Fila de Banco

Aguarde sua senha

11:11PM

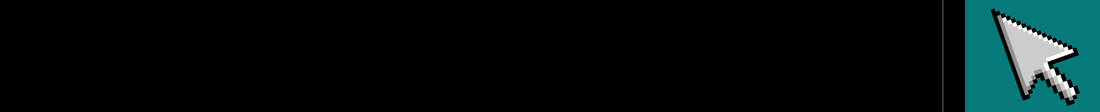
Funcionários:

- Camila Fontes Santos
- Laila Valença
- Marlysson Silva Dantas
- Miguel Ferreira França
- Vinícius Lima Santos

- Cada cliente, ao chegar, recebe uma senha que informa se ele é cliente pessoa física ou pessoa jurídica, nome do cliente e hora e minuto de chegada.
- Implemente a resolução para a seguinte situação: há dois caixas de banco ativos, um trata especificamente de atendimento de pessoa jurídica e outro de pessoa física. Cada um desses caixas possui sua própria fila. Um deles precisa sair para almoçar e as duas filas precisam se tornar somente uma. Faça um merge das filas dos dois caixas, levando em consideração a hora e minuto de chegada dos clientes.
- Considererando que há clientes que, por lei, tem direito a prioridade no atendimento. Para que os clientes que não são prioridade não fiquem indefinidamente sem atendimento, as filas devem funcionar de forma que após o atendimento de 3 clientes com prioridades, deve-se atender um cliente que não seja prioridade. Caso não haja prioridades esperando atendimento os clientes não prioridade serão atendidos normalmente.

Função para inserir ordenado

```c
int insereOrdenado(tipoLista *listaEnc, int tipocliente, char cliente[30], int prioridade, int horarioMin)
{
    tipoNo *novoNo, *aux;
    novoNo = (tipoNo *)malloc(sizeof(tipoNo));

    if (novoNo == NULL)
        return 0;

    strcpy(novoNo->nome, cliente);
    novoNo->senha = tipocliente;
    novoNo->prioridade = prioridade;
    novoNo->tempo = horarioMin;

    if (listaEnc->quant == 0 || listaEnc->inicio->tempo <= horarioMin)
    {
        return insereNaFrente(listaEnc, tipocliente, cliente, prioridade, horarioMin);
    }
    else
    {
        aux = listaEnc->inicio;

        while (aux->proxNo != NULL && aux->proxNo->tempo > horarioMin)
        {
            aux = aux->proxNo;
        }
        if (aux->proxNo != NULL)
        {

            novoNo->proxNo = aux->proxNo;
            novoNo->antNo = aux;
            aux->proxNo->antNo = novoNo;
```



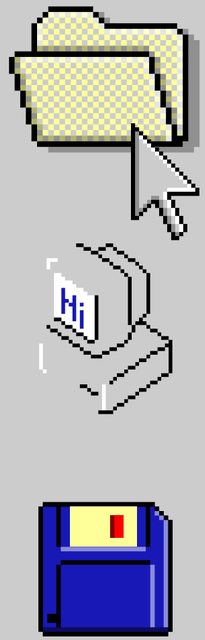
Banco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro

Função para inserir ordenado

```
            aux->proxNo->antNo = novoNo;
            aux->proxNo = novoNo;
        }
        else
        {
            aux->proxNo = novoNo;
            novoNo->antNo = aux;
            novoNo->proxNo = NULL;
            listaEnc->fim = novoNo;
        }
    }

    listaEnc->quant++;
    return 1;
}
```

Função para atender cliente

```c
int atendeCliente(tipoLista *listaEnc, int *contaPrioridade)
{
    if (listaEnc->inicio == NULL)
        return 0;
    tipoNo *atual = listaEnc->fim;

    if (*contaPrioridade < 3) // atendera 3 com prioridade e alterna para sem prioridade
    {
        if (listaEnc->fim->prioridade == 1)
        {
            *contaPrioridade = *contaPrioridade + 1;
            return removerFim(listaEnc);
        }
        else
        {
            while (atual != NULL && atual->prioridade != 1)
            {
                atual = atual->antNo;
            }
            if (atual == NULL)
            {
                *contaPrioridade = 3;
            }
            else if (listaEnc->inicio == atual)
            {
                *contaPrioridade = *contaPrioridade + 1;
                return removerInicio(listaEnc);
            }
```



