CMP1056 - Técnicas de Programação II
Max Gontijo de Oliveira
1º Trabalho - Jogo de Batalha

1 Resumo

Criar um programa em C++ utilizando conceitos de orientação à objetos que implemente um jogo de batalha de turnos entre dois jogadores.

2 Detalhes

O jogo de batalha a ser implementado consiste em dois jogadores disponibilizados em um tabuleiro matricial se enfrentem até a morte. A batalha deverá ser realizada por meio de movimentos em turnos, de modo que, em cada turno, cada jogador deverá fazer jogadas relacionadas ao turno (um ataca e o outro defende). Ambos os jogadores escolhem suas jogadas em segredo e, quando finalizadas as escolhas, as jogadas são realizadas, começando por quem ataca.

Outros elementos também estarão presentes no tabuleiro: rochas e baús . As rochas impedem o movimento de um jogador em sua direção. Contudo podem ser quebradas com um golpe. Os baús não impedem o movimento em sua direção, podendo o jogador ocupar a mesma célula que um baú ocupa.

- O atacante pode escolher entre as seguintes jogadas:
  - Ataque (arma e direção devem ser informados)
  - Movimento (direção deve ser informada)
  - Uso de um item (item deve ser escolhido)
  - Delay (não fazer nada)
  - Abrir baú (deve estar em uma célula com um baú)
- O defensor pode escolher entre as seguintes jogadas:
  - Bloqueio (arma/escudo e direção devem ser informados)
  - Movimento (direção deve ser informada)
  - Uso de um item (item deve ser escolhido)
  - Delay (não fazer nada)
  - Abrir baú (deve estar em uma célula com um baú)

Os ataques realizados irão diminuir os pontos de vida do adversário (quando acertar). O jogo acaba quando um dos jogadores tiver seus pontos de vida zerados.

A Figura 1 apresenta uma visualização do que seria o tabuleiro em algum instante de uma partida.

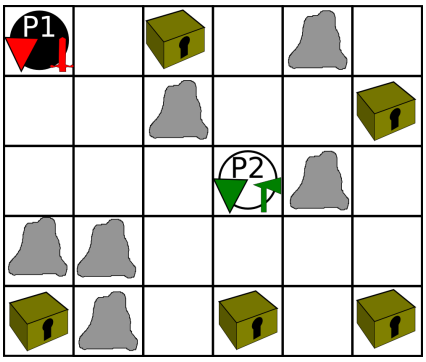


Figura 1: Exemplo de tabuleiro

Na figura, os elementos P1 e P2 são, respectivamente os jogadores 1 e 2. O jogador 1 carrega uma espada na mão esquerda e um escudo na mão direita, enquanto o jogador 2 carrega um machado na mão esquerda e um escudo na direita. Os símbolos amarelos representam baús e os cinzas representam rochas.

2.1 Tabuleiro

O tabuleiro será matricial (como em damas ou xadrez). O tamanho $N \times M$ do tabuleiro deverá ser informado antes da partida. Considerando $i \in \{0, 1, 2, \ldots, N-1\}$ o índice da linha e $j \in \{0, 1, 2, \ldots, M-1\}$ o índice da coluna, o jogador 1 deverá ser posicionado na posição $(i, j) = (0, 0)$ do tabuleiro, enquanto o jogador 2 na posição $(i, j) = (N-1, M-1)$.

Os demais itens (rochas e baús) deverão ser inseridos no tabuleiro em posições aleatoriamente selecionadas (ver funções `srand`, `time` e `rand`). As quantidades de rochas e baús também devem ser configuradas antes da partida.

Os jogadores começam a partida com uma arma relacionada a raça escolhida na mãe esquerda. A mão direita inicia sem nada. Os jogadores também começam sem nenhuma poção de cura.

2.2 Raças

Cada jogador deverá escolher entre as raças Humano e Orc. Se escolher humano, o jogador terá ainda que escolher entre ser um Bárbaro ou um Rogue, enquanto se escolher Orc deverá escolher ainda entre ser um Uruk-Hai ou um Troll.

O que diferencia um Humano de um Orc é apenas a quantidade de poções de cura que pode carregar. Humano pode carregar até três, enquanto Orc pode carregar até dois.

O que difere Bárbaro, Rogue, Uruk-Hai e Troll são seus pontos de vida, força e destreza iniciais.

A força irá interferir no dano de ataque com espadas ou machado e na capacidade de carregar a arma ou escudo por conta do peso; a destreza irá interferir na quantidade de células que o jogador pode avançar em um movimento e na capacidade de empunhar armas de maior complexidade (arcos).

2.3 Itens de Mão

Os itens de mão são armas ou escudos que o jogador poderá empunhar durante a partida. Cada item de mão possui quatro atributos importantes: dano, defesa, força requerida e a destreza requerida.

O dano denota o quanto de vida a arma poderá tirar do adversário. A defesa se refere a quantidade de dano que poderá ser evitada quando o jogador for atingido. A força requerida e a destreza requerida indicam o quanto de força e destreza que é necessário que o jogador tenha para poder empunhar e utilizar a arma/escudo. Caso uma dessas restrições não seja satisfeita, o jogador não poderá pegar o item, que ficará onde ele o encontrou.

Importante: Cada jogador deverá começar com uma arma na mão esquerda. Essa arma será fixa em relação à raça/classe e a seus atributos.

Raça	Classe	Arma	Dano	Defesa	Força Requerida	Destreza Requerida
Humano	Bárbaro	Espada	7	3	20	50
	Rogue	Arco	5	1	15	60
Orc	UrukHai	Espada	8	2	40	50
	Troll	Machado	10	3	70	15

Observação: O arco possui flechas infinitas.

2.4 Poções de cura

As poções de cura são itens que aumentam a vida do jogador em 50% do seu máximo de vida. Quando utilizada, a poção de cura irá sair do inventário do jogador, abrindo vaga para outra poção que possa ser encontrada no tabuleiro.

Importante: Cada jogador deverá começar sem nenhuma poção de cura e só poderá pegar outra poção se tiver espaço em seu inventário.

2.5 Rochas

As rochas são peças que obstruem o movimento de um jogador. Contudo, elas podem ser destruídas com um único golpe de espada ou machado ou com uma flecha. Após destruída, a rocha sai do tabuleiro e não volta mais, possibilitando que um jogador possa passar por onde ela se encontrava.

2.6 Baús

Os baús contém itens que só podem ser identificados quando abertos. Dentro deles podem haver uma poção de cura ou uma das armas ou um escudo ou uma bomba.

Se o item do baú for uma poção de cura, o jogador irá imediatamente e automaticamente pegar a poção e adicionar em seu inventário se houver espaço. Caso não haja, a poção permanecerá dentro do baú.

Se o item do baú for uma arma ou escudo, o jogador irá ver os atributos do item e poderá optar pegar o novo item e carregá-lo em uma de suas mãos. Se o jogador já tiver com as duas mãos ocupadas, será obrigado a se desfazer de um deles caso queira pegar o novo item.

Se o item do baú for uma bomba (sim, uma bomba!), ocorrerá uma explosão que irá tirar em vida, a quantidade de dano que ela causar. Esse dano deverá ser atribuído à bomba de forma randômica, variando entre 5 e 15.

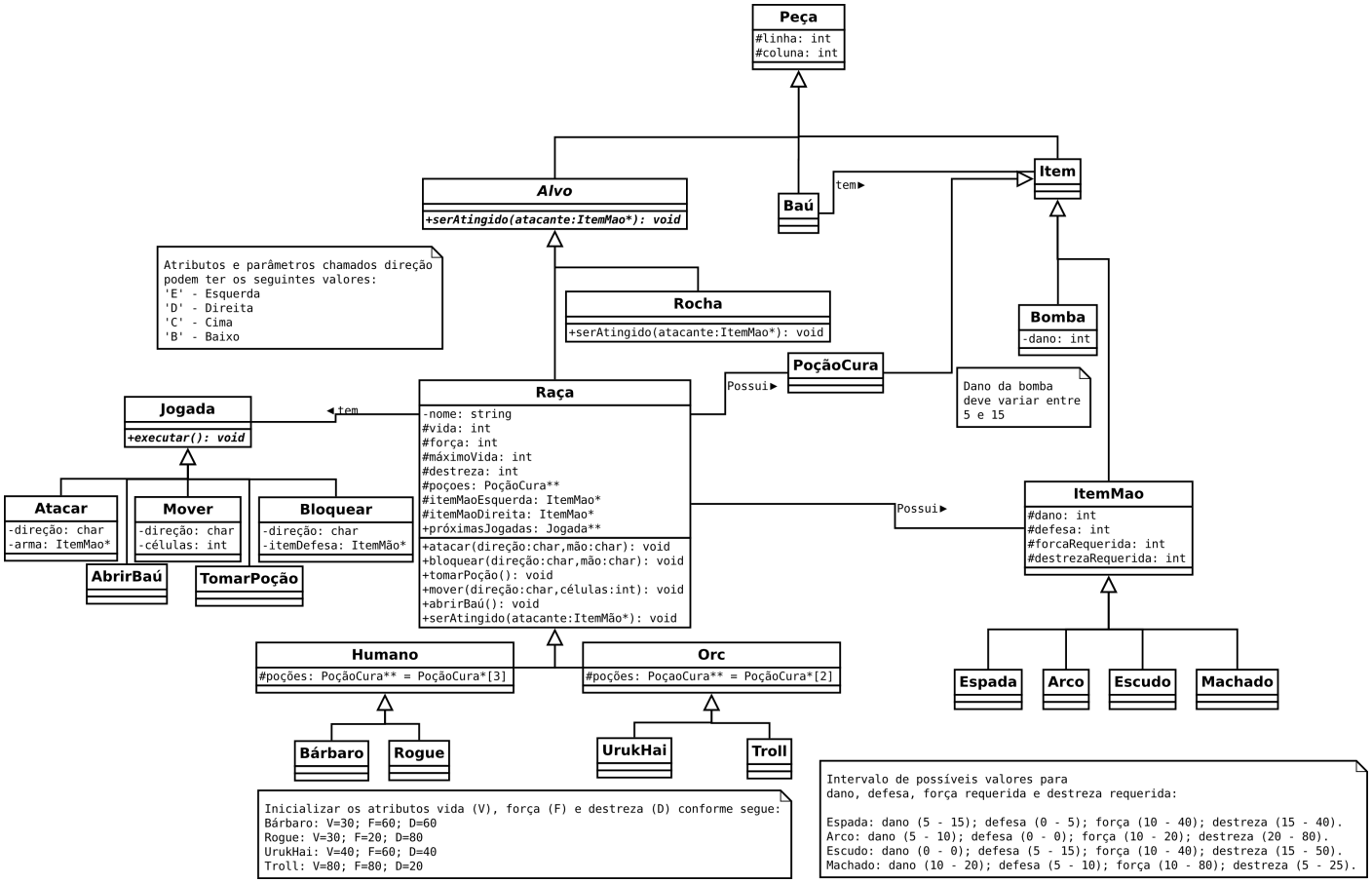
Importante: O item dentro de cada baú deverá ser selecionado de forma randômica, entre as opções, bomba, poção, espada, arco, escudo e machado. Os atributos de cada item (com exceção da poção), deverão ser selecionados de forma randômica também, respeitando os intervalos de possíveis valores.

3 Implementação

O programa deverá ser totalmente escrito em C++. Caso o aluno tenha interesse, poderá utilizar bibliotecas de interfaces gráficas ricas, como QT, GTK ou OpenGL.

3.1 Orientação à objetos

O programa deverá ser desenvolvido utilizando os conceitos de orientação à objetos. Conforme já conversado em sala de aula, **é obrigatório** que sejam criadas e utilizadas as classes apresentadas no seguinte diagrama de classes:



Importante: Note que esse diagrama não contempla todas as classes do jogo a ser implementado. Ele contempla apenas a parte lógica do jogo, que independe de interface gráfica. Classes referentes à configuração, entrada e saída de dados deverão ser implementadas à parte.

Dica geral: Para identificar qual é o tipo de um objeto específico, utilize o comando `typeid` da biblioteca `typeinfo`.

3.2 Classe `Raça`

A classe `Raça` possui alguns atributos bastante importantes.

Atributo	Descrição
vida	Quando zerado, o jogador morre.
máximoVida	Utilizada para determinar quanto uma poção cura.
força	Utilizada para cálculo de ataque e para carregar itens de mão.
destreza	Utilizada na defesa, no movimento e para carregar itens de mão.
poções	Poções carregadas pelo jogador.
itemMãoEsquerda	Arma ou escudo
itemMãoDireita	Arma ou escudo
próximasJogadas	Jogadas que o jogador irá executar no turno

A classe ainda conta com cinco métodos (um herdado).

Método	Descrição
atacar	Adiciona à `próximasJogadas` uma jogada `Atacar`.
bloquear	Adiciona à `próximasJogadas` uma jogada `Bloquear`.
tomarPoção	Adiciona à `próximasJogadas` uma jogada `TomarPoção`.
mover	Adiciona à `próximasJogadas` uma jogada `Mover`.
abrirBaú	Adiciona à `próximasJogadas` uma jogada `AbrirBaú`.
serAtingido	Deve ser implementado de forma que o jogador atingido possa lidar com o ataque.

3.2.1 Método `serAtingido` da classe `Alvo`

Jogadores (`Raça`) e rochas (`Rocha`) podem ser atingidos por ataques. Assim, quando uma jogada do tipo `Atacar` é executada com sucesso, o alvo terá o método `serAtingido` invocado. Na classe `Rocha`, o método deve implementar apenas a remoção da rocha do tabuleiro. Na classe `Raça`, o método deverá calcular a intensidade do ataque, considerando dano da arma, o bloqueio do escudo/arma do alvo e a destreza do alvo. Paralelamente, no mesmo turno, o alvo poderá ter efetuado um bloqueio.

Caso o alvo tenha efetuado um bloqueio na direção em que veio o ataque, a quantidade de vida perdida deverá ser calculada da seguinte forma:

$$vidaPerdida = max(0, danoArma - (defesaItemMaoEsquerdaAlvo + defesaItemMaoDireitaAlvo) \times (destrezaAlvo \times p + 1))$$

onde p é um número randômico entre 0 e 1;

Caso o alvo não tenha bloqueado ou tenha efetuado o bloqueio para um lado errado, o cálculo será o mesmo, sendo que $defesaItemMaoEsquerdaAlvo$ e $defesaItemMaoDireitaAlvo$ terão valor zero.

3.3 Jogadas

Em cada turno, um jogador assumirá o papel de atacante e o outro de defensor. O atacante terá o direito de efetuar quatro jogadas, enquanto o defensor poderá efetuar três.

As jogadas de cada jogador devem ser escolhidas. Após a escolha das jogadas, o jogo irá executar as jogadas uma a uma, alternadamente, começando e terminando pelo atacante.

Após o término das jogadas, os papeis se invertem e, quem era atacante vira defensor, quem era defensor vira atacante.

3.3.1 Atacar

Um ataque pode ser realizada a partir de qualquer célula do tabuleiro quando se está no papel de atacante. Contudo, o ataque só irá atingir um alvo se algumas restrições forem respeitadas.

- A direção em que o ataque é realizado precisa ter um alvo.
  - Os possíveis valores para o parâmetro `direção` são: `'E'`, `'D'`, `'C'` e `'B'`.
- Se for um ataque utilizando espada ou machado, a célula onde se encontra o alvo precisa estar adjacente.
- Se for um ataque utilizando arco, a célula precisa apenas estar na mesma direção. Nesse caso, o alvo será o primeiro que aparecer naquela direção.
- Escudo ou mão limpa não podem ser utilizados para atacar.

Quando o ataque atinge um alvo, então as consequências disso devem ser executadas, conforme descrito na Seção 3.2.1.

3.3.2 Bloquear

Um bloqueio pode ser realizado a partir de qualquer célula do tabuleiro quando se está no papel de defensor. Se trata de uma jogada com certo *delay*, pois o jogador estará bloqueando na jogada seguinte de seu atacante. Se no momento desse ataque o defensor estiver bloqueando na MESMA DIREÇÃO de onde vem o ataque, então os pontos de defesa e destreza serão ignorados, conforme descrito na Seção 3.2.1.

Caso o bloqueio seja executado e, na jogada seguinte, o atacante não efetue um ataque ou não acerte o alvo, então o bloqueio simplesmente não terá causado efeito algum.

3.3.3 Tomar poção

Essa jogada se refere a usar uma poção de cura que o jogador tenha guardado em seu inventário. Caso não haja nenhuma poção, a jogada irá acontecer da mesma forma, contudo, será perdida pois nada vai acontecer.

Quando a poção é tomada, os pontos de vida do jogador são acrescidos em 50% da quantidade de pontos de vida máximo que o jogador pode ter. Obviamente, esse aumento está limitado a esse mesmo máximo de pontos de vida.

3.3.4 Mover

Essa jogada recebe direção e quantidade de células que o jogador deseja avançar na direção informada. Contudo, o movimento possui algumas restrições.

- A quantidade de células que são efetivamente avançadas são limitadas ao resultado da seguinte operação: $destreza/20$.
- Se o método de movimentar for chamado com uma quantidade superior ao limite de células imposto pela destreza, a quantidade de células a serem avançadas será o limite.

- Se alguma célula tiver algo que bloqueie o avança do jogador, o movimento irá se limitar a esse bloqueio.
  - Somente duas coisas podem bloquear o movimento de um jogador: rochas e o outro jogador.
- Cada movimento só pode ser realizado em uma única direção. Se, por exemplo, o jogador poder andar três células, ele não pode andar duas para a direita e uma para baixo em uma única jogada. Para fazer isso, ele teria que realizar duas jogadas de movimento, sendo a primeira com 2 células à direita e a segunda, com uma para baixo.

3.3.5 Abrir baú

Quando existe um baú em uma célula, se o jogador estiver na mesma célula, poderá realizar essa jogada. Se essa jogada for realizada fora de uma célula que tenha um baú, ela simplesmente não terá efeito.

Essa jogada consiste no seguinte fluxo:

1. Baú e aberto, exibindo o item ao jogador.
2. Se for um item de mão:
   (a) Jogo mostra atributos dos itens de mão do jogador e do item que ela acaba de pegar.
   (b) Jogador decide se vai querer ficar com item.
   (c) Se o jogador quiser o item, deverá escolher a mão que irá segurar o item.
   (d) Se houver um item de mão na mão escolhida para o novo item, o item que estava na mão é deixado na mesma célula, podendo ser, posteriormente, pegado por outro jogador ou ele mesmo.
   (e) Jogador só poderá pegar o item se ele tiver força e destreza suficiente.
3. Se for uma poção:
   (a) O jogador só poderá pegar a poção se ele tiver espaço para carregar mais uma poção.
   (b) Se pegar a poção, poderá deixar um item de mão, se ele quiser.
4. Se for uma bomba:
   (a) A bomba simplesmente explode e causará no jogador o dano que ela tiver descrito em seu atributo.
   (b) Nesse caso, o jogador não poderá deixar nada dentro do baú na mesma jogada.
5. Baú é fachado novamente, vazio ou não.

O baú jamais sai do tabuleiro. Mesmo que ele esteja vazio. Só cabe um item dentro do baú.

4 Fluxo de execução do programa

O programa deverá efetuar a seguinte sequência de passos:

1. Programa possibilita configuração de uma nova partida.
   - Tamanho tabuleiro
   - Quantidade de rochas
   - Quantidade de baús
2. Jogador 1 escolher seu personagem, entre Bárbaro, Rogue, Uruk-Hai e Troll. Escolhe ainda seu nome.
3. Jogador 2 escolher seu personagem, entre Bárbaro, Rogue, Uruk-Hai e Troll. Escolhe ainda seu nome.
4. Tabuleiro é criado
   (a) Cria tabuleiro no tamanho especificado.
   (b) Insere o jogador 1 na posição $(i, j) = (0, 0)$.
   (c) Insere o jogador 2 na posição $(i, j) = (N - 1, N - 1)$.
   (d) Insere rochas no tabuleiro em posições randômicas.
   (e) Insere baús no tabuleiro em posições randômicas.
   (f) Para cada baú, selecione randomicamente o seu conteúdo.
      i. Selecionar randomicamente entre espada, arco, escudo, machado, poção ou bomba.
      ii. Selecionar randomicamente os atributos dos itens (exceto se for poção, pois não tem atributo).
5. Jogo é iniciado

6. Jogador 1 inicia no papel de Atacante e jogador 2, no papel de Defensor.
   (a) O Atacante escolhe suas **quatro** jogadas.
   (b) O Defensor escolher suas **três** jogadas.
   (c) As sete jogadas são executadas alternando entre jogada do Atacante e jogada do Defensor, começando pela primeira jogada do Atacante.
   (d) Se em alguma jogada o Atacante ou o Defensor zerar os pontos de vida, o jogo termina com a vitória, obviamente, de quem não zerou os pontos de vida.
   (e) Se ninguém zerar os pontos de vida dentro das sete jogadas, o fluxo de jogadas recomeça , retornando ao passo 6(a), trocando os papeis dos jogadores (quem era Atacante passa a ser Defensor e vice-versa).
7. Programa apresenta uma mensagem de informando vitorioso e derrotado.

5 Entrega

O trabalho deverá ser entregue em duas etapas: a primeira é a entrega dos arquivos de código fonte via e-mail; a segunda é a apresentação do trabalho em sala de aula diretamente para o professor.

5.1 Código fonte

Os arquivos de código fonte deverão ser compactados em um único arquivo (ZIP, RAR, 7Z, etc.) e enviados até o dia 12/12/2014 às 20:30. Caso sejam enviados após essa data, o valor da nota total conquistada pelo aluno será multiplicada por 0.7.

5.2 Apresentação

Todos os alunos deverão apresentar o trabalho ao professor em sala de aula ATÉ o dia 15/12/2014.

Nenhum trabalho será avaliado após essa data.

Se o aluno quiser apresentar antes da data final, poderá fazê-lo. Basta informar ao professor.

A apresentação será em duplas e irá considerar a capacidade de CADA aluno de explicar todo o código desenvolvido ou partes específicas determinadas pelo professor.

6 Avaliação

A avaliação será realizada em duas etapas: uma referente à entrega do programa em si; a outra referente à apresentação.

A entrega vai considerar o respeito ao prazo e o uso de orientação a objetos de forma correta.

A apresentação vai considerar a explicação dos alunos acerca do código desenvolvido e a capacidade de os alunos responderem perguntas sobre o código e/ou os algoritmos. O professor poderá ainda solicitar alterações no código que os alunos deverão ser capazes de realizar.