UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Ciência, Tecnologia e Ensino Superior

U.C. 21046

Estruturas de dados e algoritmos fundamentais

11 de Setembro de 2015

INSTRUÇÕES

PARA A RESOLUÇÃO DO EXAME, ACONSELHA-SE QUE LEIA ATENTAMENTE O SEGUINTE:

1. O exame está dimensionado para um tempo de resolução de duas horas e trinta minutos
2. Este exame cobre toda a matéria da unidade curricular.
3. O exame é constituído por 4 questões.
4. A cotação de cada uma das questões está indicada.
5. O exame deve ser resolvido SEM CONSULTA.
6. O exame deve ser escrito a esferográfica.
7. A interpretação dos enunciados das questões também faz parte da sua resolução, pelo que, se existir alguma ambiguidade, deve indicar claramente como foi resolvida.

Bom Trabalho!

1º Questão (5 Valores)

a) Construa uma função recursiva que calcula a soma dos primeiros **K** múltiplos de 3. O cabeçalho da função é o seguinte: (2 val)

```
int somaKMultiplos3(int k)
{
  if(k>0)
    return 3*k + somaKMultiplos(k-1);
  else
    return 0;
}
```

b) Construa uma função iterativa que dado um vector com **N** elementos indica o comprimento da maior sequencia de números pares, em que todos os valores dessa sequência são divisores de **K**. Indique a complexidade do seu algoritmo. O cabeçalho da função é o seguinte: (2 val)

```
int maiorSeqParesDivK(int k, int n, int vec[])
{
  int atual=0;
  int maior=0;
  int i;

  for(i=0;i<n;i++)
    if(vec[i]%2==0 && vec[i]%k==0)
      atual++;
    else
    {
      if(atual>maior)
        maior=atual;
      atual=0;
    }
   return maior;
}
```

A complexidade é O(n)

Ex: Suponha X=[1,4,12,6,24,2,8]; K=3; N=7
somaKMultiplos3(5) = 3+6+9+12+15 = 45
maiorSubSeqParesDivK(K,N,X)=3

c) Indique a complexidade do seguinte algoritmo justificando. (1 val)

```
int soma=0;
for(int i=0; i<n; i++)
    for(int j=i; j<=n;j++)
        soma+=vec[i][j]
```

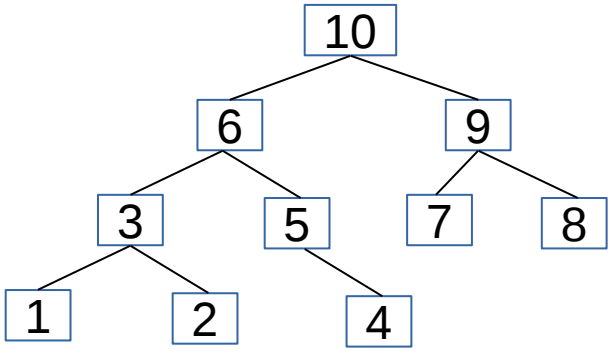
A complexidade é $O(n^2)$ pois temos um duplo ciclo for em que o primeiro é executado n vezes e o segundo ciclo é executado (n-1), (n-2) …
Logo um limite superior para a complexidade seria $O(n^2)$.

2º Questão (5 Valores)

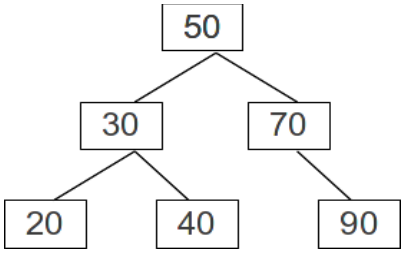
a) Desenhe uma árvore binária de altura 3 cuja travessia em pós-ordem (post-order) é a seguinte: (2 val)

1-2-3-4-5-6-7-8-9-10

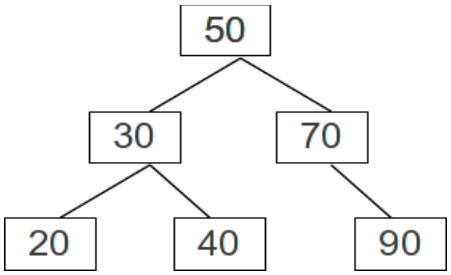
```
void posOrdem(Struct No *pNo){
    if(pNo != NULL){
        posOrdem(pNo→pEsquerda);
        posOrdem(pNo→pDireita);
        visita(pNo);
    }
}
```



