Laboratório V
Biblioteca e STL

1. Objetivo

Trabalho individual com o objetivo deste exercício é colocar em prática conceitos do paradigma de Programação Orientada a Objetos (POO) na linguagem de programação C++, em particular pela implementação de Bibliotecas e o uso de STL's.

2. Orientações gerais

Você deverá observar as seguintes observações gerais na implementação deste exercício:

1) Apesar da completa compatibilidade entre as linguagens de programação C e C++, seu código fonte não deverá conter recursos da linguagem C nem ser resultante de mescla entre as duas linguagens, o que é uma má prática de programação. Dessa forma, deverão ser utilizados estritamente recursos da linguagem C++.

2) Você deverá utilizar apenas um editor de texto simples (tais como o Gedit ou o Sublime) e o compilador em linha de comando, por meio do terminal do sistema operacional Linux.

3) Durante a compilação do seu código fonte, você deverá habilitar a exibição de mensagens de aviso (*warnings*), pois elas podem dar indícios de que o programa potencialmente possui problemas em sua implementação que podem se manifestar durante a sua execução.

4) Aplique boas práticas de programação. Codifique o programa de maneira legível (com indentação de código fonte, nomes consistentes, etc.) e documente-o adequadamente na forma de comentários. A título de sugestão, anote o seu código fonte para dar suporte à geração automática de documentação utilizando a ferramenta Doxygen (http://www.doxygen.org/). Consulte o documento extra disponibilizado na Turma Virtual do SIGAA com algumas instruções acerca do padrão de documentação e uso do Doxygen.

5) Busque desenvolver o seu programa com qualidade, garantindo que ele funcione de forma correta e eficiente. Pense também nas possíveis entradas que poderão ser utilizadas para testar apropriadamente o seu programa e trate adequadamente possíveis entradas consideradas inválidas.

6) Para melhor organização do seu código, implemente diferentes funções e faça a separação da implementação do programa entre arquivos cabeçalho (.h) e corpo (.cpp).

3. Tarefas

Implemente em C++ uma biblioteca que gera monstro para o conjurador que perdeu todos os seus em batalhas. Para isso o conjurador deve elevar seu espirito para o *Eter das Criaturas*. No salão principal está Saruman, o criador, que tem o poder de conceder novos monstros aos conjuradores derrotados.

O conjurador deve utilizar de toda a sua rusticidade e vencê-lo em um duelo de Jokenpo.

Para isso, vocês devem seguir o seguinte fluxo de trabalho:

1. Só poderá ter acesso ao Eter das Criaturas, os conjuradores que perderem todos os seus monstros.
2. Saruman só jogará cinco partidas de Jokenpo.
3. Cada partida vencida pelo conjurador, Saruman lhe concederá um novo monstro.

3.1. Especificação do Projeto

O código desenvolvido deve seguir as especificações abaixo:

1. O Jokenpo deverá ser uma biblioteca.
2. O Gerador deverá ser outra biblioteca.
3. O Grimório Mestre de Saruman deverá ser um conteiner STL.
4. Os monstros sequem as definições do Laboratório IV.
5. Deveram existir monstros fortes, monstros normais e monstros fracos nesse grimório.
6. A probabilidade de Saruman entregar um monstro fraco e maior que um normal e a de um normal é maior que a de um forte.
7. Cada vitoria do conjurador, Saruman dará um monstro que deve ser adicionado no Grimório do Conjurador.
8. Saruman tirará onda com você a cada vitoria dele.
9. Acabado as cinco partidas, Saruman expulsa o conjurador.
10. Saruman não perde monstros, afinal, ele é O Criador.

3.2. Organização do Projeto

O código deverá ser devidamente comentado e anotado para dar suporte à geração automática de documentação no formato de páginas Web (HTML) utilizando a ferramenta Doxygen. Para maiores informações, você poderá acessar a página do Doxygen na Internet (http://www.doxygen.org/).

O código do projeto deve seguir a configuração de pastas e arquivos:

1. /bin – código executável
2. /src – código fonte
3. /docs – documentação

4. makefile

5. `README` – arquivo contendo informações sobre: configuração, compilação e execução. Também deve conter uma sessão com as informações sobre quais arquivos e as linhas que contêm: Uso de alocação dinâmica de memória. Sobrecarga de funções; Pilha e fila implementada para o livro e os usuários; Sobrecarga de Operadores.

4. Entrega

Todos os códigos fonte referentes à implementação do trabalho deverão ser disponibilizados **sem erros** de compilação e devidamente testados e documentados através do repositório Git. Além disso, você deverá submeter, até 23:59 do dia 20 de junho de 2017, um único arquivo compactado através da opção *Tarefas* na Turma Virtual do SIGAA contendo:

- (1) os mesmos arquivos de código fonte disponíveis no repositório Git;
  - Jokenpo.
  - Gerador de Monstros.
  - Batalha dos Monstros.
- (2) a documentação do projeto na forma de páginas HTML, geradas automaticamente com a ferramenta Doxygen;

É importante destacar que serão avaliados **única** e **exclusivamente** os arquivos submetidos via SIGAA.

5. Avaliação

A avaliação deste trabalho será feita principalmente sobre os seguintes critérios: (1) utilização correta dos conteúdos vistos nas aulas presenciais da disciplina; (2) a corretude da execução do programa implementado, que deve apresentar saída em conformidade com a especificação e as entradas de dados fornecidas; (3) a aplicação correta de boas práticas de programação, incluindo legibilidade, organização e documentação de código fonte, e; (4) qualidade do relatório técnico produzido. O trabalho possuirá nota máxima de 10,00 (dez) pontos para a terceira unidade, distribuídos de acordo com a seguinte composição:

Itens Avaliados	Nota máxima
Biblioteca Jokenpo;	2,50
Biblioteca Geradora;	3,00
Uso de conteiner STL no Grimório Mestre adequado ao problema;	0,50
Probabilidade das forças dos monstros;	0,60
Adição do monstro no Grimório do conjurador;	0,20
Uso de construtores;	0,10
Uso consistente de alocação dinâmica de memória e uso do This.	0,20
Limite 5 Partidas;	0,20
Documentação do código.	0,50
Integração das bibliotecas no projeto do laboratório IV;	1,00
Sobrecarga de operadores;	0,20
Criação do Makefile.	0,5
Criação do Readme.	0,25
	0,25
Total	10

Por sua vez, o não cumprimento de algum dos critérios abaixo especificados poderá resultar nos seguintes decréscimos, calculados sobre a nota total obtida até então:

Falta	Decréscimo
Programa apresenta erros de compilação, não executa ou apresenta saída incorreta	–70%
Falta do README e/ou Makefile	-0,5
Implementação na linguagem C ou resultante de mistura entre as linguagens C e C++	–30%
Programa compila com mensagens de aviso (*warnings*)	–50%
Plagiarismo	–100%