ACH2023 – turmas 04 e 94 – EP2 2º Semestre de 2020

Fila Preferencial

O objetivo deste EP é gerenciar as filas de atendimento de uma loja. Os clientes dessa loja podem ter direito ao atendimento preferencial ou não. Inicialmente, o gerente da loja havia pensado em ter duas filas convencionais: uma só com clientes não preferenciais e outra apenas com clientes preferenciais, mas ele observou que, eventualmente, um cliente preferencial teria que esperar mais tempo na fila do que clientes não preferenciais (a fila preferencial poderia ficar com muita gente ou os atendimentos desta fila poderiam demorar mais do que da outra). Para evitar essa situação, o gerente pediu uma modificação na implementação do sistema que gerencia as filas de atendimento. Ele quer que existam duas filas, uma preferencial e outra geral, a fila preferencial continuará a ter apenas clientes preferenciais e funciona como uma fila tradicional: o cliente preferencial, ao chegar na fila, é inserido no final e o cliente preferencial que estiver no início da fila será o primeiro a ser atendido. A outra fila (chamada aqui de fila geral) terá um comportamento de fila também, mas **não** será exclusiva para clientes não preferenciais. Isto é, todo cliente deverá ser inserido no final desta fila (sendo preferencial ou não) e o atendimento nesta fila (assim como em qualquer outra fila convencional) é: atende-se primeiro o cliente que estiver no início da fila. Observe que quando um cliente preferencial entra “na fila” ele na verdade entra no final das duas filas, da preferencial e da geral. Isto garante que um cliente preferencial nunca será atendido depois de um cliente não preferencial que entrou “na fila” depois dele, resolvendo o problema identificado pelo gerente.

Você será responsável por implementar a solução computacional para gerenciar a entrada dos clientes na fila e a ordem de atendimento. Para isto, você deverá gerenciar uma estrutura que contém duas listas ligadas de pessoas (**estas listas não serão circulares e não possuirão nó cabeça**). Essas listas conterão informações das pessoas que entrarem nas filas (de acordo com o pedido do gerente: uma fila terá todos os clientes a serem atendidos e a outra apenas os clientes preferenciais). Na representação em memória, cada elemento da estrutura possuirá dados de uma pessoa (um *identificador* e um campo para identificar se ela tem ou não *direito ao atendimento preferencial*) e um *ponteiro* para o próximo elemento da fila. Observe que um cliente preferencial será representado, em memória, por dois elementos diferentes, um em cada fila.

Dentre as operações previstas para esta estrutura estão (em negrito estão destacadas as funções que deverão ser implementadas por você neste EP):

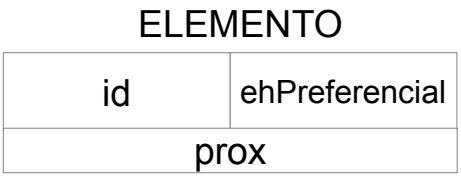
- criação da estrutura chamada fila preferencial;
- busca por uma pessoa;
- consulta à quantidade de pessoas na fila geral (tamanho da fila geral);
- consulta à quantidade de pessoas na fila preferencial (tamanho da fila preferencial);
- **inserção/entrada de uma pessoa na(s) fila(s);**
- **atendimento preferencial de uma pessoa (da primeira pessoa da fila);**
- **atendimento geral de uma pessoa (da primeira pessoa da fila);**
- **desistência/saída de uma pessoa da(s) fila(s);**

Para este EP, você deverá implementar um conjunto de funções de gerenciamento desse sistema de filas, chamado de “Fila Preferencial” utilizando principalmente os conceitos de **filas e listas ligadas não circulares e sem nó cabeça**.

A seguir são apresentadas as estruturas de dados envolvidas nesta implementação e como elas serão gerenciadas.

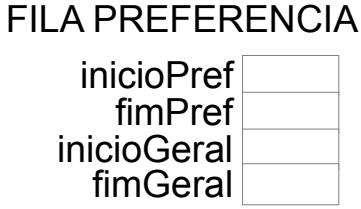
Os elementos básicos dentro de uma fila serão representados pela estrutura *ELEMENTO*, que contém três campos: *id* (identificador inteiro da pessoa), *ehPreferencial* (variável booleana para indicar se a pessoa é preferencial [valor *true*] ou não [valor *false*]), *prox* (ponteiro para o endereço do próximo elemento da fila).

```
typedef struct aux {
  int id;
  bool ehPreferencial;
  struct aux* prox;
} ELEMENTO, * PONT;
```