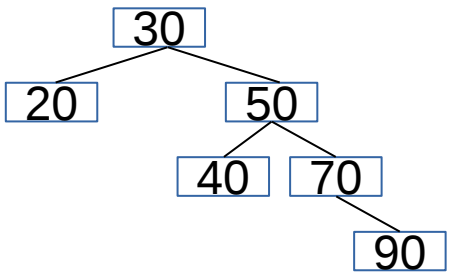
b) Considere a seguinte árvore de promoção (Splay Tree). Indique a forma da árvore após o acesso aos seguintes elementos: 50, 30, 40, 70. Deve desenhar a árvore após cada um dos acessos, ou seja faz 4 representações da árvore. (3 val)



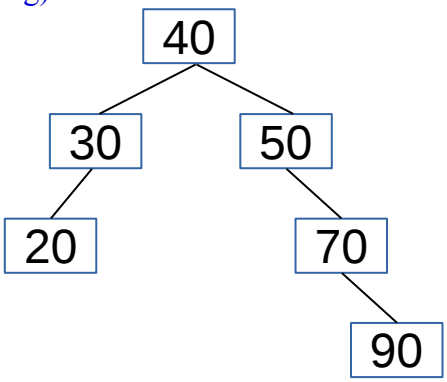
Após o acesso ao elemento 50



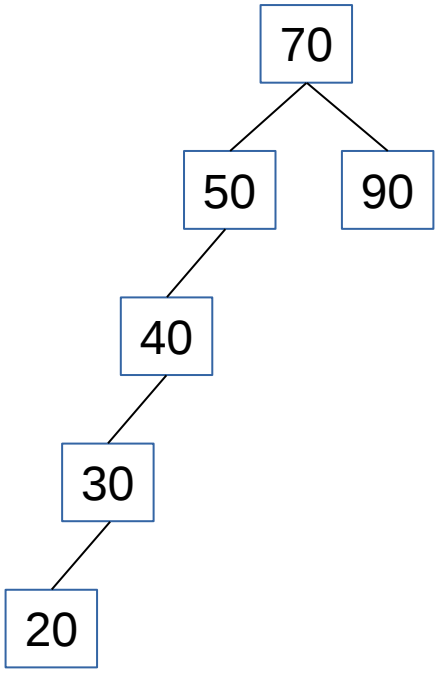
Após o acesso ao elemento 30 (Zig)



Após o acesso ao elemento 40 (Zig-Zag)



Após o acesso ao elemento 70 (Zig-Zig)



3º Questão (5 Valores)

Considere uma árvore B^+ de ordem 3 e capacidade 2.

a) O que significam os termos **ordem** e **capacidade** numa árvore B^+ (1 val)

A ordem é o número máximo de filhos de cada nó de uma árvore. A capacidade é o número máximo de elementos armazenados numa folha

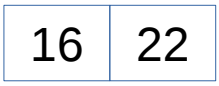
b) Indique quais as situações em que o uso de árvores B é bastante recomendado. (1 val)

O principal objetivo é minimizar os acessos a memória secundária aproveitando toda a informação dos blocos.

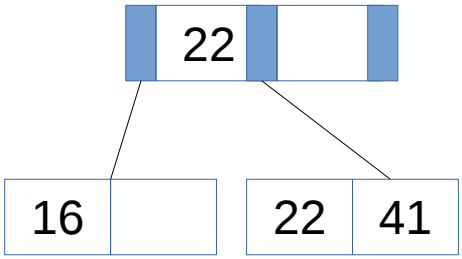
c) Desenhe a árvore após as seguintes operações: Inserção 22, Inserção 16, Inserção 41, Inserção 88, Inserção 8, Inserção 41, Remoção 8. Para cada operação represente a forma da árvore. (3 val)

