ALGORITMOS E ESTRUTURAS DE DADOS I (SCC-202)

Trabalho 1

Professor: Dr. Thiago A. S. Pardo (*taspardo@icmc.usp.br*)

Estagiário PAE: Fernando Alva Manchego (*falva@icmc.usp.br*)

1. Objetivo

Empregar os conhecimentos adquiridos em aula sobre estruturas de dados (TAD, listas, pilhas e filas) em uma implementação de uma aplicação para computador, seguindo um conjunto de especificações funcionais e de projeto, assim como boas práticas de programação e documentação.

2. Indicações Gerais

- O trabalho deve ser desenvolvido em **grupos de dois alunos**.
- Cada grupo deverá **escolher uma das opções** descritas na seção 5.
- Serão aceitos somente trabalhos em **linguagem de programação C**.
- Data de apresentação: **02/11 (até meia noite)**

3. Critérios de Avaliação

Os trabalhos serão avaliados de acordo com os seguintes critérios:

- Usabilidade da interface: a interface com o usuário deve ser clara, flexível e intuitiva;
- Corretude do programa: o programa deve fazer o que foi especificado;
- Estruturas de dados utilizadas: adequação e eficiência;
- Observação dos "bons modos" da programação: TAD, modularidade do código, documentação interna, indentação, etc.

A cada dia de atraso, 1 ponto é descontado da nota. Lembrem-se de que a média final dos trabalhos deve ser maior ou igual a 5 para que o aluno seja aprovado e que os trabalhos têm peso 3 na média final.

O **plágio** de programas não será tolerado. Quaisquer programas similares terão nota zero independentemente de qual for o original e qual for a cópia.

4. Entrega do Trabalho

A entrega dos trabalhos será via e-mail. Serão requeridos:

- Um arquivo comprimido (RAR, ZIP, 7z, etc.) com (a) arquivos de código-fonte do programa e (b) arquivo executável do programa (pode ser necessário trocar a extensão para enviar por e-mail – troque o .exe por .ex, por exemplo);

- Relatório breve (documentação externa) do programa de, no máximo, 5 páginas, contendo, pelo menos, (a) breve descrição do trabalho feito, (b) apresentação das estruturas de dados utilizadas e justificativas e (c) uma seção detalhando como compilar e rodar o programa, com telinhas de exemplo da execução do sistema. Deixe claro que compilador usou (que versão, se utilizou alguma biblioteca diferente, etc.).

5. Especificações

Nesta seção são apresentadas as especificações funcionais do programa que será desenvolvido. Lembrar que só deve escolher uma das seguintes opções para implementar.

Qualquer que seja a opção selecionada, faça duas implementações da estrutura de dados (usando TADs), sendo que uma implementação deve ser estática e a outra dinâmica. O programa principal deverá usar apenas um dos dois TADs de cada vez: teste uma vez com um TAD, depois compile novamente para testar o segundo TAD SEM ALTERAR QUALQUER COMANDO NO PROGRAMA PRINCIPAL (exceto o nome do arquivo do TAD no include: o arquivo .h).

5.1. Opção 1: Let's get this party started!

A nova diretoria do CAASO acha que a forma atual de controlar a participação dos sócios nas festas não é apropriada. Acreditam que utilizar um sistema de informação para automatizar o processo permitiria oferecer um melhor serviço aos sócios. Para isso, solicitam a ajuda dos alunos de Alg1 para a implementação do sistema. Depois de realizar algumas entrevistas com a diretoria, conseguiu-se determinar o estado atual da informação e, também, determinar os requerimentos funcionais do sistema a ser implementado.

Informação Disponível

Toda a informação encontra-se armazenada em arquivos de texto, assim:

- **Curso.txt**: Contém a informação dos cursos da universidade. Cada linha possui o código do curso e o seu nome.
- **Socio.txt**: Contém a informação dos socios. Cada linha possui o código do sócio, nome, sobrenome, código do curso do sócio e data (dd-mm-aaaa) em que ele se tornou sócio.
- **Festa.txt**: Contém a informação das festas realizadas. Cada linha possui o código da festa, nome e data (dd-mm-aaaa) em que foi realizada.
- **Participacao.txt**: Contém a informação de quais sócios foram em quais festas. Cada linha possui o código de uma festa e o código de um sócio.

