Universidade Federal da Bahia - UFBA
Instituto de Matemática - IM
Departamento de Ciência da Computação - DCC
Curso de Bacharelado em Ciência da Computação

MATA40 - Estrutura de Dados e Algoritmos Período: 2020.1 Data: 03/03/2020.

Prof. Antonio L. Apolinário Jr.

Roteiro do Laboratório 0 - Revisão Linguagem C

Objetivos:

- Recordar os conceitos básicos de tipos de dados, variáveis, constantes e estruturas de controle em linguagem C.

Roteiro:

1. Codifique um programa, em linguagem C[1], que implemente um jogo de adivinhação. O usuário deve fornecer, via teclado, um número entre 1 a 99 (valores fora dessa faixa não devem ser aceitos). Em seguida seu programa deve tentar "adivinhar" esse número a partir do sorteio de valores aleatórios[2] dentro dessa faixa, até que acerte o número, ou que alcance um número máximo de tentativas (1.000, por exemplo). Ao final do processo seu programa deve informar se ele conseguiu acertar o número escolhido e quantas tentativas foram feitas.

2. Experimente variar os parâmetros do seu programa: intervalo para adivinhação e número máximo de tentativas. Para isso use **valores constantes** dentro do programa. Avalie o que acontece com essas mudanças.

3. Modifique o seu programa para que o usuário possa definir o número escolhido no comando de chamada do programa[3]. A verificação do intervalo de adivinhação deve continuar a ser feita.

4. Acrescente mais parâmetros na linha de comando. Inclua os limites para a adivinhação e o número máximo de interações como parâmetros que o usuário pode fornecer na chamada do programa. Para isso há duas possibilidades: a mais simples, em que uma ordem é estabelecida, ou a mais complexa, em que os parâmetros podem ser fornecidos em qualquer ordem. Pesquise sobre as duas e tente implementá-las.

---

[1] ou linguagem C++, sem uso de classes ou *templates* STL (*Standard Template Library*).
[2] Pesquise quais as funções da linguagem C que dão suporte a geração de números randômicos e como usá-las.
[3] Presumindo que você está chamando o programa de um terminal de linha de comando, como o **`bash`** do Linux, não dando 2 cliques no executável dentro de uma aplicação de gerenciamento de arquivos, ou rodando dentro de uma IDE, como o ***Code::Blocks***

5. Você deve ter notado que o processo de solução proposto no item 1 é bastante simples e também pouco eficiente. Caso seja permitido que o programa faça perguntas ao usuário sobre o número escolhido como palpite, o processo de adivinhação pode se tornar mais eficiente. Se a cada palpite o usuário informar se o número escolhido é maior ou menor que o palpite, como seu algoritmo poderia se beneficiar desta informação a mais?
   Implemente uma nova versão em que seu programa "simule" essas perguntas ao usuário, ou seja, compare o palpite ao número escolhido, e passe a gerar novos palpites em função da resposta obtida.

6. Compare os resultados obtidos no programa do item 1 e os obtidos no programa do item 5. Tente relacionar o número de palpites de uma implementação e da outra, variando de forma igual o intervalo de adivinhação e o limite de palpites. É possível construir alguma relação entre os dois resultados?