Trabalho Prático I: Extração-Z

Lucas S. Teles

1. Introdução

Esta documentação descreve a modelagem e implementação da solução para o problema proposto no primeiro Trabalho Prático da disciplina de Estruturas de Dados, semestre 2020/2 da UFMG.

Foi implementado um módulo para a Base de robôs; um módulo para os Robôs; um módulo para a interpretação de um Mapa, de acordo com as especificações do Trabalho; por fim, um módulo para uma Fila Prioritária (com política FIFO, mas que permite a inserção de um elemento no início da fila), que utiliza o módulo da Célula de Comando, que por sua vez utiliza o módulo do Item de Comando.

Na seção 2, a implementação do problema será descrita com mais detalhes; na seção 3, instruções de compilação e execução serão apresentadas; na seção 4, será mostrada a análise de complexidade de tempo e espaço do sistema completo; na seção 5 serão feitos alguns comentários a respeito da utilização do código fonte para melhor visualização do sistema; finalmente, na seção 6, estará a conclusão da documentação com comentários sobre o percurso durante a solução do problema.

2. Implementação

Como descrito brevemente na seção anterior, o sistema consiste de 6 módulos e um arquivo main.cpp que se comunicam entre si para solucionar o problema proposto no TP: Extração-Z.

Apenas uma estrutura de dados foi implementada, a Fila Prioritária, que é utilizada pela Base, para registrar os comandos do arquivo passado. Nesse momento, a funcionalidade de prioridade da fila não é utilizada. Cada Robô também possui duas filas, uma para os comandos em espera do comando “EXECUTAR k” da Base e uma para o histórico de comandos já executados. Apenas a fila com os comandos em espera utiliza a funcionalidade de prioridade da Fila Prioritária.

A escolha de implementar apenas uma estrutura de dados, foi tomada visando reduzir a complexidade do sistema, já que o problema em si é relativamente simples.

Para a representação do mapa foi utilizada uma matriz de caracteres. Após algumas revisões, essa foi a maneira encontrada que melhor representava o problema, sem utilizar estruturas da biblioteca padrão.

A implementação da fila, seguiu às implementações das aulas quase que diretamente, além de incluir a funcionalidade de prioridade, apenas comentada em aula. A implementação de uma função furaFila() não apresentou dificuldade adicional já que a implementação da fila apresentada pelo professor permite a criação de tal função apenas utilizando um ponteiro temporário adicional e o ponteiro para a célula cabeça da fila.

A Base é muito simples tendo apenas um construtor que preenche a fila de ordens e uma função que executa todas as ordens chamando as funções necessárias dos robôs.

O módulo Robô é o maior e mais complexo, com as três funções que os robôs possuem segundo a descrição do problema, uma função que executa todos os comandos assim como na Base, uma função que imprime o relatório do Robô, vários getters e setters. As classes Robo e Base possuem uma função que converte uma entrada em string, essa função foi implementada apenas como suporte a uma possível futura escalabilidade, como aumento de funções do Robô e da Base, seguindo as diretivas de boas práticas de programação orientada a objetos.

As classes FilaPrioritaria, Base, Robo e MapReader possuem destrutores para liberar os espaços de memória alocados dinamicamente por elas.

Os arquivos seguem algum tipo de padronização na ordem dos #include (primeiro os arquivos próprios, depois as bibliotecas da linguagem), na declaração de métodos e variáveis, e implementação de métodos. As declarações e a implementação dos métodos é dividade em seções que são similares de alguma forma, além de ser indentadas a fim de facilitar a legibilidade do código. Ainda no tópico de legibilidade, em todo o sistema foram utilizadas duas variáveis para as coordenadas i,j dos robôs e mapa para facilitar a compreensão do código. Por quase todo o sistema existem comentários, alguns separando as seções similares outros para melhor entendimento do código.

Quase nenhum tratamento de erro é feito, já que duplicaria a complexidade do sistema e supõe-se que os arquivos do mapa e dos comandos respeitarão a formatação indicada na descrição do problema.

O Sistema Operacional utilizado para testar o programa foi o Windows 10, a linguagem utilizada foi C++, o compilador é o GNU GCC Compiler. O processador é um Intel Core i3-2310M e o computador possui 6 GB de RAM.

Os arquivos de header estão no formato .hpp seguindo boas práticas de programação, por mais que esse projeto é pequeno e só utilizou a linguagem C++.

Alguns comentários adicionais sobre a compilação serão encontrados na seção 5.

3. Instruções de Compilação e Execução

Para compilar o programa, basta executar o comando make no terminal no diretório TP1 (onde se encontra o arquivo Makefile), dentro do diretório base “projeto”, sem necessidade de nenhum argumento adicional à chamada do comando. Os arquivos objeto .o vão aparecer no diretório obj e o executável run.out (como pedido na descrição e no Makefile disponibilizado) será compilado no diretório bin.

O programa requer dois argumentos adicionais ao ser chamado, a localização do arquivo do mapa e do arquivo dos comandos, nessa ordem, como descrito na especificação do Trabalho.

