| ProfessorlogomarcaFACCATcorreta.jpg | |
| --- | --- |
| Rubrica | Nota |



NOME DO ACADÊMICO: Allan Kevin Scain CÓDIGO: 1191031

(Legível) 

DISCIPLINA: Estrutura de dados 2021/1

PROFESSOR: Luiz Rodrigo Jardim da Silva

CURSO: Sistemas de Informação/Sistemas para Internet/Jogos Digitais

**DATA: 28/06/2021**

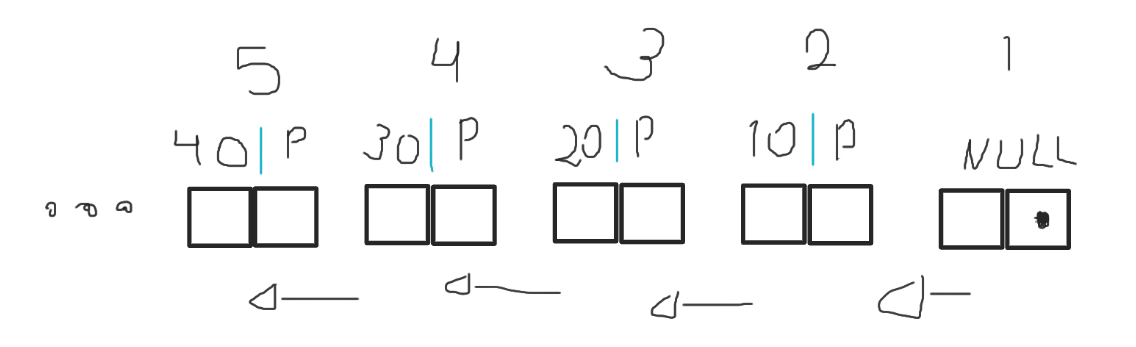


**Responda as questões abaixo (total 10 pontos).**

1. **(1,0)**  Uma lista encadeada é uma representação de uma sequência de objetos, todos do mesmo tipo, na memória RAM (*random access memory*) do computador. Cada elemento da sequência é armazenado em uma célula da lista: o primeiro elemento na primeira célula, o segundo na segunda, e assim por diante. Supondo que a lista só pode ser armazenada com números inteiros, simule a inclusão de 4 números. Essa simulação pode ser representada com figuras, passo a passo. Observação: a inserção deve ser realizada no início da lista;

**Quatro valores serão adicionados na lista encadeada V={10, 20, 30, 40 }**

**a inserção acontece assim:**

****

**STEP 1 começamos colocando o null, este é o primeiro “valor” da lista, indica o final dela**

**STEP 2 então alocamos o 10, este é o segundo “valor” da lista, antes encontramos o NULL.**

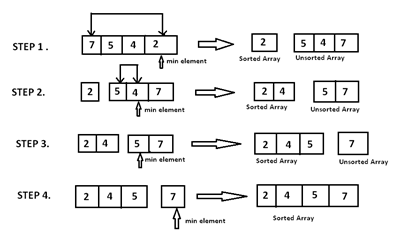
**STEP 3 então alocamos o 20, este é o terceiro “valor”, o antecessor é o 10.**

**STEP 4 então alocamos o 30, este é o quarto “valor”, antes vem o 20.**

**STEP 5 então alocamos o 40, este é o quinto “valor”, antes vem o 30.**

**Ok 0,4 - No caso deveria ser inserido no inicio da lista. Também, indicar qual o primeiro elemento.**

1. **(1,0)** Ordenação de dados tem por objetivo organizar chaves de modo a mantê-las sequencialmente arranjadas, obedecendo uma regra, que pode ser ordem numérica ou alfabética. Dentro do contexto de ordenação de dados, existem inúmeros algoritmos que são amplamente estudados e utilizados, sendo um deles representado na figura abaixo.



Pode-se afirmar que a figura acima representa qual algoritmo de ordenação? Escolha a alternativa correta:

a) BubbleSort; nao é

b) SelectionSort; não é

c) MergeSort; nao é

d) QuickSort; ESTA É A ALTERNATIVA CORRETA

e) InsertionSort; não é

Ok, 0,0 letra b

1. **(1,5)** Faça um *Ranking* dos algoritmos de ordenação estudados, de acordo com sua performance. Destaque qual algoritmo em média apresenta melhores resultados, conforme a literatura estudada, e qual o fator chave do seu sucesso. O *Ranking* pode ser criado preenchendo a tabela abaixo. Não esqueça de deixar claro a ordem do melhor para o menor algoritmo. Use [este link](http://math.hws.edu/eck/jsdemo/sortlab.html) para auxiliar na investigação.

