Trabalho de Programação I

2022/2023

O “Jogo dos dedos módulo 5” é um jogo para 2 jogadores que se joga com os dedos das duas mãos, normalmente nos recreios das escolas primárias. O jogo começa com os jogadores a mostrar as mãos, com 1 dedo em cada mão. Depois fazem jogadas alternadas com as regras descritas em baixo. Quem ficar sem dedos nas duas mãos perde. Todas as operações são em aritmética módulo 5.

Em cada turno, o jogador pode escolher atacar o adversário ou, em certos casos, igualar os dedos das suas próprias mãos:

- Atacar: O jogador escolhe uma das suas mãos (atacante) e uma das mãos do adversário (atacada). Ambas as mãos, atacante e atacada, têm de ter 1 ou mais dedos. O jogador atacado soma os dedos da sua mão atacada com os dedos da mão atacante (módulo 5) de maneira que a mão atacada fica com o resultado (0 a 4 dedos). A mão atacante não sofre alteração.
- Igualar: Se o jogador tiver 0 dedos numa mão e um número par de dedos na outra, pode dividir igualmente os dedos pelas duas mãos sem realizar ataque. Esta operação conta como uma jogada.

O programa deverá mostrar o estado do jogo, com o número de dedos em cada mão. As jogadas possíveis indicam a mão atacante e a mão atacada, ‘e’ para esquerda e ‘d’ para direita, ou apenas a jogada ‘=’ para dividir igualmente os dedos pelas duas mãos, se essa for uma jogada possível. Um jogador também pode desistir jogando ‘.’ (ponto-final) ao que corresponde uma derrota.

De seguida mostra-se um exemplo de um jogo completo, com o input do jogador humano marcado a vermelho. O símbolo ‘$’ representa a prompt da shell onde se escreve o comando para executar o jogo.

```
$ ./dedos chico-esperto humano

Jogo dos dedos - chico-esperto vs humano


   chico-esperto 1:   1, 1
          humano 2:   1, 1

vez do chico-esperto 1: ee
dedo da esquerda ataca dedo da esquerda, ficam 2 dedos

   chico-esperto 1:   1, 1
          humano 2:   2, 1
```

```
vez do humano 2: EW
jogada inválida!

vez do humano 2: ee
2 dedos da esquerda atacam dedo da esquerda, ficam 3 dedos

   chico-esperto 1:   3, 1
          humano 2:   2, 1

vez do chico-esperto 1: ed
3 dedos da esquerda atacam dedo da direita, ficam 4 dedos

   chico-esperto 1:   3, 1
          humano 2:   2, 4

vez do humano 2: ed
2 dedos da esquerda atacam dedo da direita, ficam 3 dedos

   chico-esperto 1:   3, 3
          humano 2:   2, 4

vez do chico-esperto 1: ee
3 dedos da esquerda atacam 2 dedos da esquerda, ficam 0 dedos

   chico-esperto 1:   3, 3
          humano 2:   0, 4

vez do humano 2: =
divide os 4 dedos pelas duas mãos

   chico-esperto 1:   3, 3
          humano 2:   2, 2

vez do chico-esperto 1: ee
3 dedos da esquerda atacam 2 dedos da esquerda, ficam 0 dedos

   chico-esperto 1:   3, 3
          humano 2:   0, 2

vez do humano 2: dd
2 dedos da direita atacam 3 dedos da direita, ficam 0 dedos

   chico-esperto 1:   3, 0
          humano 2:   0, 2

vez do chico-esperto 1: ed
3 dedos da esquerda atacam 2 dedos da direita, ficam 0 dedos

   chico-esperto 1:   3, 0
          humano 2:   0, 0

vitória do chico-esperto 1!!!
```

O programa deverá receber dois argumentos da linha de comando indicando os dois jogadores. As possibilidades são “humano” e as estratégias “chico-esperto” e “ao-calhas” usadas pelo computador.

