Trabalho 2 - Jogando RPG

Introdução à Ciência de Computação

Prof. Moacir Ponti / João Batista
Monitores/PAE: Gabriel Cruz, Alex Sander Silva, Mariana Rodrigues, Cezar, Kelvin Oliveira

Implemente suas atividades sem compartilhar ou olhar código de seus colegas. Procure usar todos os conceitos vistos nas aulas. Documente a sua aplicação por meio de comentários no programa

Descrição do trabalho

Esse trabalho tem como objetivo treinar os primeiros passos em programação, fazendo o aluno pensar em:

1. Quais valores serão armazenados na memória;
2. Que informações deverão ser entradas pelo usuário;
3. Que informações deverão ser exibidas na tela;
4. Como utilizar as estruturas de decisão;
5. Como utilizar as estruturas de repetição;
6. Como processar os dados para obter o resultado desejado;

Jogo RPG

Drizzt Do'Urden é um personagem fictício ambientado no cenário de um jogo RPG. Muito hábil com suas duas cimitarras, ele está batalhando contra o seu arquirrival, a terrível assassina **Artemis Entreri**. Para derrotá-la, Drizzt deve fazer uma série de jogadas de dados e obter resultados maiores do que as defesas de Artemis, enquanto Artemis não obtém resultados melhores do que as defesas de Drizzt.

Nesse jogo, tudo é determinado através dos dados. O primeiro, **Dado de Determinação de Acerto (DDA)** é o dado que define se um ataque será autorizado ou não em cada rodada. O **Dado de Ataque (DA)**, por sua vez, determina o dano causado pelo ataque no oponente, ou seja, quantos pontos de vida o oponente perderá por causa do ataque. Note que são diferentes!

LUTA : corresponde a uma simulação de um confronto, onde os personagens alternam tentativas de ataque até que um deles não possua mais pontos de vida.

1. *Ordem*: Para definir quem começa, antes do início de cada luta, Drizzt e Artemis, *nessa ordem*, lançam o dado (DDA). Começa atacando aquele que tirar o maior número. Em caso de empate, convencionamos que Drizzt começa.

2. *Autorização de Ataque*: Para um personagem poder atacar o outro, o primeiro passo é lançar o DDA. Se o resultado obtido for estritamente maior do que a **Classe de Armadura (CA)** do oponente, o ataque é autorizado. Se bem-sucedido, o personagem pode agora utilizar sua arma para atacar o rival.

3. *Ataque*: Uma arma é representada pela notação *'NdP'*, em que 'N' simboliza o número de dados de ataque da arma (ou seja, número de ataques possíveis em um turno), e 'P' simboliza a **Força da Arma (FA)**, que indica o número de faces do DA. Exemplificando, uma arma '2d8' permite que, em cada ataque, sejam sorteados dois DA's com 8 faces. Por outro lado, uma arma '1d5' permite que apenas um DA com 5 faces seja sorteado.

4. *Dano ao oponente*: Em cada ataque autorizado, o resultado obtido no DA é o número de pontos de vida que deve ser descontado do rival. Quando a arma permitir jogar mais que um DA (ou seja, N $>$ 1), a soma dos 'N' resultados obtidos é descontado da vida do rival. Eventos especiais:

   - *Extra Damage*: é um evento especial em que, quando o valor obtido no ataque com a arma é o máximo possível, o número de pontos de vida a ser subtraído do oponente é o aumentado em 50%. Exemplos: se NdP = '2d5' e os dois dados obtiverem valor 5 totalizando 10, então o dano total será 15. Se NdP = '3d3' e os três dados obtiverem valor 3 totalizando 9, então o dano total será $\lfloor 13,5 \rfloor = 13$ (note que consideramos apenas a parte inteira do número, arredondando-o para baixo).
   - *Miss*: é um evento especial em que, se o valor obtido no ataque é o mínimo possível, consideramos que o atacante errou o golpe, e nenhum ponto de vida é descontado do oponente.

5. *Resolução da luta*: Assim que um ataque é finalizado, autorizado ou não, a vez de atacar passa para o oponente, e eles continuam alternando ataques até que um deles não possua mais pontos de vida. Nesse caso, consideramos que a luta acabou, declarando o personagem que ainda está "de pé" (ainda tem pontos de vida) como vencedor dessa luta.

BATALHA : consiste em uma disputa de *"Melhor de m lutas"*, em que m é um número ímpar e o personagem que vencer a maioria das lutas é declarado vencedor da batalha. Enquanto nenhum dos personagens obter as vitórias necessárias para vencer a batalha, começamos uma nova luta, retomando a vida inicial de ambos. Quando um dos personagens conseguir o número de vitórias suficiente, ou seja $\lfloor m/2 \rfloor + 1$, o declaramos vencedor da batalha e finalizamos o jogo.

Tarefa

Nesse trabalho, sua tarefa será simular a batalha entre os dois inimigos mortais. Os personagens já possuem alguns parâmetros pré-determinados:

Drizzt:

- *CA:* 10
- *Arma:* Cimitarra (FA = 1d9)

Artemis:

- *CA:* 7
- *Arma:* Espada de duas mãos (FA = 2d7)

No início do programa você deve ler quatro números inteiros: O número de lutas que compõem a batalha, a vida inicial de Drizzt, a vida inicial de Artemis e o número de faces do DDA, **nessa ordem**.

A partir daí você deve simular a batalha, iniciando a primeira luta, permitindo os ataques alternados dos personagens conforme explicado anteriormente. Quando encerrar uma luta, lembre-se de verificar se a batalha terminou (um dos personagens atingiu a o número de vitórias necessárias) e, caso contrário, inicialize as vidas dos personagens como sendo as vidas iniciais lidas na entrada, para assim poder iniciar uma nova luta.

Dica: Para simular os lançamentos dos dados, utilize a função *rand()* (Lembre-se que a função *rand()* gera números em um intervalo [0, RANDMAX], e que nesse programa você precisará gerar números no intervalo [1, Número de Faces do Dado desejado]).

A semente da geração aleatória, definida pela função *srand()* deverá ser definida **APENAS UMA VEZ** no programa, logo no ínicio, como sendo o quadrado do número de faces do DDA.

Entrada

A entrada é formada por quatro inteiros: O número de lutas, a vida inicial de Drizzt, a vida inicial de Artemis e o número de faces do DDA, **nessa ordem**.

Exemplo de Entrada:

```
3
20
20
16
```

Saída

Quando começar uma luta, o programa deve imprimir:

Luta <**x**>`\n` *(Sendo <**x**> o número que identifica tal luta (Luta 1, 2, 3,...))*

Durante a luta, em cada ataque bem-sucedido, o programa tem que imprimir:

<Nome_do_atacante> <Dano_do_ataque>`\n` *(Onde <Nome_ do_ atacante> é o nome do personagem que realizou o ataque, e <Dano_ do_ ataque> é o dano causado pela arma do atacante sobre a vida do defensor)*

Quando a luta tiver um vencedor, imprima:

Fim da luta. Vencedor: <Nome_do_vencedor>`\n` *(Onde <Nome_ do_ vencedor> é o nome do jogador que venceu a luta correspondente)*

Quando a batalha acabar, imprima:

Fim da batalha. Vencedor: <Nome_do_vencedor>`\n` *(Onde <Nome_ do_ vencedor> é o nome do jogador que venceu a batalha,* ***em letras MAIÚSCULAS****)*

Obs: Note os símbolos `\n` denotando uma quebra de linha da função `printf()`

Exemplo de Saída:

```
Luta 1
Drizzt 7
Artemis 4
Drizzt 7
Drizzt 4
Artemis 10
Drizzt 5
Fim da luta. Vencedor: Drizzt
Luta 2
Artemis 5
Artemis 9
Drizzt 2
Drizzt 2
Drizzt 2
Drizzt 8
Drizzt 7
Fim da luta. Vencedor: Drizzt
Fim da batalha. Vencedor: DRIZZT
```

Instruções

O trabalho será avaliado levando em consideração:

1. Realização dos objetivos
2. Representação correta da entrada e saída dos dados
3. Uso de comentários e estrutura no código (e.g. indentação, legibilidade)
4. Número de acertos no sistema Run.Codes
5. Uso da memória

ATENÇÃO:

- O projeto deverá ser entregue apenas pelo (`http://run.codes`) no formato de **código fonte**, ou seja apenas o código C.
- O prazo está no sistema run.codes
- Em caso de projetos **copiados** de colegas ou da Internet, todos os envolvidos recebem nota zero. Inclui no plágio a cópia com pequenas modificações, cópia de apenas uma parte ou função. Portanto programe seu próprio trabalho.