Um exemplo do conteúdo dos arquivos é apresentado a seguir:

CURSO.TXT

```
C1 Ciências da Computação
C2 Química
C3 Estatística
.....
```

SOCIO.TXT

```
S1 João Silva C2 29-08-2007
S2 José Santos C3 06-10-2010
S3 Maria Souza C1 15-09-2011
.....
```

FESTA.TXT

```
F1 Minerva 15-04-2008
F2 Black & White 20-11-2009
F3 Metamorfose 01-10-2010
.....
```

PARTICIPACAO.TXT

```
F1 S3
F1 S2
F3 S1
.....
```

Requerimentos Funcionais

Utilizando a informação descrita, com base em listas e filas (pelo menos), o sistema a ser implementado deve permitir realizar as seguintes operações:

- **Cadastro de sócios e festas.** O sistema deverá solicitar ao usuário toda a informação que considere necessária. Todos os dados são obrigatórios.
- **Listagem de sócios, festas e participações.** O sistema deverá mostrar a informação de todos os registros no sistema ordenados por código. No caso das participações, o sistema não deverá mostrar os códigos (pois não são explicativos), mas os nomes correspondentes às festas e aos sócios. A mesma coisa deve ser levada em consideração para listar os alunos e apresentar o curso ao qual eles pertencem.
- **Registro de interesse na festa.** Os sócios poderão registrar o seu interesse em participar de uma ou várias festas. Depois de registrar o seu interesse, eles entram em uma fila de espera para cada festa em que desejam participar.
- **Registro automático de participação.** Em um determinado momento, o organizador de uma festa indicará quantas pessoas podem participar dela (capacidade máxima da festa). Após isso, o sistema, automaticamente, deverá registrar a participação na festa dos sócios que registraram o seu interesse. Se a capacidade máxima da festa foi atingida, a inscrição nessa festa é fechada (mesmo que todos que tenham manifestado interesse não consigam entrar).
- **Registro manual de participação.** Se a capacidade máxima não foi atingida, o registro da participação na festa ainda fica em aberto e poderá ser realizado manualmente, solicitando ao usuário toda a informação necessária. Novamente, este tipo de registro só será possível até atingir a capacidade máxima da festa.

Ao iniciar a aplicação, o sistema deve carregar os dados dos arquivos de texto em memória, usando as estruturas de dados mais apropriadas. Quando a aplicação for fechada, os dados devem voltar para os arquivos de texto, considerando os novos registros inseridos pelo sistema. Se for necessário criar novos arquivos de texto para armazenar informação diferente daquela contida nos arquivos originais, a estrutura deles pode ser estabelecida pelo grupo desenvolvedor.

5.2. Opção 2: You can play one, but can you make one?

Um dos tipos de jogos mais conhecidos é o RPG (*role-playing games*). A história típica envolve um herói que deve cumprir uma missão, passando por diferentes desafios pelo caminho, como monstros e inimigos. No caminho, o herói pode encontrar objetos (armas, poções, etc.) que melhoram as suas próprias capacidades (vida, ataque, defesa, etc.) e servem para derrotar os inimigos. Pouco a pouco, tanto o herói como os inimigos e os desafios vão subindo de nível, incrementando a dificuldade do jogo.

Solicita-se a implementação de um pequeno jogo RPG. Os componentes do jogo, assim como as regras para jogar e a interface com o jogador são especificados nas seguintes seções.

