— IND + 1

FIM

FIM

=>AS

Argumentaçao de Sequencia:

-AE:

-o vetor ELE pode estar vázio, ou com um ou mais elementos.

-AS:

O vetor ELE está ordenado em ordem crescente se não for vazio. Caso ele seja vazio não faz nada

-AI1:

IND vale 1, ou seja, aponta para o primeiro elemento do vetor caso ele não seja vazio

BLOCO B1:

INICIO

FIM

Argumentação de repetição:

AE: AI1

AS: AI2

AINV:

a)O vetor ELE está dividido em dois grupos, o já pesquisado e o a pesquisar.

b)Ind aponta para um elemento do vetor ELE.

-AE => AINV

Pela AE, IND aponta para o primeiro elemento do vetor, portanto o conjunto dos já pesquisados é vazio e o dos a pesquisar corresponde ao vetor inteiro.

-AE && (C == F) => AS

Pela AE o vetor é vazio, logo LL = 0 e ind = 1, portando a repetição não é executada. Vale AS.

-AE && (C == T) + B1 => AINV

Pela AE o vetor não é vazio e ind é incrementado em uma unidade, o que faz com que ele aponte para o segundo elemento do vetor. O conjunto dos elementos já pesquisados possui agora o primeiro elemento do vetor e o de a pesquisar corresponde ao resto do vetor, portanto vale AINV.

-AINV && (C == T) + B1 => AINV

Para que AINV continue valendo, B1 deve garantir que um elemento passe do conjunto a pesquisar para o já pesquisado e IND aponte para o próximo elemento de a pesquisar

-AINV && (C == F) => AS

Nesse caso IND vale LL + 1, logo o conjunto a pesquisar corresponde ao vetor inteiro e o já pesquisado é vazio e o vetor está ordenado em ordem crescente, logo AS é válida.

-Término:

Como a cada ciclo o elemento apontado por IND vai do conjunto a pesquisar para o já pesquisado e a quantidade de elementos deste conjunto é finita, a repetição termina após um número finito de passos.

ATUAL <— ELE[IND]

=>AI2

AUX <— IND - 1

=>AI3

ENQUANTO AUX >= 1 E ELE[AUX] > ATUAL

INICIO

ELE[AUX+1] <— ELE[AUX]

AUX <— AUX + 1

FIM

=>AI4

ELE[AUX + 1] <— ATUAL

=>AI5

IND <— IND + 1

=>AI6

FIM

Argumentação de Sequencia:

AI2: Atual recebe o valor do elemento apontado por IND

AI3: AUX aponta para o o elemento anterior a AUX, ou vale 0, se IND estiver apontando para o primeiro elemento do vetor

AI4: AUX aponta para um elemento no vetor, anterior a IND, o qual é menor ou igual ao elemento apontado por IND(ATUAL), ou aux = 0. A porção do vetor entre o segundo elemento do vetor e o apontado por IND está ordenada em ordem crescente

AI5: O elemento posterior ao apontado por AUX possui o mesmo valor que o elemento apontado por IND(ATUAL). A porçao do vetor entre o primeiro elemento do vetor e o apontado por IND está ordendado em ordem crescente.

AI6: IND aponta para o próximo elemento a ser pesquisado no vetor.

B2:

ENQUANTO AUX >= 1 E ELE[AUX] > ATUAL

INICIO

ELE[AUX+1] <— ELE[AUX]

AUX <— AUX + 1

FIM

Argumentação de Repetição:

AE: AI3

AS: AI4

AINV:

a)O subvetor de ELE compreendido entre o primeiro elemento do vetor e o elemento apontado por AUX está dividido em dois grupos, o já pesquisado e o a pesquisar.

b)AUX aponta para um elemento do subvetor.

AE => AINV

-Pela AE, AUX está apontando para o ultimo elemento do subvetor e, portanto, o conjunto já pesquisado do subvetor é vazio e o a pesquisar contém o subvetor inteiro, portando, vale AINV

AE && (C == F) => AS

- Pela AE, caso a condição seja falsa ou aux = 0, ou aux>0 e o elemento apontado por AUX é menor ou igual ao valor de ATUAL. Em Ambos os casos vale AS.

AE && (C == T) + B2 => AINV

-Pela AE, IND terá que apontar para um elemento no vetor posterior ao primeiro e o elemento apontado por AUX é maior do que o valor de ATUAL. Assim o bloco B2 será executado, o que fará com que o conjunto dos já visitados contenha somente o último elemento do subvetor e o de a visitar contenha o resto do subvetor.

