Trabalho 1 – Memorizando Sequências

Introdução à Ciência da Computação I – 1º semestre 2019

Prof. Rudinei Goularte e Moacir Ponti

Monitores/PAE: Tamires T. S. Barbieri, Henrico Brum, Lucas T. da Silva, André Luis M. Fakhoury e Gabriel Van Loon

Introdução

Nossa memória de curto prazo possui um número limitado de elementos que consegue reter. Segundo um estudo realizado por George Miller, uma pessoa é capaz de se lembrar de 7 itens (+- 2).

Com base nessas informações, o objetivo deste trabalho é desenvolver um jogo em que sequências numéricas são geradas e apresentadas ao usuário para testar sua capacidade de memorização.

Descrição do trabalho

O jogo consiste em gerar uma sequência de números aleatórios entre 0 e 9, exibi-la ao usuário por determinado tempo e, então, solicitar que a reproduza confiando em sua memória.

O programa deve receber como entrada:

- uma **semente** para a geração dos números aleatórios, a qual deve ser passada à função *srand()* e definida apenas uma vez, no início do programa;
- o **nível** do jogador, que pode assumir os valores F (fácil), I (intermediário) ou A (avançado).

O nível do jogador define algumas configurações do jogo, explicadas a seguir.

- **Nível F** (fácil):
  - O programa deve gerar 3 sequências numéricas para o usuário memorizar.
  - Cada sequência deve possuir 5 valores entre 0 e 9.
  - Cada sequência deve ser apresentada por 5 segundos.
- **Nível I** (intermediário):
  - O programa deve gerar 4 sequências numéricas para o usuário memorizar.
  - Cada sequência deve possuir 7 valores entre 0 e 9.
  - Cada sequência deve ser apresentada por 7 segundos.
- **Nível A** (avançado):
  - O programa deve gerar 5 sequências numéricas para o usuário memorizar.
  - Cada sequência deve possuir 9 valores entre 0 e 9.
  - Cada sequência deve ser apresentada por 10 segundos.

Por exemplo, um jogador no nível F, visualizará uma sequência de 5 números por 5 segundos e, em seguida, digitará 5 valores.

O programa deve verificar se os valores fornecidos pelo jogador são iguais aos da sequência apresentada. Em caso positivo, isto é, o jogador memorizou com sucesso os números exibidos, deve-se escrever a mensagem **"Correto!"** e gerar uma nova sequência, até que o limite de sequências definido para cada nível seja atingido. Em caso negativo, isto é, o jogador forneceu um ou mais valores diferentes da sequência apresentada, deve-se exibir a mensagem **"Incorreto!"** e **mostrar novamente a mesma sequência**.

O jogo conta com um **número limitado de chances**. Cada jogador sempre inicia com 3 tentativas disponíveis. A cada erro, desconta-se uma tentativa. O jogo só continua se o usuário ainda possuir tentativas disponíveis. Caso contrário, é encerrado e a mensagem **"Não há mais tentativas disponíveis... Tente novamente!"** é exibida.

Se o usuário conseguir acertar o número máximo de sequências dentro das tentativas disponíveis, deve-se parabenizá-lo com a mensagem: **"Ótima memória! Parabéns!!!"**

Sempre após verificar se o jogador acertou a sequência e exibir as mensagens **"Correto!"** ou **"Incorreto!"** é importante informar ao usuário quantas tentativas ele ainda possui, por meio da mensagem: **"Tentativas disponíveis: <nº>"**

Há ainda um **sistema de pontuação**, calculada da seguinte maneira:

- A pontuação máxima para uma sequência é equivalente a:
  10 x <quantidade de números na sequência>
  Por exemplo, se o jogador acertar todos os valores de uma sequência de tamanho 7, acumula uma pontuação igual a 70.
- A cada valor incorreto digitado, o jogador perde 10 pontos da pontuação máxima. Porém, como já foi explicado, quando um jogador não acerta todos os valores de uma sequência, ela é repetida para que ele tente novamente. Nesse caso, a pontuação obtida na tentativa anterior é considerada a pontuação máxima que o jogador pode alcançar.
  Exemplo 1: na primeira vez que a sequência é exibida o jogador acerta 5 números de um total de 7. Logo, a pontuação máxima para essa sequência passa a ser 50 e a sequência é exibida novamente. Na segunda tentativa, ele acerta todos os valores, assim, continua com 50 pontos.
  Exemplo 2: na primeira vez que a sequência é exibida o jogador acerta 6 números de um total de 9, sendo 60 a pontuação máxima para essa sequência. Na segunda tentativa, ele acerta 7 de 9 números. O desconto é feito sobre a pontuação máxima da sequência (60), sendo a nova pontuação máxima 40 pontos. Na terceira tentativa, ele acerta a sequência completa, acumulando 40 pontos.

  Vale ressaltar que o jogador não deve ficar com pontuação negativa, caso os descontos por erros sejam maiores que a pontuação máxima, deve ser atribuído o valor zero.

Entrada

A entrada do programa é formada por dois valores:
`<semente> <nível>`

Além disso, o jogador deve digitar os números memorizados após a exibição de cada sequência.

Saída

Para cada sequência gerada o programa deve exibir:

```
Sequencia #<nº da sequencia>:
<n1 n2 n3 n4 n5 [n6 n7 n8 n9]>

Incorreto!
Tentativas disponiveis: <nº de tentativas>
Pontuacao: <pontos>
```
ou
```
Correto!
Tentativas disponiveis: <nº de tentativas>
Pontuacao: <pontos>
```
(de acordo com a resposta do jogador)

Se acabarem as tentativas disponíveis, mas o jogador não atingiu o número máximo de sequências, o programa deve exibir:

```
Nao ha mais tentativas disponiveis... Tente novamente!
```

Se o jogador concluir as sequências dentro das tentativas disponíveis, escreve-se:

```
Otima memoria! Parabens!!!
```

Exemplo de entrada/saída

Entrada	Saída
5 F 5 5 4 6 6 3 0 9 5 7 3 0 9 3 7 3 0 9 4 7 3 1 2 0 5	Sequencia #1: 5 5 4 6 6 Correto! Tentativas disponiveis: 3 Pontuacao: 50 Sequencia #2: 3 0 9 4 7 Incorreto! Tentativas disponiveis: 2 Pontuacao: 90 Sequencia #2: 3 0 9 4 7 Incorreto! Tentativas disponiveis: 1 Pontuacao: 80 Sequencia #2: 3 0 9 4 7 Correto! Tentativas disponiveis: 1 Pontuacao: 80 Sequencia #3: 3 1 2 0 5 Correto! Tentativas disponiveis: 1 Pontuacao: 130 Otima memoria! Parabens!!!

Instruções

O código fonte (apenas arquivo .c) deve ser entregue pelo http://run.codes.

IMPORTANTE: para submeter o trabalho ao run.codes comentem as linhas de código referentes à espera em segundos.

O trabalho será avaliado levando em consideração:

1. realização dos objetivos;
2. representação correta da entrada e saída de dados;
3. uso de comentários e estrutura do código (ex.: indentação, legibilidade);
4. número de acertos no sistema run.codes.