u évora
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

EINF MA CID

Disciplina: Estruturas de Dados e Algoritmos I-2019/2020 Prova: mini-teste 2 (5-11-2019)

Esta prova tem a duração de **1 hora** e é **sem consulta**. Identifique TODAS as folhas de teste.

1. Considere no código Java, que implementa as **LinkedList**, foi definido o método q1, como a seguir se apresenta:
   Nota: As linkedList, são implementadas com um dummy node(`header`), no inicio da lista, sendo o último nó da lista, referenciado por `tail`.

```
public  LinkedList<T> q1 (int n,int m,int k){
        SingleNode<T> no=header().next;
        LinkedList<T> res=new LinkedList<T>();
        int i;
        for (i=0;i<n;i++){
            if (no==null)
                return res;
            else
                no=no.next;
        }

        while(no!=null && i<m){
            res.add(no.element());
            for (int p=0;p<k;p++){
                if(no!=null) {
                    no = no.next;
                    i++;
                }
                else return res;
            }
        }
        return res;
```

Assumindo $l = [1; 2; 3; 4; 5; 6; 7; 8; 9]$

(a) Qual o resultado da execução de `l.q1(0,10,5)`?

(b) Qual o resultado da execução de `l.q1(2,7,1)`?

(c) Qual a complexidade do método `q1`?

2. Considere que na classe **LinkedList**, foi definido o código `q2` e considere o excerto do método main que se apresenta :

```
public static <E> void q2(SingleNode<E> x){
    SingleNode<E> a=x.next;
    SingleNode<E> b=x.next.next;
    x.next=b;
    a.next=b.next;
    b.next=a;
}
public static void main(String[] args){
        for (int i=1;i<10;i++)
            l.add( i );
        q2(l.header.getNext());
        System.out.println(l);
        SingleNode<Integer> x=l.header();
        int i= 0;
        while (i++<l.size()-2) {
            q2( x );
            x = x.next;
        }
        System.out.println(l);
 }
```

(a) Qual a lista obtida no 1º "System.out.println(l)"?

(b) Qual a lista obtida no 2º "System.out.println(l)?

(c) Qual a complexidade do método `q2`?

3. Considere que na classe **BTree**, definiu o método `q3`

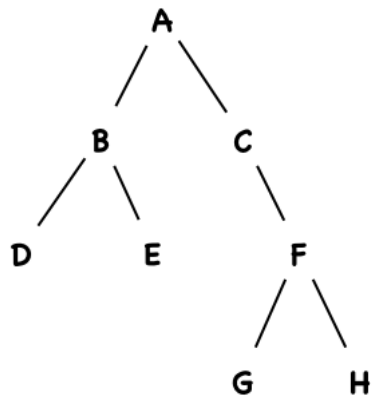
```
public static <T> LinkedList<T> q3(BNode<T> n){
        if (n==null)
            return new LinkedList<T>();
        else
        if (n.esq()==null && n.dir()== null) {
            LinkedList<T> l = new LinkedList<T>();
            l.add(0, n.elemento);
            return l;
        }
        else{
            LinkedList<T> X=q3(n.esq);
            LinkedList<T> Y=q3(n.dir);
            for(T a:Y)
                X.add(0,a);
            return X;
        }
```

Considere BTree<Integer> b1= new BTree<>(10, new BNode<Integer>(2, new BNode<Integer>(5),null) , new BNode<Integer>(8, new BNode<Integer>(6),new BNode<Integer> (4)));

(a) Desenhe a árvore `b1`

(b) Qual a lista obtida por `q3(b1.root)`?

4. Considere a seguinte árvore:



(a) Apresente a lista dos nós visitados usando um percurso pré-ordem

(b) Apresente a lista dos nós visitados usando um percurso pós-ordem

(c) Apresente a lista dos nós visitados usando um percurso em-ordem

(d) Qual a altura do nó C?

(e) Qual a profundidade do nó A?

(f) Qual a altura da árvore?

5. Implemente o método *static void boolean ordenado(LinkedList<Integer> x)* que retorna true se a lista passada por argumento estiver ordenada(ordem crescente). Indique a complexidade do método.

6. Implemente o método *static void LinkedList<Integer> fusão(LinkedList<Integer> x, LinkedList<Integer> y)* que retorna a fusão ordenada das listas x, y. Deve ser o mais eficiente possível. Indique a complexidade do método implementado. Por exemplo se $x = [1; 3; 19; 26]$, e $y = [2, 7]$ a fusão deve resultar na lista $[1; 2; 3; 7; 19; 26]$