Simulador-Porto

Detalhes de projeto e descrição das funções do simulador.

Definições

```C
    #define MAX_TEMPO 100
```

Define a quantidade de iterações de tempo realizadas pelo projeto.

Estruturas

- Estrutura Container

```
typedef struct {
    int id;
} Container;
```

Id para controle de alocação.

- Estrutura Navio

```
typedef struct {
    unsigned long int id;
    unsigned long int tempo;
    Lista * pilhas[4];
} Navio;
```

Id para controle de alocação; Tempo para contagem de tempo de espera; Pilhas de containers para armazenamento.

Variáveis Globais

```C
    Lista * fila_atracadouro[4];
    Lista * pilha_travessa[5];
    unsigned long int last_id = 0;
    int id_containers = 0;
    int mod_travessa = 0;
    int cont_travessa = 0;
```

```
- `fila_atracadouro[]`

Para controle das filas de navios.

- `pilha_travessa`

Para controle das travessas de containers.

- `last_id`

Módulo para alocação dos Navios nas filas, armazena o último id utilizado para alocação.

- `id_containers`

Módulo para alocação dos Containers, armazena o último id utilizado para alocação.

- `mod_travessa`

Módulo para controle das travessas para evitar desbalanceamento.
```

Funções

- Funções de alocação e liberação de memória

```
Container * alocar_container();
void liberar_container(Container * c);
Navio * alocar_navio();
void liberar_navio(Navio * nav);
void alocar_filas();
void liberar_filas();
void alocar_pilhas(Navio * nav);
void liberar_pilhas(Navio * nav);
void alocar_travessas();
void liberar_travessas();
```

- Funções de controle dos navios

```
int qtd_containers(Navio * nav);
void inserir_container(Navio * nav, Container * c);
Container * remover_container(Navio * nav);
```

1. qtd_containers(): Retorna a quantidade de containers no navio;

2. inserir_container(): Insere um container no navio;

3. remover_container(): Remove um container do navio;

- Funções de controle das filas

```
void entrar_fila(int n);
void incrementar_tempo();
void movendo_gruas();
```

1. entrar_fila(): Insere n navios na fila e printa os ids dos navios armazenados nas filas.
2. incrementar_tempo(): Adiciona +1 ao tempo armazenado nos navios.
3. movendo_gruas(): Como definido na apresentação do projeto, cada área de atracamento deve possuir uma grua para movimentar os containers dos Navios até uma das 5 pilhas de travessas; foi projetado um loop (0 - 4) que simula a ação das gruas movimentando um container até uma das pilhas. Se inserção na travessa falhar, o container é devolvido ao navio. Uma vez movido, se o navio não tiver mais containers ele é liberado, senão ele retorna para a fila.

- Funções de entrada e saída das pilhas

```
int inserir_travessa(Container * c);
void esvaziar_travessas();
```

1. inserir_travessa(); Insere um container na travessa, controla o tamanho das travessas para evitar desbalanceamento, utilizando um módulo. Retorna 1 em caso de sucesso e 0 em caso de falha.
2. esvaziar_travessas(): Verifica o tamanho de todas as 5 travessas, se houver uma travessa de tamanho maior ou igual a 5, todos os containers são liberados (movidos para o pátio).

- Funções de prints

```
void mov_grua();
void media_tempo();
```

1. mov_grua(): Printa a ação da grua, descrevendo o número da grua e o id do navio.
2. media_tempo(): Printa a média de tempo de espera dos navios nas filas de atracamento.