Universidade Federal do Rio Grande do Norte - UFRN
Instituto Metrópole Digital - IMD
Bacharelado em Tecnologia da Informação - BTI

Disciplina: Linguagem de Programação I - LP1
Docentes: Everton Cavalcante
Silvio Sampaio
Discente: Joaliton Luan

Laboratório 10 - Tratamento de Exceções

1. Introdução

Este relatório tem como objetivo explicar de forma sucinta e direta a metodologia utilizada para implementação e desenvolvimento do Laboratório 10 da disciplina de Linguagem de Programação I. Para tanto, utilizou-se conhecimentos obtidos em sala de aula para identificar e tratar possíveis erros (exceções) nos códigos presentes da biblioteca gerada no Laboratório 9.

2. Detalhes da implementação

O projeto foi implementado seguindo boas práticas de programação. A implementação foi totalmente baseada em modularização de arquivos, respeitando um determindado sistema de diretórios, definidos a partir das extensões dos arquivos e seguindo uma linha hierárquica. As funções para o funcionamento correto do programa, a princípio, foram implementadas no Laboratório 9 e posteriormente modificadas neste Laboratório.

2.1 Organização do Projeto

O projeto foi desenvolvido e organizado em modularização de arquivos. E, como já citado, divido em diretórios conforme suas extensões e aplicações.

2.1.1 ARQUIVOS

No diretório raiz encontram-se a descrição do projeto, o *Makefile* e o arquivo de configuração *Doxygen* (*Doxyfile*).

2.1.2 BIN

No diretório *bin* encontram-se os executáveis gerados a partir da compilação das bibliotecas através do *Makefile*.

2.1.2 BUILD

No diretório *build* encontram-se os arquivos objetos (.o) gerados pelo compilador *G++*.

2.1.3 DOC

No diretório *doc* encontram-se os arquivos gerados pela documentação automática através do programa *Doxygen*.

2.1.4 INCLUDE

No diretório *include* encontram-se os arquivos *headers* que declaram as funções genéricas presentes na biblioteca.

2.1.5 LIB

No diretório *lib* encontram-se as bibliotecas geradas pela compilação do programa através do *G++*. As bibliotecas geradas encontram-se tanto nas suas versões estáticas e dinâmicas para *windows* e *linux*.

2.1.6 SRC

No diretório *src* encontra-se o arquivo corpo que contém a função principal do programa para testar os casos de tratamentos de exceções implementados para a biblioteca do Laboratório 9.

3. Metodologia

A metodologia utilizada foi totalmente baseada no sistema de versionamento. Para este projeto utilizou-se a ferramente de versionamento *git* e um repositório remoto no *github.com* no qual foram armazenados as alterações e implementações feitas para este Laboratório.

3.1 Exceções encontradas

Foram encontradas diversas exceções que poderiam ser tratadas pelo criador da biblioteca, dentre elas destacaram-se principalmente as seguintes exceções: envio de valor inválido para o tamanho do vetor nos algoritmos de busca e de ordenação, tentativa de remoção de um elemento presente nas TEDs vazias e tentativa de impressão dos próprios elementos das TEDs caso elas estejam vazias. Também foram encontradas exceções para os casos de alocação inválida de memória em todas as esferas da biblioteca.

3.2 Soluções implementadas

Foram implementadas soluções para a maioria das exceções encontradas. Realizadas da seguinte forma:

3.2.1 TAMANHO INVÁLIDO DE VETOR

Para os casos em que o usuário envie um tamanho inválido de vetor (menor que 1), o programa identifica e trata esses casos, exibindo um sinal de erro, não permitindo a execução da função e encerrando-a imediatamente. Caso necessário, retorna-se o valor default da função.

3.2.2 REMOÇÃO DE ELEMENTO INEXISTENTE

Para os casos em que o usuário tente remover um elemento em uma TED vazia, esta exceção é tratada exibindo-se um alerta para erro, impedindo a execução da remoção e finalizando a função imediatamente.

3.2.3 EXIBINDO UMA TED VAZIA

Para os casos em que o usuário tente exibir uma TED que esteja vazia, esta exceção será tratada exibindo um sinal de alerta e impedindo a impressão da TED, em seguida, finaliza-se a função.

3.2.4 ALOCAÇÃO FALHA DE MEMÓRIA

Esta execeção não foi tratada neste Laboratório justificada pelo fato do próprio compilador tratar esta exceção, exibindo sinal de erro e encerrando imediatamente a aplicação.

4. Discussão

Como desejado, as exceções que necessitavam de um tratamento foram identificadas e devidamente consertadas, seguindo as regras de tramento de exceções da linguagem C++, para isto, implementou-se classes filhas herdeiras de <*exception*> que conseguissem tratar os erros identificados.