Os arquivos DEVEM respeitar a formatação apresentada na proposta do Trabalho para funcionalidade total e correta do sistema.

4. Análise de Complexidade

Complexidade de Tempo

O método MapReader() é O(n * m) pois cria uma matriz n x m elemento a elemento, que é O(1) para cada elemento.

Os métodos limpaFila(), ~FilaPrioritaria(), Base(), ~Base, executarOrdens(), ~MapReader(), executarComandos(), imprimirRelatorio() são O(n) pois todos percorrem uma estrutura de tamanho n por completo, uma única vez, a cada vez executando comandos O(1), mesmo que sejam chamadas a outros métodos.

O método ~Robo() é O(max(n, m)), sendo n o tamanho do histórico e m o tamanho da fila de ordens pendentes.

Todos os outros métodos são O(1).

No geral programa todo é $O(n^2)$, sendo n a quantidade de comandos, esse não é um cálculo exato ou exaustivo, mas sim uma constatação que na utilização comum do sistema, existirão uma proporção de comandos aos robôs maior que ordens à base, em torno de uma média.

Complexidade de Espaço

A classe FilaPrioritaria como um todo tem complexidade O(n) para espaço porém todos os seus métodos são O(1).

As classes ItemComando e CelulaComando são O(1) no geral e em seus métodos.

A classe Base é O(n * m) por conter um mapa, seus métodos Base() e executarOrdens() são O(n) e o restante O(1).

A classe Robo é O(max(n, m)) sendo n o tamanho da fila do histórico e m o tamanho da fila de ordens pendentes, seus métodos executarComandos() e imprimirRelatorio() são O(n) e todos os outros O(1).

A classe MapReader e seu método MapReader() são O(n * m) por causa do mapa e todos os outros métodos são O(1).

Construtores e destrutores foram chamados de métodos a fim de manter a legibilidade do documento. Os argumentos dos métodos foram omitidos pela mesma razão.

5. Comentários Adicionais

O operador « foi sobrecarregado para a classe MapReader a fim de facilitar a visualização do mapa, antes e depois da execução do sistema, na saída padrão.

Surgiu um erro na compilação do programa utilizando o Code::Blocks (antes de compilar usando o Makefile, quando o erro não mais ocorreu), sobre algumas .dll faltando. O erro é muito grande para colocá-lo aqui. A solução encontrar foi utilizar a linkagem estática do compilador que deixa o executável um tanto maior (mas nada mais que 2 MB).

Existe uma inconsistência no uso das línguas Portuguesa e Inglesa no código fonte e nessa documentação. Algumas palavras não tem tradução (ou tem tradução difícil) e as que tem são pouco usadas no meio da Computação. Portanto, alguma liberdade foi tomada para escolher os melhores termos para cada variável e função a fim de descrever melhor o propósito delas.

6. Conclusão

O sistema construído para a solução do problema buscou diminuir a complexidade e a utilização de recursos de memória (e quando possível, de tempo) do computador.

A resolução desse trabalho permitiu uma maior familiarização com a linguagem e suas funcionalidades, além das suas obscuridades.

Por fim, foram feitos em torno de três sistemas completos até chegar na versão final. As outras versões utilizavam estruturas da biblioteca padrão para implementar um leitor de comandos e o mapa de MapReader. Além da não implementação do módulo de leitura de comandos, foram encontradas maneiras alternativas de modelar o problema e daí vieram as maiores dificuldades enfrentadas. A Fila e suas funcionalidades foram as primeiras a serem implementadas, logo depois o leitor de mapas e de comandos, por fim a base que mudou várias vezes e o robô que foi o mais difícil, pelas restrições impostas pela especificação do trabalho.

Mesmo com todas essas dificuldades, a versão final do sistema ficou menor e menos complexa que as modelagens anteriores, exatamente por essa razão que o trabalho está sendo entregue atrasado.

Bibliografia

Chaimowicz, L. e Prates, R. (2020). Slides virtuais da disciplina de estruturas de dados. Disponibilizado via moodle. Departamento de Ciência da Computação. Universidade Federal de Minas Gerais. Belo Horizonte.

https://stackoverflow.com/questions/20594520/what-exactly-does-stringstream-do

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/4702732/the-program-cant-start-because-libgcc-s-dw2-1-dll-is-missing

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/13768515/how-to-do-static-linking-of-libwinpthread-1-dll-in-mingw

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/46728353/mingw-w64-whats-the-purpose-of-libgcc-s-seh-dll

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/18138635/mingw-exe-requires-a-few-gcc-dlls-regardless-of-the-code

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/2425728/delete-vs-delete-operators-in-c

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/26695393/c-and-when-to-use-delete

Acessado por último em 16/02/2021

http://www.cplusplus.com/reference/sstream/stringstream/

Acessado por último em 16/02/2021

https://stackoverflow.com/questions/152555/h-or-hpp-for-your-class-definitions

Acessado por último em 16/02/2021