# GABRIEL FELIPE PAIVA PEREIRA – ATIVIDADE 02 – BUSCA ÓTIMA

```
 public static void buscaOtima() {
      Scanner sc = new Scanner(System.in);
      System.out.println("Digite o número de chaves a serem lidas");
      int n = sc.nextInt();
      No[] nos = new No[n];
      for (int i = 0; i < n; i++) {
         //System.out.println("Digite a chave "+i);
         int valor = sc.nextInt();
         nos[i] = new No(valor);
         nos[i].freq = sc.nextInt();
      }
      ArvoreBusca abk = new ArvoreBusca();
      abk.permute(abk, nos, 0, n);
      abk.achaMelhor();
}
 public void permute(ArvoreBusca ab,No[] v, int start, int n) {
      if (start == n - 1) {
         print(v, n,ab);
      } else {
         for (int i = start; i < n; i++) {
            No tmp = v[i];
            v[i] = v[start];
            v[start] = tmp;
            permute(ab,v, start + 1, n);
            v[start] = v[i];
            v[i] = tmp;
         }
      }
   }
public void print(No[] v, int size,ArvoreBusca ab) {
```

```
        No[] aux = new No[size];
        if (v != null) {
            for (int i = 0; i < size; i++) {
                aux[i]=v[i];
            }
            colocaArvore(ab, aux);
        }
    } // print
public void colocaArvore(ArvoreBusca a, No[] nc) {
        Arvore tree = new Arvore();
        tree.raiz = nc[0];
        for (int i = 0; i < nc.length - 1; i++) {
            tree.Inserir(tree.raiz, nc[i + 1].valor, nc[i + 1].freq);
            //System.out.println("entrou");
            //System.out.println(tree.raiz.direita);
        }
        Arvores.add(tree);
    }
public Arvore achaMelhor() {
        int menorCusto = 0, cst;
        Arvore ArvoreMelhor = null;
        for (int i = 0; i < Arvores.size(); i++) {
            Arvore k = Arvores.get(i);
            k.calculaCustoNos(k.raiz, k);
            k.somaCustos(k.raiz);
            if (i == 0) {
                menorCusto = k.CustoTotal;
                ArvoreMelhor = Arvores.get(i);
              //  System.out.println(menorCusto);
            } else {
                //cst = GeraCusto(Arvores.get(i), Arvores.get(i).raiz, Arvores.get(i).altura(k));
```

```
            cst =k.CustoTotal;
            if (cst < menorCusto) {
               menorCusto = cst;
               ArvoreMelhor = Arvores.get(i);
            }
          }
       }
       System.out.println("Custo :" + menorCusto);
       ArvoreMelhor.CustoTotal=menorCusto;
       System.out.println("Altura :" +((ArvoreMelhor.altura(ArvoreMelhor)+1)));
       return ArvoreMelhor;
    }
  public void calculaCustoNos(No no, Arvore tree) {
       if (no.esquerda != null) {
          calculaCustoNos(no.esquerda, tree);
       }
       if (no.direita != null) {
          calculaCustoNos(no.direita, tree);
       }
       //System.out.println(tree.altura(tree));
       //System.out.println(tree.altura(tree) - tree.alturaRecur(no) + 1);
       no.custo = no.freq * ((tree.altura(tree) - tree.alturaRecur(no)+1 ));
       //System.out.println("OLHA O CUSTO DE "+no.freq+"AE"+ no.custo);
    }
    public void somaCustos(No no) {
       //CustoTotal = 0;
       if (no != null) {
          CustoTotal = CustoTotal + no.custo;
       }
       if (no.esquerda != null) {
          somaCustos(no.esquerda);
```

```
    }
    if (no.direita != null) {
        somaCustos(no.direita);
    }
}
```