Inserção 22

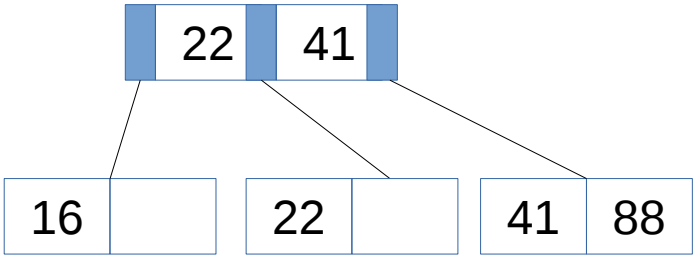
Inserção 16



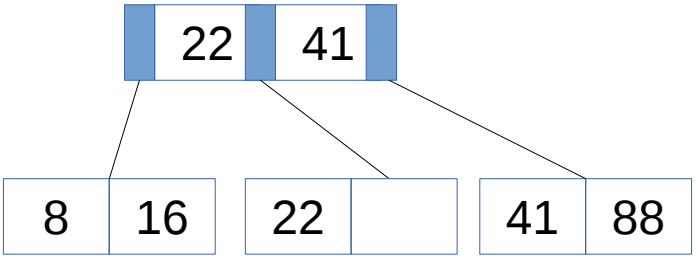
Inserção 41



Inserção 88

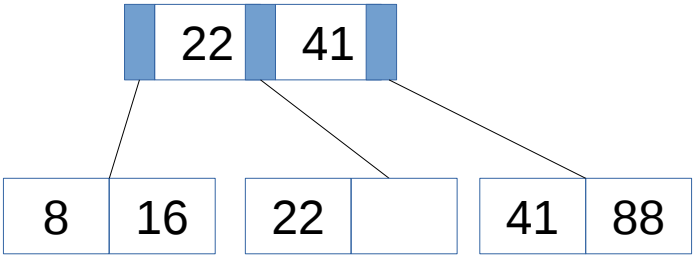


Inserção 8

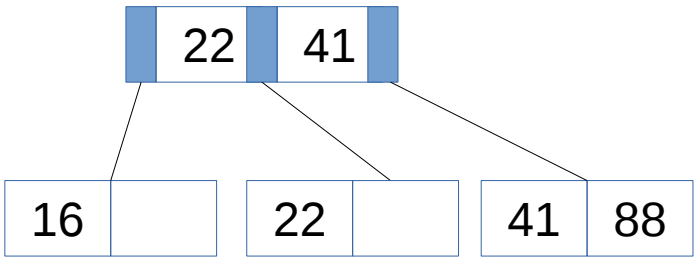


Inserção 41

(já existe o 41)



Remoção 8



4º Questão (5 Valores)

Relativamente aos tipos de dados **Pilha** e **Fila** de inteiros definidos respetivamente pelas seguintes operações. Onde **in**, **inout**, e **out** indicam se o parâmetro é de entrada, entrada-saída e saída respetivamente.

CriaPilha(**out** P:Pilha);
Evazia(**in** P: Pilha) : boolean;
Topo(**in** P: Pilha, **out** N: int);
Push(**inout** P: Pilha, **in** N: int);
Pop(**inout** P: Pilha);

CriaFila(**out** F: Fila);
Evazia(**in** F: Fila): boolean
Insere(**inout** F: Fila, **in** N: int);
Remove(**inout** F: Fila, **out** N: int);

a) Implemente em pseudo-código o tipo de dados Fila com 2 pilhas, da forma mais eficiente possível. Por outras palavras, codifique as 4 primitivas sobre filas, assumindo que dispõe apenas de 2 pilhas (P1 e P2) e que só pode recorrer às 5 operações sobre pilhas acima mencionadas. Indique a complexidade temporal dos seus algoritmos, no pior caso. (5 val)

As pilhas possuem disciplina LIFO e as filas disciplina FIFO. A pilha P2 terá os elementos em ordem FIFO. A pilha P1 é utilizada como auxiliar para obter em P2 a ordem FIFO.

```
CriaFila(out F: Fila)
{
  CriaPilha(P1);
  CriaPilha(P2);
}
```

A complexidade é O(1)

```
Evazia(in F:Fila): boolean
{
  EVazia(P2);
}
```

A complexidade é O(1)

```
Insere(inout F:Fila, in N:int)
{
   while not EVazia(P2)
  {
      T = Topo(P2);
      push(P1,T);
      pop(P2);
```

```
    }
     push(P1,N);
     while not EVazia(P1)
    {
       T=Topo(P1);
       push(P2,T);
       pop(P1);
     }
}
```

A complexidade é O(N)

```
Remove(inout F:Fila, in N:int)
{
    N=Topo(P2);
   pop(P2)
}
```

A complexidade é O(1)

FIM