Trabalho Teórico 3 - Exercícios da Unidade 0 - Nivelamento

1 de setembro de 2021

1. Parte

(a) Resposta:

```
public static boolean busca(int[] array, int elemento){
for(int i=0; i<array.length; i++){
if(array[i] == elemento)
return true;
}
return false;
}
```

(b) Resposta:

```
public static boolean buscaBinaria(int[] array, int elemento){
int meio;
int limiteS = array.length-1;
int limiteI = 0;

while(limiteI <= limiteS){
meio = (limiteS+limiteI)/2;
if(elemento == array[meio])
return true;
if(elemento < array[meio])
limiteS = meio - 1;
else
limiteI = meio + 1;
}
return false;
}
```

(c) Resposta:

```
public static void maiorMenor(int[] array){
int menor = array[0];
int maior = menor;

for(int i=0; i<array.length; i++){
if(array[i] > maior)
maior = array[i];
if(array[i] < menor)
menor = array[i];
}
MyIO.println("Maior: " + maior);
MyIO.println("Menor: " + menor);
}
```

(d) Resposta:

Mínimo de comparações possíveis.

(e) Resposta:

Caracter é testado via-parâmetro, testando se é vogal ou não.

(f) Resposta:

Segundo.

(g) Resposta:

As variáveis poderiam ser menor.

(h) Resposta:

No primeiro, irá retornar o valor de i e depois decrementá-la em 1, já no segundo, primeiro irá decrementá-la em 1 e depois retornar o valor da variável i.

(i) Resposta:

b é uma progressão de 0 até 1, a s de 0 à 32767, a i de 0 à 2147483647 e l de 0 à 2147483647.

2. Parte

(a) Resposta:

```
public static String conteudoArq(String path) throws IOException {
BufferedReader Arq = new BufferedReader(new FileReader(path));

String conteudo = "";
String aux = "";
while (aux != null) {
conteudo += aux;
aux = Arq.readLine();
if(aux!=null)
aux+='\n';
}
Arq.close();

return conteudo;
}

public static void copiaArquivo(String pathOriginal, String pathCopia) throws IOEx
BufferedWriter Arq = new BufferedWriter(new FileWriter(pathCopia));
Arq.append(conteudoArq(pathOriginal));
Arq.close();
}
```

(b) Resposta:

```
public static String copiarParaString(String path) throws IOException{
BufferedReader arq = new BufferedReader(new FileReader(path));

String conteudo = "";
String aux;

while(true){
aux = arq.readLine();
if(aux == null)
break;
```

```
else{
conteudo += aux;
conteudo += '\n';
}
}

arq.close();
return conteudo;
}
public static void substituiParaOriginal(String path, String conteudo) throws IOEx
BufferedWriter arq = new BufferedWriter(new FileWriter(path));

arq.append(conteudo);

arq.close();
}
public static void inserePilha(String path, String frase, String conteudoExistente)
BufferedWriter arqF = new BufferedWriter(new FileWriter(path));

frase += '\n';
frase += conteudoExistente;

arqF.append(frase);

MyIO.println(conteudoArq(path));
arqF.close();
}
public static void deletar(String path) throws IOException{
BufferedReader arq = new BufferedReader(new FileReader(path));

boolean primeiraRep = true;
String aux;
String strFinal = "";

while(true){
aux = arq.readLine();
System.out.println(aux);
if(aux == null)
break;
else{
if(primeiraRep)
primeiraRep = false;
else{
strFinal += aux;
strFinal += '\n';
}
}
}

arq.close();
}

public static String conteudoArq(String path) throws IOException {
BufferedReader Arq = new BufferedReader(new FileReader(path));
```

```
String conteudo = "";
String aux = "";
while (aux != null) {
conteudo += aux;
aux = Arq.readLine();
if(aux!=null)
aux+='\n';
}
Arq.close();

return conteudo;
}

public static void main(String[] args) throws IOException{
String path = "pilha.txt";
String conteudoInicial = copiarParaString(path);


String aux;
String frase;
int opcao;

do{

MyIO.println("\nInsira uma opcao\n1. Inserir\n2. Remover\n3. Listar\n4. Sair");
opcao = MyIO.readInt();
if(!(opcao<1 || opcao>4)){
switch(opcao){
case 1:
frase = MyIO.readLine();
aux = conteudoArq(path);
inserePilha(path, frase, aux);
break;

case 2:
System.out.println("M todo indispon vel, bug na hora de representar a fun  o, e
break;

case 3:
MyIO.println(conteudoArq(path));
break;

default:
MyIO.println("ERROR, INSIRA OUTRA OPCAO");
}
}
else
MyIO.println("\nOpcao invalida, insira outra.\n");

}while(opcao!=4);
MyIO.println(conteudoArq(path));
substituiParaOriginal(path, conteudoInicial);
```

```
}
```

3. Parte

(a) Resposta:

```
public static float media(int[] vetor){
float mediaE = 0;
for(int i=0; i<vetor.length; i++)
mediaE+=vetor[i];

return mediaE/vetor.length;
}

public static void main(String[] args){
int n = MyIO.readInt();
int[] vetor = new int[n];

for(int i=0; i<n; i++)
vetor[i] = MyIO.readInt();

float mediaElementos = media(vetor);

for(int i=0; i<n; i++){
if(vetor[i] > mediaElementos)
MyIO.println(vetor[i]);
}
}
```

4. Parte

(a) Resposta:

A recursividade está na 6 linha de ambos os códigos, sendo no primeiro uma função que calcula o fatorial. E no segundo ele chama a função duas vezes. Com isso, é a realizada a soma das duas funções, formando uma soma de Fibonacci.