A ORDEM ESTÁ DO MELHOR(QUICKSORT) PARA O PIOR(BUBLLESORT)

| **Algoritmo** | **Cópias** | **Comparações** |
| --- | --- | --- |
| quicksort | 44 | 63 |
| insertsort | 82 | 65 |
| selectionsort | 45 | 120 |
| mergesort | 128 | 47 |
| bubblesort | 201 | 120 |

O quick sort se destaca porque este aplica uma estratégia que se chama “divisão e conquista”.A ideia é dividir o problema de ordenar um conjunto com x valores em dois problemas menores. Os problemas menores são ordenados independentemente e os resultados são combinados para produzir a solução final.

O insert sort é considerado um método estável de ordenação, consiste em cada passo a partir do segundo elemento, selecionar o próximo item da sequência e colocá-lo no local apropriado de acordo com o critério de ordenação.

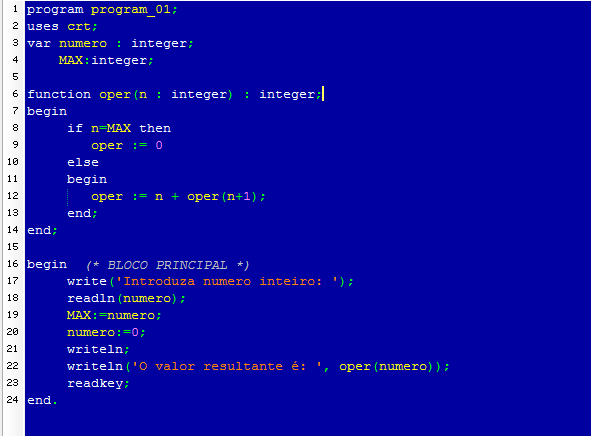
(Expliquei os dois melhores métodos , não sei se devo explicar os outros.)

Ok 1,5

1. **(1,5)** Identifique que tipo de algoritmo é executado no trecho de código representado na figura abaixo. Análise linha a linha e diga o que ele faz (você pode copiar e executar no Pascal Zim).

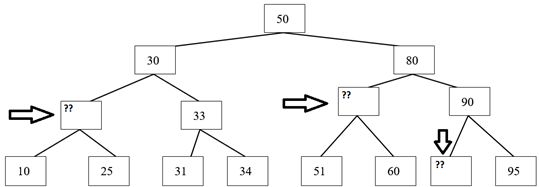
O programa implementado faz soma de 0 até X, a menos que X=0.

o usuário digitará um valor e o algoritmo pegará este, adicionado-o em MAX e dirá que a var NÚMERO é zero. Após testará, NÚMERO é igual a MAX? se for, retorna para a função. Se não for retornará o NÚMERO mais a própria função somando o NUMERO + 1, esta última parte(NUMERO + 1) fará com que o algoritmo entenda quantas vezes precisa somar N.



Ok. 0,9 - Faltou indicar que era um algoritmo recursivo

1. **(1,0)** Preencha os valores faltantes (??) da árvore de pesquisa binária abaixo:



subárvore da esquerda vai receber o 24

subárvore da direita vai receber o 55 e 81

Ok 1,0

**6) (1,0)** Tome como referência a árvore da questão 5 e responda:

a) Quantos níveis essa árvore possui?

a árvore possui 3 níveis {0, 1, 2, 3}

b) Qual o nível do nodo de número(s) **??** ?

o 24 tem nível 2, o 55 tem nível 2 e o 81 tem nível 3.

c) Qual a altura dessa árvore?

a altura da árvore é 3

d) Qual o grau do nodo 50?

O nodo 50 tem grau 2

e) Quais são os nodos terminais ou folhas?

Os nodos terminais são 10, 25, 31, 34, 51, 60, 81, 95.

Ok 1,0

**7) (1,5)** Quais as formas utilizadas para percorrer árvores? Escolha ao menos 2 delas e aplique na árvore da questão 5.

Existem tres formas de percorrer uma árvore, pré ordem, em ordem e pós ordem

vamos fazer as duas primeiras para usar como exemplo.

PRÉ-ORDEM:

50,30,24,10,25,33,31,34,80,55,51,60,90,81,95.

