INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS – CAMPUS FORMIGA

Leandro Souza Pinheiro

TRABALHO DE ESTRUTURA DE DADOS DOCUMENTAÇÃO: SIMULAÇÃO DE UM PORTO

FORMIGA-MG

2017

1. **Execução do Sistema:**

O sistema foi projetado para ser executo no sistema operacional Linux pelo fato de utilizar alguns comandos específicos para esse S.O. Para facilitar a execução do sistema foi criado um arquivo *MakeFile,* esse arquivo é utilizado para compilar o projeto. Para compilar o sistema é necessário seguir os três passos a baixo:

1. Abrir o terminal do Linux na pasta do projeto.
2. Executar o comando *"make"* (sem aspas).
3. Executar o comando *"./exe"* (sem aspas).

2. **Descrição das bibliotecas:**

A aplicação como um todo foi dividida em basicamente em quatro bibliotecas com o intuito de gerar uma aplicação bem encapsulada e com funções modulares para que assim seja facilitado o entendimento do código e a manutenção do sistema. Para um melhor entendimento da documentação será dividido em quatro tópicos onde será explicado cada uma das bibliotecas.

3. **Pilha:**

Essa biblioteca é composta de dois arquivos um chamado Pilha.h e outro Pilha.c, no arquivo Pilha.h é descrito todas as *structs* e cabeçalhos das funções responsáveis pelo funcionamento da biblioteca, já o arquivo Pilha.c é responsável por toda a codificação da biblioteca, sendo elas:

Structs:

- Conteiner, responsável por representar os contêineres que serão inseridos nos navios em suas respectivas pilhas.
- Celula, responsável por representar cada elemento na que está na pilha, ela guarda um apontador para o elemento que está a sua frente e outro apontador para o elemento atrás, além de conter um campo para o contêiner que está inserido naquela célula da pilha.
- Pilha, responsável por representar a pilha como um todo, essa *struct* possui um apontador para célula, sendo ele o topo da pilha, além disso ela possui um campo inteiro que guardará o tamanho da pilha.

Funções:

- Pilha* cria_pilha(), essa função é considerada a mais importante da biblioteca, ela será responsável por criar a pilha alocar a mesma na memória, apontar o topo da pilha para NULL indicando que está vazia e retornar a pilha que foi criada.
- boolean destroi_pilha(Pilha *pilha), função responsável por receber por referência um tipo pilha, chamar a função responsável por remover todos os elementos da pilha e posteriormente liberar o ponteiro da pilha, vale ressaltar que essa função retorna *true* indicando que a pilha foi destruída ou *false* indicando que ocorreu algum problema ao deletar a pilha.
- boolean verifica_vazia_pilha(Pilha *pilha), essa função recebe um tipo pilha por parâmetro, e verifica o tamanho da mesma caso o tamanho

esteja com o valor zero indica que a pilha está vazia e assim ela retornará *true*, ou *false* indicando que a pilha está ou não vazia.

- void insere_pilha(Pilha *pilha,Conteiner conteiner), essa função é responsável por receber um tipo contêiner que será inserido na pilha e a respectiva pilha que conterá o contêiner, para isso ela primeiramente verifica se a pilha está vazia, se estiver vazia aloca o topo na memória e insere o elemento nessa posição e aponta o ponteiro de próximo elemento e anterior para *NULL*, indicando que não existe esses elementos ao final a função coloca o tamanho da pilha com o valor 1 indicando existir um elemento na pilha. Caso a pilha não esteja vazia a função irá alocar uma posição à frente do topo na memória, passar o ponteiro do topo para essa nova posição e apontar o ponteiro de anterior para o elemento que antes era o topo, e posteriormente apontar o próximo elemento para *NULL*, vale ressaltar que ao final da função é incrementado 1 no contador do tamanho da pilha para representar o novo elemento.
- void remove_pilha(Pilha *pilha, Conteiner *conteiner), função responsável por remover o elemento do topo da pilha, para isso ela recebe um ponteiro para receber o elemento que será removido e a pilha onde o mesmo se encontra, assim ela remove o elemento do topo caso a pilha não esteja vazia e decrementa 1 no contador indicando que aquele elemento não se encontra mais na base de dados.
- void mostra_pilha(Pilha *pilha), função responsável por listar todos os elementos contidos na pilha, para isso ela recebe por referência um ponteiro da pilha e varre a pilha do topo até o primeiro elemento, mostrando o mesmo para o usuário.

4. Fila:

Essa biblioteca é composta de dois arquivos um chamado Fila.h e outro Fila.c, no arquivo Fila.h é descrito todas as *structs* e cabeçalhos das funções responsáveis pelo funcionamento da biblioteca, já o arquivo Fila.c é responsável por toda a codificação da biblioteca, sendo elas:
Structs:

- Navio, essa *struct* será responsável por representar os navios que serão inseridos na fila, ela contém um campo inteiro para representar a quantidade de contêineres, outro para representar o *id* do navio e outro ainda para representar o tempo que está na fila, além disso o navio possui também quatro apontadores de pilha que serão responsáveis por armazenar os contêineres que esse navio possui para descarregar.
- Nodo, essa *struct* será responsável por representar todos elementos contidos na fila, para isso ela possui um apontador para o elemento a sua frente e um tipo navio que será o navio contido na determinada posição da fila.
- Fila, essa *struct* será responsável por representar a fila em questão, para isso ela possui dois apontadores, um para o primeiro elemento e outro para o último, além de possuir um campo inteiro para armazenar o tamanho da fila.