A estrutura *FILAPREFERENCIAL* possui quatro campos do tipo ponteiro para ELEMENTO (endereço de memória): *inicioPref, fimPref, inicioGeral e fimGeral*. A variável *inicioPref* deve apontar para a primeira pessoa da fila com direito ao atendimento preferencial (caso a fila esteja vazia, esta variável deve conter o valor *NULL*); a variável *fimPref* deve apontar para a última pessoa com direito ao atendimento preferencial (caso não haja nenhuma pessoa com direito ao atendimento preferencial, esta variável deve conter o valor *NULL*); a variável *inicioGeral* deve apontar para a primeira pessoa da fila geral (que pode ou não ser uma pessoa com direito ao atendimento preferencial), caso não haja ninguém na fila, esta variável deve conter o valor *NULL*; já a variável *fimGeral* deve apontar para a última pessoa da fila geral (caso a fila esteja vazia, esta variável deve conter o valor *NULL)*. Estas filas de elementos não serão circulares e não possuirão nó cabeça, a representação da estrutura pode ser vista a seguir:

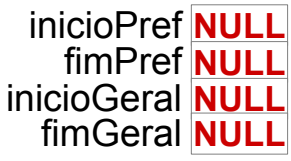
```
typedef struct {
  PONT inicioPref;
  PONT fimPref;
  PONT inicioGeral;
  PONT fimGeral;
} FILAPREFERENCIAL, * PFILA;
```



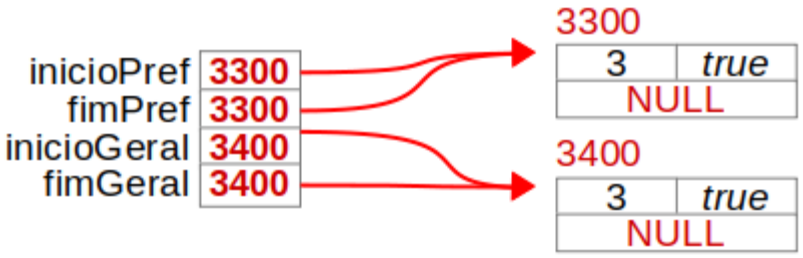
A função *criarFila* é responsável por criar uma nova estrutura FILAPREFERENCIAL, preencher os valores iniciais dos campos da estrutura FILAPREFERENCIAL e retornar o endereço da estrutura criada:

```
PFILA criarFila(){
   PFILA res = (PFILA) malloc(sizeof(FILAPREFERENCIAL));
   res->inicioPref = NULL;
   res->fimPref = NULL;
   res->inicioGeral = NULL;
   res->fimGeral = NULL;
   return res;
}
```

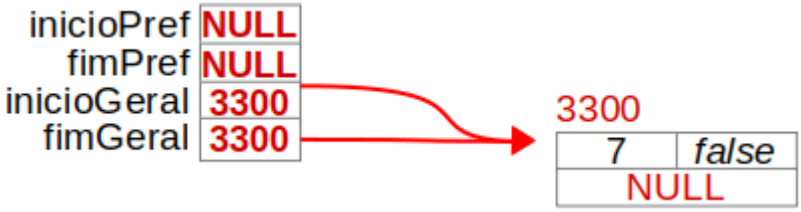
Exemplo de fila recém-criada:



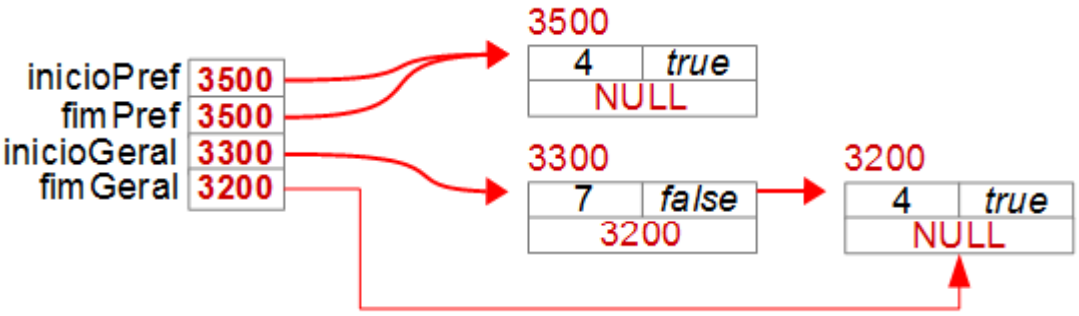
Ao se inserir **uma primeira pessoa** na estrutura, esta deverá ser inserida no início da fila geral (que estava vazia) e também deverá ser apontada pelo campo *fimGeral*. Adicionalmente, **há dois casos diferentes**:
1º) Se a pessoa tiver direito ao atendimento preferencial (isto é, o valor *ehPreferecial* é igual a *true*). Exemplo de fila após a inserção do elemento com id=3 e *ehPreferencial=true*:



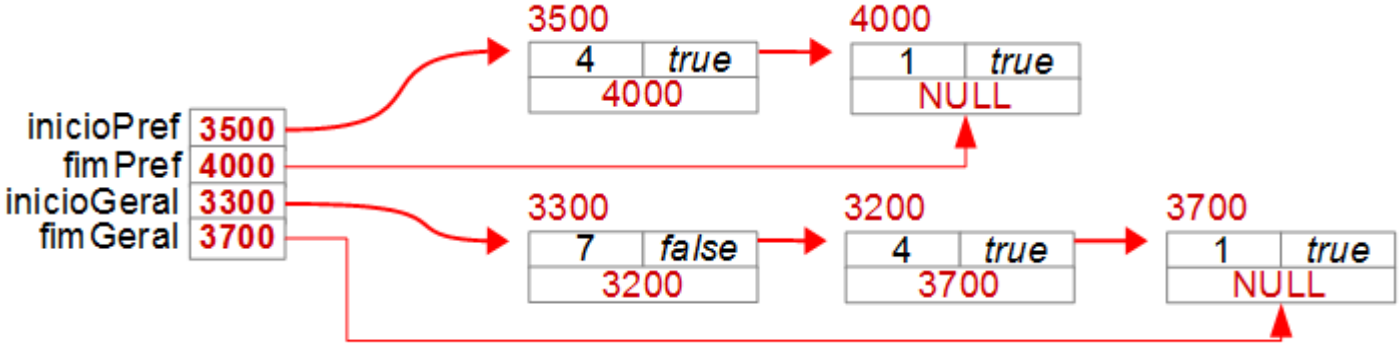
2º) Se a fila estava vazia e a pessoa não tiver direito ao atendimento preferencial (isto é, o valor *ehPreferecial* é igual a *false*). Exemplo de fila após a inserção do elemento com id=7 e *ehPreferencial=false*:



Partindo da fila do 2º exemplo (com apenas uma pessoa com id=7 e *ehPreferencial=false*), se uma nova pessoa entrar na fila com id=4 e *ehPreferencial=true* (ou seja, com direito ao atendimento preferencial), a representação em memória resultante ficará assim:



Caso uma nova pessoa com direito ao atendimento preferencial seja inserida nesta estrutura que já possui duas pessoas, por exemplo, com id=1, a representação em memória ficará assim:



Funções que deverão ser implementadas no EP

bool inserirPessoaNaFila(PFILA f, int id, bool ehPreferencial): função que recebe o endereço de uma estrutura "FILAPREFERENCIAL", o identificador de uma pessoa e se a pessoa tem ou não direito ao atendimento preferencial e retorna um valor booleano.
Esta função deverá retornar *false* caso: o identificador seja menor que zero ou caso já exista uma pessoa com o mesmo identificador na **fila geral**.
Caso contrário, a função deverá alocar memória para o novo elemento, preencher os campos *id* e *ehPreferencial* com os valores passados como parâmetro e inserir esse elemento no final da fila geral (após o último elemento dessa fila, caso a fila não esteja vazia, ou como único elemento da fila se a fila geral estava vazia antes da inserção). Adicionalmente, se a pessoa tiver direito ao atendimento preferencial (isto é, *ehPreferencial* tiver valor igual a *true*), a função deverá alocar memória para mais um elemento, preencher os campos *id* e *ehPreferencial* com os valores passados como parâmetro e inserir esse elemento no final da fila preferencial (após o último elemento dessa fila, caso a fila não esteja vazia, ou como único elemento da fila se a fila preferencial estava vazia antes da inserção). Nestes dois casos (em que foi possível realizar a inserção em uma das filas ou nas duas), a função deverá retornar *true* após realizar a inserção.

bool atenderPrimeiraDaFilaPreferencial(PFILA f, int id):* função que recebe o endereço de uma FILAPREFERENCIAL e o endereço de uma variável do tipo inteiro e retorna um valor booleano.
Esta função deverá retornar *false* se a **fila geral** estiver vazia.
Caso contrário, há duas situações. Se a fila preferencial não estiver vazia, a função deverá colocar o identificador da primeira pessoa da fila presencial na variável apontada pelo endereço armazenado em *id*. Deverá remover esta pessoa da fila preferencial e também da fila geral, acertar os ponteiros necessários para a fila não ficar inconsistente, liberar a memória dos elementos correspondentes à pessoa que foi excluída/atendida e retornar *true*. Se a fila preferencial estiver vazia, mas a fila geral não, a função deverá colocar o identificador da primeira pessoa da fila geral na variável apontada pelo endereço armazenado em *id*. Deverá também remover esta pessoa da fila geral, acertar os ponteiros necessários para a fila não ficar inconsistente, liberar a memória do elemento correspondente à pessoa que foi excluída/atendida e retornar *true*.
Notem que, após o atendimento, é possível que a fila preferencial ou mesmo as duas filas fiquem vazias.

