UNIVERSIDAD AUTONOMA GABRIEL RENE MORENO

FACULTAD DE INGENIERIA Y CIENCIAS DE LA COMPUTACION Y TELECOMUNICACIONES

INTELIGENCIA ARTIFICIAL

CONTENIDO: MOCHILA, LISTAS DE LISTAS.

PORCENTAJE TERMINADO: 100%

INTEGRANTES	DT	HG	HI	EVAL
Flores Veizaga Eudenia Gandira		1	1	100
Garcia Taboada Brayan Albaro		1	1	100

Fecha de presentación : Jueves, 21 de Marzo de 2024

Fecha Presentada : : Jueves, 21 de Marzo de 2024

Días de Atraso : 0

```
public class Mochila {


   public static void main(String[] args) {
      LinkedList<Integer> L1=new LinkedList();
      LinkedList<Integer> L2=new LinkedList();
      LinkedList<LinkedList<Integer>> L3=new LinkedList();
        L1.add(6);
        L1.add(7);
        L1.add(5);
        L1.add(3);


      mochila(L1, L2, L3, 10, 0);  // max = 10, primera_posicion = 0
      System.out.println(L3); // muestra todas las listas soluciones
      System.out.println("Soluciones = " + L3.size());


      //consulta 1: Mostrar listas soluciones de longitud k.
```

```
        //mostrarCantidad(L3, 3); // muestra soluciones mochila con 3 objetos.


         //consulta 2: Mostrar listas de máximos objetos.
         mostrarPesoMaximo(L3);
         //consulta 3: Mostrar la cantidad de combinaciones maximas con la que se llena la
mochila...
        combinacionesMaximas(L3);
        //consulta 4: Mostrar cantidad maxima de objetos en una mochila
        CantidadObjMaximos(L3);
        //consulta 5: CAntidad maxima a llevar con el menor peso...
        minimoPesoMaxObj(L3);


    }


    private static void mochila(LinkedList<Integer> L1, LinkedList<Integer> L2, int max, int i) {
    int sum = suma(L2);
    if(sum>max)return;
        System.out.println(L2);
        int j=i;
        while(j<L1.size()){
            L2.add(L1.get(j));
            mochila(L1,L2,max,j+1);
            L2.remove();
            j=j+1;
        }


    }


    private static int suma(LinkedList<Integer> L2) {
        int sum=0;
        for (Integer integer : L2) {
```

```
        sum+=integer;
    }
    return sum;
}

public static void mochila(LinkedList<Integer> L1,
        LinkedList<Integer> L2,
            LinkedList<LinkedList<Integer>> L3, int max, int i){
    int sum = suma(L2);
    if (sum>max) return;
    L3.add(new LinkedList(L2));
    int j=i;
    while (j<L1.size()){
        L2.add(L1.get(j));
        mochila(L1,L2,L3,max,j+1);
        L2.removeLast();
        j=j+1;
    }
}
public static int getCantidad(LinkedList<LinkedList<Integer>> L1,int k){
    int i=0;
    int sum=0;
    while(i<L1.size()){
        if (L1.get(i).size()==k){
            sum++;
        }
        i++;
    }
    return sum;
}
```

```
public static void mostrarCantidad(LinkedList<LinkedList<Integer>> L1,int k){
   int i=0;
   while(i<L1.size()){
      if (L1.get(i).size()==k){
         System.out.println(L1.get(i));
      }
      i++;
   }
}

public static void mostrarPesoMaximo(LinkedList<LinkedList<Integer>> L1){
   int i=1;
   LinkedList<Integer> mayor = L1.getFirst();
   while(i<L1.size()){
      if (suma(L1.get(i))>suma(mayor)){
         mayor = L1.get(i);
      }
      i++;
   }

   System.out.println("Peso maximo soportado "+ mayor +" = "+suma(mayor));
}
  public static void combinacionesMaximas(LinkedList<LinkedList<Integer>> L1){
   int i=1;
   LinkedList<Integer> mayor = L1.getFirst();
   while(i<L1.size()){
      if (L1.get(i).size() > mayor.size()){
         mayor = L1.get(i);
      }
      i++;
   }
```

```
    System.out.println("Cantidad de conbinaciones maximas: "+getCantidad(L1,mayor.size()));
}

  public static void CantidadObjMaximos(LinkedList<LinkedList<Integer>> L1){
    int i=1;
    LinkedList<Integer> mayor = L1.getFirst();
    while(i<L1.size()){
        if (L1.get(i).size() > mayor.size()){
            mayor = L1.get(i);
        }
        i++;
    }

    System.out.println("Cantidad maxima de objetos soportados: "+mayor.size());
}

public static void minimoPesoMaxObj(LinkedList<LinkedList<Integer>> L1){
    int i=1;
    LinkedList<Integer> mayorLong = L1.getFirst();
    while(i<L1.size()){
        if (L1.get(i).size() > mayorLong.size()){
            mayorLong = L1.get(i);
        }else if (L1.get(i).size() == mayorLong.size()){
           if(suma(mayorLong) >suma(L1.get(i)) ){
            mayorLong=L1.get(i);
        }
        }

        i++;
    }
```

```
        System.out.println("Menor peso con Maximos objetos: "+mayorLong);
    }
}
```