

# APS Estrutura de Dados

Laura Moreira - 21331277  
Leonardo Leonardi - 21076739

# Objetivo

Criamos duas filas, uma comum e outra preferencial, com números aleatórios, de 1 a 60 funcionando como senha para os clientes do banco. Para organizar o atendimento criamos um contador que dentro de um laço atende os clientes da fila preferencial até o contador ser igual a 3 após isso ele atende 1 cliente da fila comum, e volta para o laço da fila preferencial. Quando a fila preferencial termina de atender todos os clientes a fila comum passa a ser atendida normalmente até o seu fim.

# Criação do TAD

```
1 public class TadFila{
2
3     //Atributo
4     private int [] dados;
5     private int tamVetor;
6     private int posFinal;
7
8     //Construtor
9     public TadFila(int tamanhoMax){
10         this.dados = new int[tamanhoMax];
11         this.tamVetor = tamanhoMax;
12         this.posFinal = -1;
13     }
```

O TAD foi criado conforme foi ensinado pelo professor nas aulas.

Criamos um vetor para armazenar a fila.

# Criação dos Métodos

```
1 //Metodo
2 public void destroyFila(){
3     this.dados = null;
4     System.gc();
5 }
6
7 public boolean isFull(){
8     if(this.posFinal == this.tamVetor -1){
9         return true;
10    }else{
11        return false;
12    }
13 }
14
15 public boolean isEmpty(){
16     if(this.posFinal == -1){
17         return true;
18    }else{
19        return false;
20    }
21 }
```

```
1 public void Enqueue(int valor){
2     if(isFull()){
3         System.out.println("Não inseriu.");
4     }else{
5         this.dados[this.posFinal + 1] = valor;
6         this.posFinal++;
7     }
8 }
9
10 public int Dequeue(){
11     if(isEmpty()){
12         System.out.println("Não remove");
13         return 0;
14     }else{
15         int elemento = this.dados[0];
16         for(int i = 0; i < this.posFinal; i++){
17             this.dados[i] = this.dados[i+1];
18         }
19         this.posFinal--;
20         return elemento;
21     }
22 }
23
24 public void imprimeFila(){
25     if(isEmpty()){
26         System.out.println("Vazia!");
27     }else{
28         for(int i = 0; i <= this.posFinal; i++){
29             System.out.println("Elemento[ " + i + " ] = " + this.dados[i]);
30         }
31     }
32 }
33 }
```

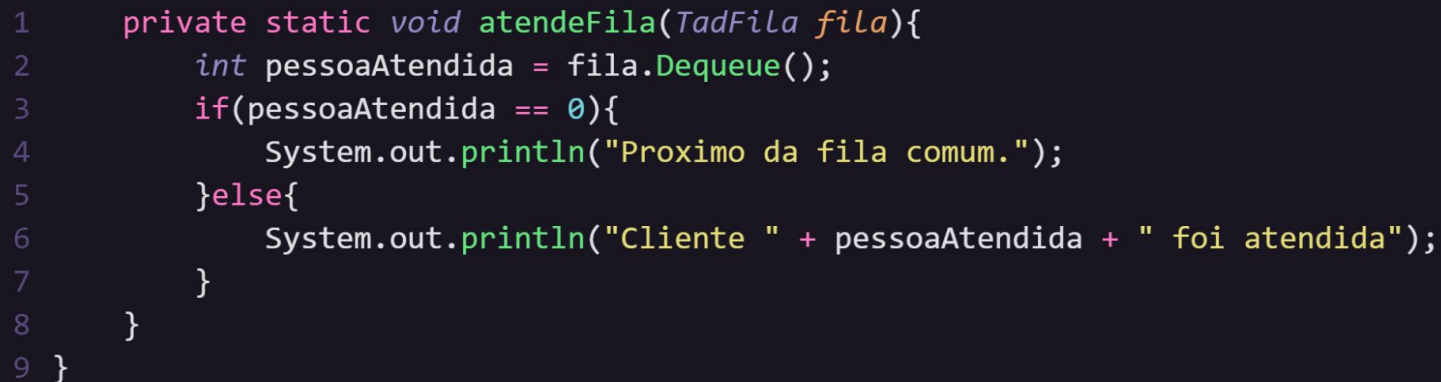
# Gerando Números Aleatórios

```
1  for(int i = 0; i<num.length; i++){
2      numeros = gerador.nextInt(60) + 1;
3      for(int j=0; j<num.length; j++){
4          if(numeros == num[j] && j != i){
5              numeros = gerador.nextInt(60) + 1;
6          }else{
7              num[i] = numeros;
8          }
9      }
10 }
11
12 for(int i = 0; i < num.length; i++){
13     fila_Comum.Enqueue(num[i]);
14 }
15
16 for(int i = 0; i<num.length; i++){
17     numeros = gerador.nextInt(60) + 1;
18     for(int j=0; j<num.length; j++){
19         if(numeros == num[j] && j != i){
20             numeros = gerador.nextInt(60) + 1;
21         }else{
22             num[i] = numeros;
23         }
24     }
25 }
26
27 for(int i = 0; i < num.length; i++){
28     fila_Prioridade.Enqueue(num[i]);
29 }
```

# Ordenação do Atendimento

```
1 while(!fila_Comum.isEmpty() || !fila_Prioridade.isEmpty()){
2     int contador = 0;
3     while(!fila_Comum.isEmpty() && contador < prioridade){
4         atendeFila(fila_Prioridade);
5         contador++;
6     }
7
8     if(!fila_Comum.isEmpty()){
9         atendeFila(fila_Comum);
10    }
11
12    if(fila_Prioridade.isEmpty()){
13        while(!fila_Comum.isEmpty()){
14            atendeFila(fila_Comum);
15        }
16    }
17 }
18 }
```

# Finalização do Código



```
1  private static void atendeFila(TadFila fila){  
2      int pessoaAtendida = fila.Dequeue();  
3      if(pessoaAtendida == 0){  
4          System.out.println("Proximo da fila comum.");  
5      }else{  
6          System.out.println("Cliente " + pessoaAtendida + " foi atendida");  
7      }  
8  }  
9 }
```

Finalização da Classe do Main quando o método atendeFila é chamado os elementos são retirados da fila utilizada e é impressa até acabarem os elementos.