As estratégias usadas pelo computador são as seguintes:

- “chico-esperto”: ataca com o maior número de dedos contra o menor número de dedos (caso tenha o mesmo número de dedos em ambas as mãos, é escolhida a mão esquerda). Nunca joga ‘=’.
- “ao-calhas”: constrói a lista de jogadas possíveis e escolhe aleatoriamente uma delas, com igual probabilidade.

Qualquer combinação de 2 jogadores/estratégias é possível. Por exemplo:

```
$ ./dedos humano humano

$ ./dedos chico-esperto humano

$ ./dedos ao-calhas chico-esperto
```

No programa que irá desenvolver, devem ser implementadas funções para:

- verificar se uma jogada é válida sem realizar a jogada (isto é, sem alterar o estado do jogo);
- realizar a jogada do humano;
- escolher uma jogada de acordo com uma estratégia (chico-esperto ou ao-calhas);
- mostrar a frase que sintetiza a jogada realizada (“...dedos da esquerda atacam…”).

Um programa é considerado correto se:

- implementar corretamente as regras do jogo;
- o input e output seja exatamente o especificado no enunciado.

Bonus points 1

Para além das 2 estratégias anteriores, pode opcionalmente implementar outras adicionais:

- Pode sugerir outras estratégias com outros nomes. Por exemplo, uma “meta-estratégia” que consiste em alternar entre as outras estratégias implementadas.
- Estratégia “minimax”: nesta estratégia, são previstas várias jogadas à frente e escolhida a melhor. Para comparar as jogadas é necessário uma função que calcule o valor que cada estado de jogo tem para um jogador. O valor do estado do jogo é avaliado para ambos os jogadores mas com sinal contrário, de maneira que o que é bom para um é mau para o outro. Um jogador tenta maximizar enquanto o outro tenta minimizar. Esta estratégia é mais desafiante de implementar… (pesquisar por “minimax” na internet)

Bonus points 2

Com as regras de jogo definidas anteriormente, o jogo pode terminar em vitória/derrota ou continuar para sempre. Implemente uma funcionalidade do jogo em que se for detectado um estado do jogo que já tenha existido antes, o jogo termina com a mensagem “empate!”. Esta funcionalidade não deve estar ativada por defeito. Para a ativar, deve usar-se uma variável de ambiente EMPATE do seguinte modo:

```
$ EMPATE=1 ./dedos humano humano
```

No programa, o valor da variável pode ser obtido com a função `getenv`:

```
char *v;

v = getenv(“EMPATE”);
```

Se a variável ‘v’ tiver o valor ‘NULL’, então isso significa que a variável de ambiente ‘EMPATE’ não está definida e portanto o jogo não detecta empates. Se a variável ‘v’ for diferente de ‘NULL’, então ‘v’ é uma string com o valor da variável de ambiente definida na shell (no exemplo acima é a string “1”) e o jogo deve detectar empates.

Submissão do trabalho

O trabalho deve ser submetido no moodle até ao final do ano 2022.

Entregue um único ficheiro comprimido com nome “xxxxx-yyyyy.tgz”, em que xxxxx e yyyyy são os números dos alunos por ordem numérica (sem os prefixos ‘L’). Por exemplo “12345-67890.tgz”.

O ficheiro deve conter:

- Código fonte (*.c, *.h),
- Makefile que compile para um executável chamado “dedos”,
- Relatório em formato pdf.

Não inclua o executável. Para comprimir o ficheiro pode usar o seguinte comando no diretório onde tem o trabalho e relatório:

```
$ tar -czvf 12345-67890.tgz *.c *.h *.pdf Makefile
```

O relatório deve descrever as funcionalidades implementadas pelo programa. Deve também listar os protótipos das funções implementadas e uma breve descrição das mesmas (o que fazem/objectivo). O código deve estar comentado e corretamente “indentado”.

Bom trabalho!! 🙂