

An abstract geometric design on the left side of the slide. It features a dark blue background with a white circle at the top left. Below the circle, there are concentric circles in a lighter blue shade. To the right of the circle, there are several overlapping geometric shapes: a pink triangle, a blue square, a pink square, and a dark blue triangle. The pink shapes have a pattern of thin, parallel lines. The dark blue triangle is at the bottom right of the design.

# APRESENTAÇÃO ESTRUTURA DE DADOS 1



# ENGENHARIA DE COMPUTAÇÃO - ED1

**Feito por:**

Nicolas Kalebe

Vitor Alexandre



# APRESENTAÇÃO DO PROBLEMA BASE

# FILA DE BANCO



Nossa ideia inicial era da criação de um programa em C que facilitasse e permitisse que a espera nas filas de bancos acontecesse de forma mais ágil. Diversas notícias correm por todo país por conta da alta demanda e correria diária enfrentadas pelos brasileiros.

Alguns  
exemplos  
de  
notícias:

## Bancos têm multas ratificadas por demora em atender cliente na fila

Bradesco e Caixa devem pagar R\$ 25 mil cada; Banco do Brasil, R\$ 50 mil, pois foi alvo de duas ações

12/09/2024 15h00

Por: Marcos Vieira



## TEMPO PERDIDO

# Tempo na fila do banco acima do limite fixado em lei, por si só, não gera dano moral

■ Danilo Vital

26 de abril de 2024, 18h52

Consumidor

O simples descumprimento do limite de tempo previsto em lei municipal ou estadual para a prestação de serviços bancários não gera, por si só, dano moral presumido.

Essa é a conclusão final sobre o tema da 2ª Seção do Superior Tribunal de Justiça, que fixou



# FILA DE BANCO



Dentro do programa, definimos os Clientes do banco em 4 aspectos:

**ID:** número de identificação do Banco

**Idade:** idade do cliente

**Nome:** nome do cliente

**Preferência:** aspecto físico ou psicológico que disponibilize ao cliente acesso prioritário (gestantes, idosos e PCD – Pessoas com Deficiência)





# ETAPA 1 DO PROJETO

Estrutura de dados

# ESTRUTURA DE DADOS

Dentro do programa, definimos a estrutura dos “clientes” do banco, gerando as informações na fila: ID, nome, idade, preferência e o ponteiro para o próximo cliente.

Estrutura da fila do tipo FIFO (First In, First Out), apresentada em sala de aula. Nesse tipo de fila, o primeiro elemento a ser inserido é o primeiro a ser removido, seguindo uma ordem dinâmica.

```
5
6 // Estrutura do cliente do banco
7 typedef struct Cliente {
8     int id;
9     char nome[50];
10    int idade;
11    bool preferencia;
12    struct Cliente *proximo;
13 } Cliente;
14
15 // Estrutura da fila
16 typedef struct Fila {
17     Cliente *frente;
18     Cliente *tras;
19 } Fila;
```



# ETAPA 2 DO PROJETO

Inicialização das filas e  
Cadastro dos clientes do  
banco



# INICIALIZAÇÃO DAS FILAS

O código apresentado inicia as filas. Iniciando uma fila, ela estará sem clientes tanto na frente quanto atrás (início e fim da fila vazios)

```
20
21  int contadorID = 0; // Contador para IDs
22
23  // Função para inicializar a fila
24  void iniciaFila(Fila *fila) {
25      fila->frente = fila->tras = NULL;
26  }
27
```

# CADASTRO DOS CLIENTES

```
83 // Função para adicionar cliente à fila
84 void adicionaCliente(Fila *filaPreferencial, Fila *filaNormal) {
85     char nome[50];
86     int idade;
87
88     printf("\nDigite o nome do cliente: ");
89     scanf("%s", nome);
90     printf("Digite a idade do cliente: ");
91     scanf("%d", &idade);
92
93     Cliente *novo = (Cliente *)malloc(sizeof(Cliente));
94     if (!novo) {
95         printf("Erro ao alocar memória.\n");
96         return;
97     }
98     novo->id = ++contadorID;
99     strcpy(novo->nome, nome);
100     novo->idade = idade;
101     novo->preferencia = (idade >= 60);
102     novo->proximo = NULL;
103
104     Fila *fila = (novo->preferencia) ? filaPreferencial : filaNormal;
105
106     if (fila->tras == NULL) {
107         fila->frente = fila->tras = novo;
108     } else {
109         fila->tras->proximo = novo;
110         fila->tras = novo;
111     }
112
113     printf("\nCliente %s cadastrado com sucesso! ID: %d\n", nome, novo->id);
114     // Mensagem dada quando o processo é finalizado corretamente
115 }
```

No código ao lado, os clientes são adicionados e cadastrados no Banco. Nesse processo, dados como a **idade** e a **preferência** irão indicar para qual fila o cliente deverá ir. O ID será usado para alocação de memória e mais fácil posterior remoção da fila que foi atendida.