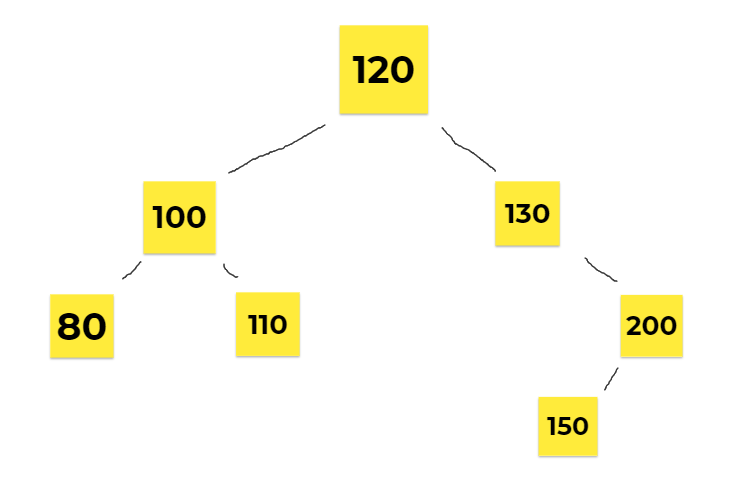
PÓS-ORDEM:

10,25,31,34,51,60,81,95,24,33,55,90,30,80,50.

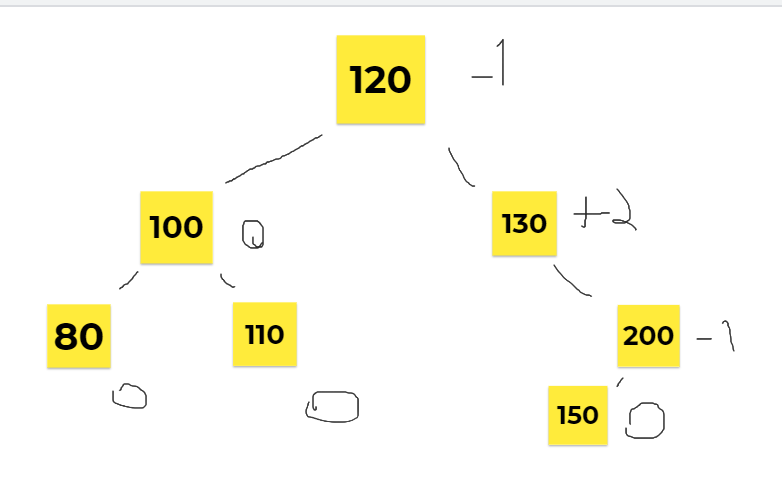
10,25,24.. SUBe, SUBd,Nó…

Ok 1,2

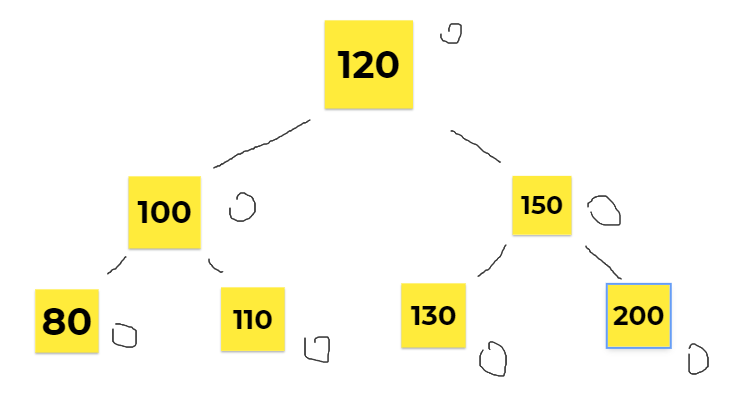
**8) (1,5)** Árvores binárias de pesquisa tendem a se degenerar para O(n) após a realização de inúmeras inserções e exclusões. O uso de técnicas de balanceamento minimizam essa degeneração. Para tanto, construa a árvore T{120,100,130,80,110,200,150} verifique se ela é uma árvore AVL. Caso não seja, efetue as devidas rotações para torná-la devidamente balanceada.



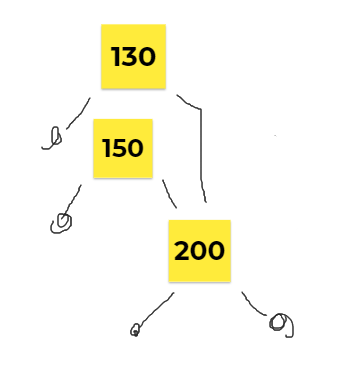
calculando o balanceamento, ficou assim:

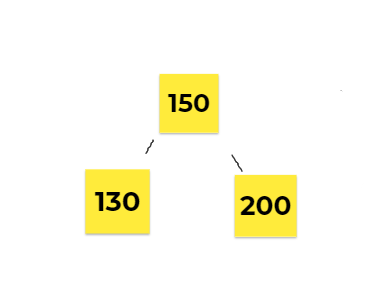


balanceando, ficou assim:



o que aconteceu foi uma rotação supla a esquerda, em 130, 200, 150:





Ok 1,5