	**UNIVERSIDADE FEDERAL DO PARÁ** **FACULDADE DE ENGENHARIA DA COMPUTAÇÃO E TELECOMUNICAÇÕES** **DATA: 30 / 08 / 2021**	**Lista 03**

Alunos: Caio Bernardo Brasil – 202006840008
Daniel Cordeiro Campos - 202006840045
Gabriel Silva Ribeiro – 202007040021
Heitor Mesquita Anglada – 202006840018

1- Na ciência da computação, um tipo abstrato de dado (TAD) é um modelo matemático para tipos de dados. Um tipo abstrato de dado é definido pelo seu comportamento (semântico) do ponto de vista de um usuário, do dado, especificamente em termos de valores possíveis, operações possíveis no dado desse tipo, e o comportamento dessas operações. Esse modelo matemático contrasta com estrutura de dados, que são representações concretas de dado, e são o ponto de vista de um implementador, não de um usuário.

2- A diferença crucial entre eles é que a estrutura de dados linear organiza os dados em uma sequência e segue algum tipo de ordem. Considerando que, a estrutura de dados não lineares não organiza os dados de maneira sequencial.

3-
```
import java.util.Scanner;

class ARRANJO{

    Scanner Inserir = new Scanner(System.in);

    String[] H = new String[10];

    String[] Transloc = new String[10];

    int aux = 0;

    String perc;

    int one;


    //Adiciona dado na primeira posição desocupada

     void LPosition(String acres){
```

```
        for(aux = 0; aux <10; aux++){
         if(this.H[aux] == null){
            this.H[aux] = acres;
            one = aux
            break;
         }
           else if(this.H[9] != null){
            System.out.println("erro");
            break;}
      }
  }
```

//Adiciona dado na primeira posição do vetor, "empurrando" os outros dados para as seguintes posições

```
    void FPosition(String acres){
      Transloc[0] = acres;
      for(aux = 0; aux <10; aux++){
         if(aux + 1 < 10)
           Transloc[aux + 1] = this.H[aux];
         else
            break;
      }
      for(aux = 0; aux <10; aux++){
        this.H[aux] = Transloc[aux];
      }
    }
```

//Adiciona dado na posição desejada do vetor, "empurrando" os outros dados para as seguintes posições

```
    void IPosition(String acres, int position){
      Transloc[position] = acres;
      for(aux = 1; position + aux < 10; aux++){
         if(aux + position < 10)
           Transloc[aux + position] = this.H[aux + position - 1];
```

```
        else
            break;
    }
       for(aux = 0; position + aux <10; aux++)
        this.H[aux + position] = Transloc[aux + position];
  }
  //Remove o ultimo dado, retornando seu valor
  String LRemove(){
        perc = this.H[9];
        this.H[9] = null;
        return perc;
   }
 //Remove o dado da primeira posição, retornando seu valor
  String FRemove(){
     perc = this.H[0];
     for(aux = 9; aux > 0; aux--)
        Transloc[aux - 1] = this.H[aux];

     for(aux = 0; aux <10; aux++)
        this.H[aux] = Transloc[aux];
     return perc;
  }
  //Remove o dado da posição desejada, retornando seu valor
  String IRemove(int position){

    perc = this.H[position];
    for(aux = 9; aux > position; aux--)
        Transloc[aux - 1] = this.H[aux];
    for(aux = 9; aux >= position; aux--)
        this.H[aux] = Transloc[aux];

    return perc;
```

```
    }
}
```

//lugar de testes dos metodos

```
public class Trabalho{
    public static void main(String[] arg){
     int c = 0;
     String Test;
     String[] A = new String[10];
     ARRANJO i = new ARRANJO();
     A[0] = "1";
     A[1] = "2";
     A[2] = "3";
     A[3] = "4";

     A[5] = "6";
     A[6] = "7";
     A[7] = "8";
     A[8] = "9";
  for(c = 0; c < 10; c++){
          i.H[c] = A[c];
     }
     Test = i.FRemove();

    for(c = 0; c < 10; c++){
        A[c] = i.H[c];
        System.out.printf("%s ", A[c]);
     }
   }
}
```

4-

A- Descobrir o tamanho da lista para só então substituir o elemento null na posição mais avançada.

B- Somar 1 a todas as posições dos elementos do arranjo e substituir a nova posição [0] pelo elemento desejado.

C- Somar 1 a posição desejada e todas as posições subsequentes a posição onde o elemento, afim de fazer com que fique um espaço livre na posição desejada para então adicionar o elemento.

D- Descobrir o tamanho do arranjo e retornar o último elemento do mesmo.

E- Subtrair 1 de todas as posições a partir da segunda para que a 2ª posição substitua a 1ª e a 3ª tome o lugar da 2ª, assim sucessivamente, fazendo com que o primeiro elemento suma e todos os demais recuem uma posição.

F- Remover o elemento da posição desejada do arranjo e subtrair 1 das posições dos elementos subsequentes a fim de manter a ordem da lista.

5- public class Arranjo{

```
    public static void main(String[]){
      int[] num=new int[5]
        num[0]=50
        num[1]=30
        num[2]=15
        num[3]=20
        num[4]=10

          System.out.println("Elemento 3 do arranjo:” +num[4]);
       }
    }
}
```

- A complexidade é que temos que definir que parâmetro deve ser usado para conseguir ter o resultado desejado.