# CADASTRO DOS CLIENTES

No exemplo do código fornecido, é possível ver que, caso o cliente possua idade igual ou superior a 60 anos, ele se encaixa nos clientes com preferência, recebendo prioridade e indo até a fila preferencial

```
97     }  
98     novo->id = ++contadorID;  
99     strcpy(novo->nome, nome);  
100    novo->idade = idade;  
101    novo->preferencia = (idade >= 60);  
102    novo->proximo = NULL;  
103  
104    Fila *fila = (novo->preferencia) ? filaPreferencial : filaNormal;  
105
```

# ETAPA 3 DO PROJETO

Exibição da Fila

# EXIBIÇÃO DA FILA

O processo de exibição das filas é muito importante não só para transparência do sistema, mas também para analisar se os clientes estão sendo atendidos corretamente.

```
116
117 // Função para exibir a fila
118 void exibirFila(Fila *fila, const char *descricao) {
119     printf("\nFila %s:\n", descricao);
120     if (fila->frente == NULL) {
121         printf("    Nenhum cliente na fila.\n");
122         return;
123     }
124
125     Cliente *atual = fila->frente;
126     while (atual != NULL) {
127         printf("    ID: %d | Nome: %s | Idade: %d | Preferência: %s\n",
128             atual->id, atual->nome, atual->idade, atual->preferencia ? "Sim" : "Não");
129         atual = atual->proximo;
130     }
131 }
```

# ETAPA 4 DO PROJETO

Atendimento ao Cliente  
e Remoção do Cliente por ID



# ATENDIMENTO AO CLIENTE

O atendimento dos clientes é necessário para que o andamento da fila ocorra não só de forma ágil, mas também de forma correta e dando prioridade sempre àqueles que possuem preferência no sistema.

```
40 // Função para remover cliente pelo ID
41 void removerClientePorID(Fila *fila, int id) {
42     if (fila->frente == NULL) {
43         printf("\nA fila está vazia.\n");
44         return;
45     }
46
47     Cliente *anterior = NULL;
48     Cliente *atual = fila->frente;
49
50     while (atual != NULL) {
51         if (atual->id == id) {
52             if (anterior == NULL) {
53                 fila->frente = atual->proximo;
54                 if (fila->frente == NULL) {
55                     fila->tras = NULL;
56                 }
57             } else {
58                 anterior->proximo = atual->proximo;
59                 if (atual == fila->tras) {
60                     fila->tras = anterior;
61                 }
62             }
63
64             printf("\nCliente com ID %d (%s) foi removido da fila.\n", atual->id, atual->nome);
65             free(atual);
66             return;
67         }
68
69         anterior = atual;
70         atual = atual->proximo;
71     }
72
73     printf("\nCliente com ID %d não encontrado.\n", id);
74 }
```

# REMOÇÃO DO CLIENTE POR ID

Nesse caso, o cliente com preferência será sempre priorizado antes no atendimento, atendendo os clientes localizados na fila normal apenas se a prioritária estiver vazia, garantindo assim dinamicidade e facilidade de atendimento as pessoas mais necessitadas.

```
154     case 3:
155         if (filaPreferencial->frente != NULL) {
156             removerClientePorID(filaPreferencial, filaPreferencial->frente->id);
157         } else if (filaNormal->frente != NULL) {
158             removerClientePorID(filaNormal, filaNormal->frente->id);
159         } else {
160             printf("\nNenhum cliente na fila para atender.\n");
161         }
162         break;
163     case 4: {
164         int id;
165         printf("\nDigite o ID do cliente a ser removido: ");
166         scanf("%d", &id);
167         removerPorID(filaPreferencial, filaNormal, id);
168         break;
169     }
```



# ETAPA 5 DO PROJETO

Criação do Menu

# criação do menu

O menu foi criado para facilitar e permitir a visualização dos comandos do código de modo eficiente para um administrador que esteja trabalhando no Banco. A partir dele, é possível **Cadastrar Cliente**, **Exibir Fila**, **Atender Próximo** e **Remover por ID**. O comando **Sair** finaliza o programa e fecha o terminal.

```
133 // Função do menu principal
134 void menu(Fila *filaPreferencial, Fila *filaNormal) {
135     int opcao;
136     do {
137         printf("\n- - - Bem vindo ao Banco UFG - - -\n");
138         printf("1. Cadastrar cliente no banco\n");
139         printf("2. Exibir filas\n");
140         printf("3. Atender próximo cliente\n");
141         printf("4. Remover cliente por ID\n");
142         printf("0. Sair\n");
143         printf("Escolha uma opção: ");
144         scanf("%d", &opcao);
145     }
```

# criação do menu

```
146 switch (opcao) {
147     case 1:
148         adicionaCliente(filaPreferencial, filaNormal);
149         break;
150     case 2:
151         exibirFila(filaPreferencial, "Preferencial");
152         exibirFila(filaNormal, "Normal");
153         break;
154     case 3:
155         if (filaPreferencial->frente != NULL) {
156             removerClientePorID(filaPreferencial, filaPreferencial->frente->id);
157         } else if (filaNormal->frente != NULL) {
158             removerClientePorID(filaNormal, filaNormal->frente->id);
159         } else {
160             printf("\nNenhum cliente na fila para atender.\n");
161         }
162         break;
163     case 4: {
164         int id;
165         printf("\nDigite o ID do cliente a ser removido: ");
166         scanf("%d", &id);
167         removerPorID(filaPreferencial, filaNormal, id);
168         break;
169     }
170     case 0:
171         printf("\nEncerrando o sistema...\n");
172         break;
173     default:
174         printf("\nOpção inválida. Tente novamente.\n");
175 }
176 } while (opcao != 0);
```

Dentro do código, a função do **Menu** foi criada e desenvolvida por meio de um *switch case*, facilitando a alteração do código e visualização das ações.

# CONCLUSÕES FINAIS

A partir do nosso programa em C, o funcionamento das filas de bancos poderá se tornar muito mais simplificado e facilitado. Com sua implementação, a agilidade em atender os clientes com prioridade permitirá maior aceitação do público e, conseqüentemente, irá melhorar a visão da empresa.

O código do programa apresenta poucas funções e comandos, aos quais são intuitivos e facilmente manipuláveis por um administrador de banco que trabalha nesse setor