Funções:

- Fila* cria_fila() essa função é considerada a mais importante da biblioteca, ela será responsável por criar a fila alocar a mesma na memória, apontar o primeiro e último elemento da fila para NULL indicando que está vazia e retornar a pilha que foi criada, além de colocar o tamanho da fila como zero.
- boolean verifica_vazia_fila(Fila*), função responsável por receber por referência um tipo fila, chamar a função responsável por remover todos os elementos da fila e posteriormente liberar o ponteiro da pilha, vale ressaltar que essa função retorna *true* indicando que a fila foi destruída ou *false* indicando que ocorreu algum problema ao deletar a fila.
- boolean destroi_fila(Fila*), essa função recebe um tipo fila por referência, e verifica o tamanho da mesma caso o tamanho esteja com o valor zero indica que a pilha está vazia e assim ela retornará *true*, ou *false* indicando se a fila está ou não vazia.
- void insere_fila(Fila*,Navio), essa função é responsável por receber um tipo navio que será inserido na fila e a respectiva fila que conterá o navio, para isso ela primeiramente verifica se a fila está vazia, se estiver vazia aloca o primeiro e último elemento na memória e insere o elemento nessa posição e aponta o ponteiro de próximo elemento para *NULL*, indicando que não existe esses elementos ao final a função coloca o tamanho da fila com o valor 1 indicando existir um elemento na fila. Caso a fila não esteja vazia a função irá alocar uma posição à frente do último elemento na memória, passar o ponteiro do último para essa nova e posteriormente apontar o próximo elemento para *NULL*, vale ressaltar que ao final da função é incrementado 1 no contador do tamanho da fila para representar o novo elemento.
- void remove_fila(Fila*,Navio*), função responsável por remover o ultimo elemento da fila, para isso ela recebe um ponteiro para ser o elemento que será removido e a fila onde o mesmo se encontra, assim ela remove o último elemento da fila, caso a mesma não esteja vazia e decrementa 1 no contador indicando que aquele elemento não se encontra mais na base de dados.
- void mostra_fila(Fila*), função responsável por listar todos os elementos contidos na fila, para isso ela recebe por referência um ponteiro da pilha e varre a pilha do primeiro até o último elemento, mostrando o mesmo para o usuário.

5. Desenvolvimento:

Essa biblioteca é composta de dois arquivos um chamado desenvolvimento.h e outro desenvolvimento.c, no arquivo desenvolvimento.h é descrito todas as structs e cabeçalhos das funções responsáveis pelo funcionamento da biblioteca, já o arquivo desenvolvimento.c é responsável por toda a codificação da biblioteca, sendo elas:

Structs:

- Travessa, essa *struct* é responsável por representar as travessas em que serão inseridos os contêineres após serem descarregados do navio, para isso ela recebe um ponteiro para a pilha onde esses elementos serão inseridos.
- Atracamento, essa *struct* é responsável por representar os atracamentos onde as travessas se localizam, para isso a *struct* possui um vetor de 4 posições do tipo travessa, assim cada atracamento criado possuirá 4 travessas que poderão receber os contêineres dos navios, além disso a *struct* possui um campo inteiro que será um contador de vezes que o veículo de transporte do atracamento foi utilizado.

Funções:

- void salva_id_txt(int id), função responsável por receber um valor inteiro representando o *ID* do navio para salva-lo em um arquivo de texto, essa função é de suma importância pois assim os *IDs* dos navios não serão repetidos.
- int gera_id_navio(), função responsável por abrir o arquivo que contém os *IDs* salvos, e gerar um no *ID* e retornar o mesmo.
- Navio gera_navio(), função responsável por gerar um navio novo, para isso ela cria um número aleatório de contêineres que o mesmo irá possuir, além de gerar o *ID* para o navio e retornar o mesmo.
- Conteiner gera_conteiner(), função responsável por gerar um contêiner novo e retornar o mesmo.
- void pilhas_navio(Navio navio), função extremamente importante para a simulação do porto, essa função recebe um navio por parâmetro e com isso coloca os contêineres que foram gerados aleatoriamente em suas pilhas, para isso ela coloca um contêiner em cada pilha, exemplo: caso o navio possua 4 contêineres a função ira colocar um contêiner na primeira pilha, um na segunda, um na terceira e por fim um na última pilha, assim sempre todas as pilhas do obrigatoriamente conterá pelo menos um elemento.
- Navio insere_navio_fila(Fila *fila), função responsável por receber uma fila por parâmetro, e gerar um novo navio e colocar o mesmo na fila que foi recebida.
- void chama_veiculo_transporte(struct TRAVESSA travessa), função responsável por receber uma travessa que está com a pilha cheia e assim utilizar o veículo de transporte para liberar travessa, e incrementar 1 no contador indicando que o veículo foi utilizado.
- boolean verifica_navio_descarregou(Navio navio), essa função é responsável por receber um navio e verificar se o mesmo descarregou por completo, ela é utilizada para evitar que um navio seja removido da fila sem ter descarregado por completo, para isso ela retorna *true* indicando que o navio descarregou por completo ou *false* indicando que o mesmo possui contêineres para descarregar em alguma de suas pilhas.
- void descarrega_navio(struct ATRACAMENTO atracamento[], Fila *fila), considerada a principal função do sistema, ela é responsável por

receber o um vetor de atracamentos e um fila, assim ela pega o primeiro navio da fila em questão e procura por travessas que possui espaços em sua pilha e assim descarrega os contêineres nas travessas que se enquadrem. Vale ressaltar que a função chama o veículo de transporte caso seja detectado que alguma travessa está cheia.

6. **Keyboard**

Essa biblioteca é composta por um arquivo Keyboard.h e um Keyboard.c. A biblioteca possui uma única função que consiste apenas em verificar se o usuário pressionou alguma tecla, retornando *true* o usuário pressionou, ou *false* indicando que o usuário não pressionou nenhuma tecla.