AINV && (C == T) + B2 => AINV

-Para AINV ser válida, B2 deve garantir que um elemento do subvetor seja transferido do conjunto dos a pesquisar para os já pesquisados e AUX aponte para o próximo elemento do subvetor.

AINV && (C == F) => AS

-No último ciclo, teremos três opções ou AUX é igual a zero, ou elemento apontado por AUX é menor ou igual ao valor de ATUAL, ou ambas as condições são verdadeiros. Em qualquer um dos casos a porção do subvetor entre o segundo elemento e o último está ordenado em ordem crescente. Em todos os casos vale AS.

Término: Como a cada ciclo um elemento do conjunto a pesquisar passa pro conjunto de já pesquisados e quantidade de elementos deste conjunto é finita, a repetição termina em um número finito de passos.

B3:

INICIO

ELE[AUX+1] <— ELE[AUX]

=>AI7

AUX <— AUX - 1

=>AI8

Argumentação de sequencia:

-AI7

Os elemento posterior ao apontado por AUX possui o mesmo valor do elemento apontado por AUX.

-A18

Aux aponta para o elemento anterior ao apontado anteriormente por AUX.

4)Transforme uma estrutura grafo em auto-verificável

R:

5)Gere quatro deturpações possíveis para a estrutura da questão 4.

R:Deturpações:

a)Eliminar o elemento corrente da estrutura.

b)Atribuir NULL ao ponteiro para um vértice sucessor.

c)Atribuir lixo ao ponteiro para um vértice sucessor.

d)Destacar um vertice do grafo sem liberá-lo com free.

6)Apresente o código do verificador para a estrutura da questão 4.

R:

GRA\_Verificador(GRA\_tppGraph g) {

int i, qtdArestas = 0;

if(g->vetAdjs == NULL) {

CNT\_Contar(“erro-vetAdjsNulo”);

}

for(i=0;i<g->qtdVert;i++) {

if(g->vetAdjs[i] == NULL) {

CNT\_Contar(“erro-listaDeAdjsNula”);

}

if(g->vetAdjs[i] == LIXO) {

return GRA\_tpCondRetErroListaDeAdjsLixo;

}

qtdArestas += g->vetAdjs[i]->qtdNos;

}

if(qtdArestas != g->qtdArestas) {

return GRA\_tpCondRetErroQtdArestasErrada;

}

}

7)Distribua controladores de cobertura no verificador e informe o relatório final de contagem executando apenas as deturpações da questão 5.OBS:Deve ser inserido o menor número de controladores necessários para testar completamente o verificador pelo critério caixa aberta.

R:(Ainda incompleta)

GRA\_Verificador(GRA\_tppGraph g) {

int i,qtdArestas = 0;

if(g->vetAdjs == NULL) {

CNT\_Contar(“erro-vetAdjsNulo”);

return GRA\_tpCondRetErroVetAdjsNulo;

}

for(i=0;i<g->qtdVert;i++) {

if(g->vetAdjs[i] == NULL) {

return GRA\_tpCondRetErroListaDeAdjsNula;

CNT\_Contar(“erro-listaDeAdjsNula”);

}

if(g->vetAdjs[i] == LIXO) {

CNT\_Contar(“erro-listaDeAdjsApontaLixo”);

return GRA\_tpCondRetErroListaDeAdjsLixo;

}

qtdArestas += g->vetAdjs[i]->qtdNos;

}

if(qtdArestas != g->qtdArestas) {

CNT\_Contar(“erro-listaDeAdjsApontaLixo”);

return GRA\_tpCondRetErroQtdArestasErrada;

}

}

8)Apresente um exemplo de codigo em C utilizando trace de evolução.

R:

void loop(int n) {

int i;

for(i=0;i<n;i++) {

printf(“%d\n”,i); //trace de evolução

}

}

9)Explique a vantage da instrumentação em relação aos testes convencionais no que diz respeito ao esforço de diagnose.

R: Utilizando a instrumentação, vários erros que são extremamente difíceis de identificar e que estão fortemente relacionados com violações de assertivas estruturais, tornam-se bem mais fáceis de serem percebidos, pois com a instrumentação sempre estamos verificando se a estrutura instrumentada está de acordo com suas assertivas estruturais.

10)Qual é o objetivo de se definir a assertiva invariante de uma repetição?Apresente o código de uma assertiva invariante executável.

R: