# **Aula Prática 9** Para os exercícios abaixo serem pontuados corretamente, você DEVE usar os nomes das funções que são descritos em cada exercício.

# Para conseguir compilar o seu programa, você deve implementar uma versão sintaticamente correta de todas as funções pedidas. Para o exercício 2, por exemplo, você pode implementar a seguinte função:

# int rolaDados() { return 0; }

# Para testar um exercício, inicie a execução no VPL e digite o número do exercício que gostaria de testar (2, 3, 4, 5 ou 6). Não haverá testes para o exercício 1. Depois, digite um número inteiro qualquer (ex: 1), que será a semente (seed) usada na função srand. Por fim, digite os valores que devem ser passado para as funções, caso elas tenham parâmetros.

# Para testar o exercício 4, você pode digitar: "4 1 18" e, nesse caso, o programa deve imprimir "3".

# Para testar o exercício 5, você deve digitar os valores dos campos de batalha de dois jogadores. Primeiro, digite os valores dos campos ataque, defesa, carisma e pontos\_vida do primeiro guerreiro (atacante). Depois, digite os valores dos campos ataque, defesa, carisma e pontos\_vida do segundo guerreiro (defensor). Exemplo de entrada em que o primeiro guerreiro tem todos os seus campos de batalha com valor 18 e o segundo com valor 17:

# 5 1

# 18 18 18 18

# 17 17 17 17

# Exercícios:

# Nesta prática, você deve criar um protótipo de um sistema de batalha entre guerreiros de um jogo. Para isso, resolva os exercícios a seguir.

# 1) Defina um novo tipo de dados chamado Guerreiro com os seguintes campos: ataque (inteiro), defesa (inteiro), carisma (inteiro), pontos\_vida (inteiro) e id\_guerreiro (inteiro).

# 2) Escreva uma função de nome rolaDados que simula a rolagem de três dados de seis faces tradicionais (1 a 6) e retorna a soma dessas rolagens. Note que somar os valores resultantes da rolagem de três dados de seis faces é diferente de rolar um dado que retorna um número entre 3 e 18. Importante: não use a solução do monitor do semestre 2020-01.

# 3) Escreva um procedimento de nome criaGuerreiro que recebe um Guerreiro por passagem de parâmetro por referência e que atribui valores aos seus campos de batalha: ataque, defesa e carisma, nessa ordem. Os seus campos de batalha devem receber um valor inteiro da função rolaDados. Depois, atribua um valor para o campo pontos\_vida, que deve receber a soma dos valores retornados por três execuções da função rolaDados. Assuma que o campo id\_guerreiro já foi preenchido fora da função.

# 4) Escreva uma função de nome bonusCarisma que recebe um valor de carisma como parâmetro e retorna o bônus dado por esse valor de carisma. A tabela de bônus funciona da seguinte maneira:

# Carisma 18: o guerreiro é extremamente carismático e tem todo o apoio da torcida, recebendo um bônus de +3.

# Carisma 16 e 17: o guerreiro é muito carismático e tem o apoio de quase toda a torcida, recebendo um bônus de +2.

# Carisma 14 e 15: o guerreiro é carismático e tem o apoio de alguns torcedores, recebendo um bônus de +1.

# Carisma 6 e 7: o guerreiro é antipático, e tem alguma torcida contra ele, recebendo uma penalidade de -1.

# Carisma 4 e 5: o guerreiro é muito antipático, e tem quase toda a torcida contra ele, recebendo uma penalidade de -2.

# Carisma 3: o guerreiro é extremamente antipático, e tem toda a torcida contra ele, recebendo uma penalidade de -3.

# Para qualquer outro valor de carisma, a sua função deve retornar 0.

# 5) Escreva um procedimento de nome ataca que recebe dois Guerreiros por passagem de parâmetro por referência e simula um ataque do primeiro guerreiro no segundo. O ataque é dado da seguinte maneira:

# O primeiro guerreiro rola três dados e soma os seus valores com o seu campo ataque e com o seu bônus de carisma. Essa soma é o valor do golpe do primeiro guerreiro.

# O segundo guerreiro rola três dados e soma os seus valores com o seu campo defesa e com o seu bônus de carisma. Essa soma é o valor do escudo do segundo guerreiro.

# Faça dano = golpe - escudo. Se o dano for maior que zero, subtraia dano dos pontos\_vida do segundo guerreiro. Ao subtrair o dano, considere que o campo pontos\_vida não pode ter valores menores que zero.

# 6) Escreva um procedimento de nome jogo que simula a batalha até a morte entre dois guerreiros. Para isso, crie dois guerreiros, um com id\_guerreiro = 1 e outro com id\_guerreiro = 2. Depois, atribua valores aleatórios para os seus campos de batalha a partir da função criaGuerreiro e inicie ataques intercalados entre esses guerreiros, ou seja, comece com o guerreiro 1 atacando o 2, depois o 2 atacando o 1, depois o 1 atacando o 2, e assim por diante. Para simular um ataque, use a função ataca. A batalha deve acabar quando um dos guerreiros, o perdedor, alcançar 0 pontos\_vida. Imprima na tela "\nVencedor: " e logo em seguida o identificador do guerreiro vencedor (exemplo de impressão: "\nVencedor: 1"). Imprima também "\nPontos de vida restantes: " e os pontos de vida restantes do guerreiro vencedor.