Função para atender cliente

```
        else
        {
            atual->antNo->proxNo = atual->proxNo;
            atual->proxNo->antNo = atual->antNo;
            free(atual);
            listaEnc->quant--;
            *contaPrioridade = *contaPrioridade + 1;

            return 1;
        }
    }
}
if (*contaPrioridade == 3)
{
    if (listaEnc->fim->prioridade == 0)
    {
        *contaPrioridade = 0;

        return removerFim(listaEnc);
    }
    else
    {
        while (atual != NULL && atual->prioridade != 0)
        {
            atual = atual->antNo;
        }
        if (atual == NULL)
            *contaPrioridade = *contaPrioridade - 1;
```

```
    else if (listaEnc->inicio == atual)
    {
        *contaPrioridade = 0;
        return removerInicio(listaEnc);
    }
    else
    {
        atual->antNo->proxNo = atual->proxNo;
        atual->proxNo->antNo = atual->antNo;
        free(atual);
        listaEnc->quant--;
        *contaPrioridade = 0;
    }
}
```

Função para unificar as filas

```
tipoLista unificar(tipoLista *lista1, tipoLista *lista2)
{
    tipoLista *listaResultante = (tipoLista *)malloc(sizeof(tipoLista));
    listaResultante->inicio = NULL;
    listaResultante->fim = NULL;
    listaResultante->quant = 0;

    tipoNo *aux1 = lista1->inicio;
    tipoNo *aux2 = lista2->inicio;

    while (aux1 != NULL && aux2 != NULL)
    {
        if (aux1->senha <= aux2->senha)
        {
            insereOrdenado(listaResultante, aux1->senha, aux1->nome, aux1->prioridade, aux1->tempo);
            aux1 = aux1->proxNo;
        }
        else
        {
            insereOrdenado(listaResultante, aux2->senha, aux2->nome, aux2->prioridade, aux2->tempo);
            aux2 = aux2->proxNo;
        }
    }

    while (aux1 != NULL)
    {
        insereOrdenado(listaResultante, aux1->senha, aux1->nome, aux1->prioridade, aux1->tempo);
        aux1 = aux1->proxNo;
    }
```

Banco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro

Função para unificar as filas

```
while (aux2 != NULL)
{
    insereOrdenado(listaResultante, aux2->senha, aux2->nome, aux2->prioridade, aux2->tempo);
    aux2 = aux2->proxNo;
}

exibeLista(listaResultante);
return *listaResultante;
```

Função para exibir previsão de atendimento

```c
int previsaoAtendimento(tipoLista *listaEnc)
{
    int atendimentosPrioritarios = 0;
    int tempoTotalAtendimento = 0;
    int tempoCliente = 0;

    tipoNo *atual; /*Variável que será usada para percorrer a fila*/
    atual = listaEnc->fim;
    // printf("--------Dados do cliente--------\n");
    while (atual != NULL)
    {

        printf("\n--------Dados do cliente--------\n");

        printf("Nome: %s", atual->nome);
        if (atual->prioridade)
            printf("Tem prioridade\n");
        else
            printf("Sem prioridade\n");
        // printf("Prioridade: %d\n", atual->prioridade);

        if (atual->senha)
            printf("Pessoa juridica\n");
        else
            printf("Pessoa fisica\n");
        // printf("Tipo de cliente: %d\n", atual->senha);
        printf("Tempo total: %d minutos\n", atual->tempo);

        atual = atual->proxNo;
    }
    printf("------------------------------\n ");
```

anco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro • Banco Programe seu futuro


Thank you!
Você foi atendido!!