(b) Resposta:

A variável i e chama a função novamente, isso até atingir a condição de parada e, logo quando a atinge, imprime o valor da variável i correspondente às outras chamadas de função.

(c) Resposta:

```
public static int multiplicacao(int a, int b){
int soma = 0;
if(b>0)
soma = a + multiplicacao(a, b-1);
return soma;
}
```

(d) Resposta:

```
public static int maiorElemento(int[] vetor, int tamanho, int maior){
if(tamanho == 0)
return maior;
else{
if(vetor[tamanho] > maior)
```

```
maior = vetor[tamanho];
return maiorElemento(vetor, --tamanho, maior);
}
}

public static void main(String [] args){
int n = MyIO.readInt();
int[] vetor = new int[n];

for(int i=0; i<n; i++)
vetor[i] = MyIO.readInt();

MyIO.println(maiorElemento(vetor, n-1, vetor[0]));
}
```

(e) Resposta:

```
public static boolean isPalindromo(char[] vect, int pInicial, boolean boo){
if(pInicial >= vect.length/2)
return boo;
else{
if(vect[pInicial] == vect[(vect.length-1)-pInicial])
return isPalindromo(vect, ++pInicial, boo);
else
return isPalindromo(vect, vect.length-1, false);
}
}
```

(f) Resposta:

```
public static int quantidadeVogais(char[] frase, int tamanho, int vogais){
if(isVogal(frase[tamanho]))
++vogais;

if(tamanho == 0)
return vogais;
else
return quantidadeVogais(frase, --tamanho, vogais);
}
```

(g) Resposta:

```
public static int quantidadeMinusculas(char[] frase, int tamanho, int qtd){
if(isMinuscula(frase[tamanho]))
++qtd;

if(tamanho == 0)
return qtd;
else
return quantidadeMinusculas(frase, --tamanho, qtd);
}
```

(h) Resposta:

```
public static int ordenacao(int[] array, int pos){
int menor = array[pos];
```

```
int aux = pos;

for(int i=pos+1; i<array.length; i++){
if(array[i] < menor){
menor = array[i];
aux = i;
}
}
array[aux] = array[pos];
array[pos] = menor;

if(pos == array.length-1)
return 0;
else
return ordenacao(array, ++pos);
}
```

(i) Resposta:

```
public static int e2(int n) {
if (n == 0)
return 1;
else
return (int) Math.pow((n - 1), 2);
}
```

5. Parte

(a) Resposta: São diferentes, pois estão comparando dois objetos e não propriedades.

(b) Resposta:

i. E: O segundo parâmetro é int e o terceiro é char.

ii. V: O objeto C é um ponteiro para o B, quando alterado, altera o outro objeto também.

iii. V: Existem métodos para alteração do atributo.

iv. E: é mutável.

6. Parte

(a) Resposta:

```
Retangulo::Retangulo(double base, double altura){
this->base = base;
this->altura = altura;
}
double Retangulo::getArea(){
double area = this->altura * this->base;
return area;
}
double Retangulo::getPerimetro(){
double perimetro = (this->altura*2) + (this->base*2);
return perimetro;
}
double Retangulo::getDiagonal(){
double diagonal = sqrt(pow(this->altura, 2) + pow(this->base, 2));
```

```
return diagonal;
}

int main(){
Retangulo *r1 = new Retangulo(10, 2);
Retangulo *r2 = new Retangulo(20, 4);

cout<<"\n"<<r2->getArea();
cout<<"\n"<<r1->getArea();

cout<<"\n"<<r2->getPerimetro();
cout<<"\n"<<r1->getPerimetro();

cout<<"\n"<<r2->getDiagonal();
cout<<"\n"<<r1->getDiagonal();

return 0;
}

class Retangulo{
private:
double base;
double altura;

public:
Retangulo(double base, double altura);
double getArea();
double getPerimetro();
double getDiagonal();
};
```

(b) Resposta:

```
#include <math.h>

class ponto{
private:
double x;
double y;
int id;
int static nextID = 0;

public:
ponto();
ponto(double x, double y);
void alterarId();
int getId();
static void alterarNextID();
int getNextId();
void setX(double valor);
double getX();
void setY(double valor);
double getY();
double dist(double ponto2);
double dist(ponto ponto);
```

```
int isTriangulo(ponto ponto1, ponto ponto2, ponto ponto3);
double getAreaRetangulo(ponto ponto);
};

ponto::ponto(){
this->x = 0;
this->y = 0;
}
ponto::ponto(double x, double y){
this->x = x;
this->y = y;
}
void ponto::alterarId(){
this->id = this->nextID;
ponto::alterarNextID();
}
int ponto::getId(){
return this->id;
}
void ponto::alterarNextID(){
nextID++;
}
int ponto::getNextId(){
return this->nextID;
}
double ponto::dist(double ponto2){
return this->x - ponto2;
}
double ponto::dist(ponto ponto){
return this->x - ponto.getX();
}
int ponto::isTriangulo(ponto ponto1, ponto ponto2, ponto ponto3){
if(ponto1.getX() > abs(ponto2.getX() - ponto3.getX()) && ponto1.getX() < ponto2.get
if (ponto2.getX() > abs(ponto1.getX() - ponto3.getX()) && ponto2.getX() < ponto1.ge
if(ponto3.getX() > abs(ponto1.getX() - ponto2.getX()) && ponto3.getX() < ponto1.get
return 1;
else
return 0;
}
else
return 0;
}
else
return 0;
}
double ponto::getAreaRetangulo(ponto ponto){
return ponto.getX()*ponto.getY();
}
```