Trabalho Prático 1
O problema da frota intergaláctica do novo imperador

Gustavo Luiz Campolina Teles

Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brasil

gusstavotelles@ufmg.br

1. Introdução

O desafio proposto por esta atividade foi o de implementar um projeto capaz de gerenciar as naves para o Imperador Palpatine, cuidando da administração do envio destas à batalhas, e cuidado com as avarias das mesmas.

2. Implementação

O programa foi desenvolvido na linguagem C++, compilada pelo compilador G++ da GNU Compiler Collection.

2.1. Estrutura de Dados

O Projeto teve como embasamento teórico a utilização de diversas das estruturas de dados estudadas (listas, filas e pilhas) considerando cada elemento destas como a classe Cédula.

Utilizou-se pilhas encadeadas para o armazenamento das naves totalmente consertadas, ou seja, sem avarias. Neste caso a ordem de aptidão da pilha de naves obedece o conceito LIFO, tornando este uso apropriado.

Utilizou-se listas encadeadas para o armazenamento de naves enviadas para batalha, pois neste caso, o gerenciamento e monitoramento é completo e possibilita a inserção e retirada de qualquer elemento a partir de sua chave/identificador.

Utilizou-se, para armazenamento das naves avariadas pós batalha, a estrutura de fila encadeada, visto que, neste caso, à medida que chegam novas naves avariadas, estas têm prioridade no reparo, obedecendo então o conceito FIFO, tornando este uso apropriado.

As funções de Enfileirar, Desenfileirar, Empilhar, Desempilhar, as variadas Inserções e retiradas foram baseadas no material de aula disponibilizado pelo professores no moodle referente à matéria de Estrutura de Dados, sendo estas adaptadas ao contexto da tarefa.

2.2. Classes

Com fins de modularização do projeto, foram criadas diversas classes, descentralizando assim, o funcionamento do programa.

A Principal é a classe Nave, a qual armazena informações importantes do estado, sendo estas fundamentais para a execução do projeto.

Outra classe decisiva para os rumos da execução é a TipoItem, a qual possui atributos que definem a ação a ser tomada, e os parâmetros para tal (a chave e a nave em questão).

Foram também implementadas classes essenciais para o funcionamento básico das estruturas escolhidas. São estas as classes ListaEncadeada, FilaEncadeada, PilhaEncadeada, Lista, Fila e Pilha.

2.3. Funcionamento

O funcionamento baseia-se em inicialmente ocorrer um preenchimento da quantidade de frotas por parte do usuário que, em seguida, informa as respectivas informações das naves de cada frota, por fim, empilhando-as na pilha navesOk, que são as naves sem avaria, isto é, prontas para o combate.

Em seguida o usuário informa o tipo de operação a qual deseja realizar, possibilitando enviar uma nave da pilha ao combate, mostrar as naves consertadas em ordem de aptidão, mostrar as naves avariadas em ordem de prioridade, ou receber a chegada de uma nave pós batalha.

3. Análise de Complexidade

3.1. Tempo

Basicamente para empilhar a frota o custo de execução se mantem constante visto que para empilhar um elemento não é necessária uma verificação, portanto conclui-se que esta complexidade é de $O(1)$.

Para qualquer operação de gerenciamento das naves em combate a complexidade é de $O(n)$, visto que o acesso necessita de comparação da chave dada pelo usuário com o identificador de cada nave inserida nesta lista. O pior caso seria o qual a última nave enviada ao campo de batalha estivesse com maiores danos e necessitasse ser retirada. Já o melhor caso seria o que a primeira estivesse com maior avaria.

3.2. Espaço

Tendo em vista que cada nave possui a mesma demanda de espaço, em qualquer situação (avaria, pronta pro combate e em batalha) o pior caso seria o qual todos estão em uma mesma situação destas anteriormente citadas. O Melhor caso seria uma ditribuição com equidade destas situações, estando aproximadamente ⅓ das naves em combate, ⅓ em fila de conserto e ⅓ prontas para combate aguardando serem enviadas.

Esta complexidade seria de: $O(2n) = O(n)$

4. Conclusão

Com o desafio deste projeto, percebe-se onde os conhecimentos adquiridos ao longo do curso foram devidamente aplicados e suas necessidades para além das barreiras de demandas acadêmicas, ou seja, aplicações no “mundo real”.

A escolha das estrutura correta para cada possibilidade de ação com as naves foi outro fator chave, sendo esta feita com exatidão a responsável pelo funcionamento de acordo com a demanda exigida nas especificações do Trabalho.

Outra conclusão notável é a necessidade e relevância de um código limpo e REFATORADO, visto que ao longo do desenvolvimento, houve imensa diminuição do número de linhas em código devido à sua refatoração, evitando comparações desnecessárias e execuções de tamanho maior que o demandado para realizar determinadas tarefas mas mantendo a integridade das funcionalidades.

Referências

Material Didático disponibilizado no Moodle da disciplina de Estrutura de Dados.