Componentes do Jogo

- **Níveis:** O jogo possuirá um número de níveis que pode ser fixo ou aleatório (decidido no início da aplicação) e deve ser maior do que um.
- **Mapa:** Cada nível possuirá um mapa/caminho. O tamanho do caminho pode ser fixo ou aleatório (decidido no início do nível). Um caminho estará constituído de caixinhas, cada uma das quais podendo estar vazia, ter um **elemento** (ver descrição mais a frente) ou um **inimigo** (ver descrição mais a frente). A última caixinha do mapa indica o final do nível.
- **Elementos:** Existem dois tipos de elementos: armas e poções. Todo elemento possui um nome e um peso. Além disso, toda arma tem uma capacidade de ataque (um número), e toda poção uma capacidade de cura (um número de pontos de vida).
- **Inimigos:** Todo inimigo possui um nome, uma capacidade de ataque e um número de pontos de vida. Os valores dessas últimas podem são aleatórios.
- **Herói:** O herói possui um nome, uma quantidade de pontos de vida e dois objetos: um cinto e uma mochila. No cinto, o herói pode colocar elementos em qualquer posição, assim como acessa-los à vontade. Na mochila, o herói só pode colocar um objeto sobre o outro e, em todo momento, só pode acessar o elemento que está sobre os demais. Para poder atingir os elementos mais ao fundo, deverá retirar (e perder) os elementos anteriores. O cinto possui um número fixo de espaços que podem conter elementos. Cada um desses espaços tem uma capacidade máxima de peso de elemento. Por outro lado, a mochila não possui limite de espaço nem peso.

Regras do Jogo

- **Percurso no mapa:** O jogador avança de caixinha em caixinha até atingir o final do mapa.
- **Interação com elementos:** Quando o herói chega numa caixinha com um elemento, ele pode decidir pegá-lo ou descartá-lo. No primeiro caso, ele deve indicar se deseja colocá-lo na mochila ou no cinto. No segundo caso, o objeto desaparece do caminho.

- **Batalhas:** Quando o herói chega numa caixinha com um inimigo, começa uma batalha. O jogador pode escolher qualquer arma que tenha no cinto ou na mochila (considerando as limitações de acesso a cada uma delas) para utilizá-la durante a batalha. Por turnos, o jogador e o inimigo atacam (os pontos de vida são diminuídos considerando a capacidade de ataque da arma ou do inimigo) até que um deles perca o total de pontos de vida. Se o jogador ganha a batalha, ele pode continuar no mapa; senão, o jogo termina.
- **Uso de poções:** As poções permitem ao jogador recuperar alguns pontos de vida. Quando o jogador decide usar uma poção, os pontos de vida dela são passados ao jogador até atingir a sua capacidade total máxima. Depois disso, a poção desaparece.
- **Movimentação:** Cada vez que o jogador chega numa caixinha, o jogo deve indicar o que ele encontrou ali e permitir o tipo de interação correspondente (se for o caso), seguindo as seguintes regras:
    - Quando o jogador estiver em uma caixinha vazia, ele poderá manipular o conteúdo do cinto ou da mochila. Isto é, ele poderá retirar/usar elementos de qualquer desses objetos. Além disso, ele poderá avançar à próxima caixinha.
    - Quando o jogador estiver em uma caixinha com um elemento, pelo menos o processo de "interação com elementos" poderá ser realizado. Depois disso, ele passa automaticamente à próxima caixinha.
    - Quando o jogador estiver em uma caixinha com um inimigo, pelo menos o processo de "batalha" poderá ser realizado. Depois disso, ele passa automaticamente à próxima caixinha.
- **Fim do jogo:** O jogo pode terminar quando o jogador perde uma batalha ou quando ele atinge o final do último nível.

Atenção: listas e uma pilha (pelo menos) devem ser usadas.

Interface do Jogo

- No inicio do jogo, a aplicação deve solicitar o nome do jogador.
- Em todo momento, deve-se apresentar o conteúdo total do cinto e o elemento na parte de cima da mochila. Também os pontos de vida restantes do jogador.
- Em todo momento, dependendo do tipo de caixinha em que o jogador esteja, as opções de movimentação ou interação disponíveis devem ser apresentadas.

O número de regras ou funcionalidades pode ser incrementado se o grupo desejar (aluno USP sempre quer fazer mais!), sempre e quando cumpra com, pelo menos, as regras/funcionalidades indicadas nessa seção.