bool atenderPrimeiraDaFilaGeral(PFILA f, int id):* função que recebe o endereço de uma FILAPREFERENCIAL e o endereço de uma variável do tipo inteiro e retorna um valor booleano.
Esta função deverá retornar *false* se a **fila geral** estiver vazia.
Caso contrário, há duas situações. Se a primeira pessoa da fila geral for uma pessoa com direito ao atendimento preferencial, a função deverá colocar o identificador da primeira pessoa da fila geral na variável apontada pelo endereço armazenado em *id*. Deverá também remover esta pessoa da fila preferencial e da fila geral, acertar os ponteiros necessários para a fila não ficar inconsistente, liberar a memória dos elementos correspondentes à pessoa que foi excluída/atendida e retornar *true*. Se a primeira pessoa da fila geral não tem direito ao atendimento preferencial, a função deverá colocar o identificador da primeira pessoa da fila geral na variável apontada pelo endereço armazenado em *id*. Deverá também remover esta pessoa da fila geral, acertar os ponteiros necessários para a fila não ficar inconsistente, liberar a memória do elemento correspondente à pessoa que foi excluída/atendida e retornar *true*.
Notem que, após o atendimento, é possível que a fila preferencial ou mesmo as duas filas fiquem vazias.

bool desistirDaFila(PFILA f, int id): função que recebe o endereço de uma FILAPREFERENCIAL e o identificador de uma pessoa e retorna um valor booleano.
Esta função deverá retornar *false* se a pessoa com identificador igual a *id* não estiver na **fila geral**.
Caso contrário, há duas situações. Se a pessoa a ser excluída possui direito ao atendimento preferencial, a função deverá excluir da fila geral e da filha presencial os elementos cujo identificador possua valor igual à *id*, acertar os ponteiros necessários para que as filas não fiquem inconsistente, liberar a memória dos elementos correspondentes à pessoa que foi excluída (que abandonou a fila) e retornar *true*. Se a pessoa a ser excluída não possui direito ao atendimento preferencial, a função deverá excluir da fila geral a pessoa cujo identificador possua valor igual à *id*, acertar os ponteiros necessários para a fila não ficar inconsistente, liberar a memória do elemento correspondente à pessoa que foi excluída (que abandonou a fila) e retornar *true*.
Notem que há diversas situações específicas que devem ser tratadas: a pessoa que abandonará a fila pode ser a única da fila geral ou da preferencial, pode ser a primeira de uma das filas (ou das duas) ou pode ser a última (de uma ou das duas filas), etc.

Informações gerais:

Os EPs desta disciplina são trabalhos individuais que devem ser submetidos pelos alunos via sistema TIDIA (ae4.tidia-ae.usp.br/) até as 23:55h (com margem de tolerância de 60 minutos).

Você receberá três arquivos para este EP:

- filapreferencial.h que contém a definição das estruturas, os *includes* necessários e o cabeçalho/assinatura das funções. Você não deverá alterar esse arquivo.
- filapreferencial.c que conterá a implementação das funções solicitadas (e funções adicionais, caso julgue necessário). Este arquivo já contém um cabeçalho, o esqueleto geral das funções e alguns códigos implementados.
- usaFilaPreferencial.c que contém alguns testes executados sobre as funções implementadas.

Você deverá submeter **apenas** o arquivo filapreferencial.c, porém renomeie este arquivo para <seuNúmeroUSP>.c (por exemplo, 12345678.c) antes de submeter.

Não altere a assinatura de nenhuma das funções e não altere as funções originalmente implementadas (*exibirLog, criarFila, etc*).

Nenhuma das funções que você implementará deverá imprimir algo. Para *debugar* o programa você pode imprimir coisas, porém, na versão a ser entregue ao professor, suas funções não deverão imprimir nada (exceto pela função *exibirLog* que já imprime algumas informações).

Você poderá criar novas funções (auxiliares), mas não deve alterar o arquivo filapreferencial.h. Adicionalmente, saiba que seu código será testado com uma versão diferente do arquivo usaFilaPreferencial.c. Suas funções serão testadas individualmente e em conjunto.

Todos os trabalhos passarão por um processo de verificação de plágios. **Em caso de plágio, todos os alunos envolvidos receberão nota zero.**