Universidade Federal de São Carlos
Bacharelado em Ciência da Computação
Algoritmos e Programação I
Prof. Tiago A. Almeida <talmeida@ufscar.br>
Profa. Tiemi C. Sakata <tiemi@ufscar.br>

LISTA 03
COMANDOS DE REPETIÇÃO I: WHILE E DO-WHILE

- **Prazo para entrega: 22/04/2018 – 23:55:00**
- **Atenção:**

  1. **Arquivo:** o nome do arquivo referente ao código-fonte deverá seguir o seguinte padrão: <número do RA>**_L**<número da lista>**EX**<número do exercício>.c. Exemplo: `123456_L03EX01.c`;
  2. **E/S:** tanto a entrada quanto a saída de dados devem ser "secas", ou seja, não devem apresentar frases explicativas. Siga o modelo fornecido e apenas complete as partes informadas.
  3. **Identificadores de variáveis:** escolha nomes apropriados;
  4. **Documentação:** inclua comentários e indentação no programa.

- **Exercícios**

  1. Você está ficando ainda melhor como programador, cada aula de Introdução a Programação abre uma porta nova para tornar seus algoritmos cada vez mais rebuscados. Então, você resolve refazer o exercício 01 da lista anterior, dessa vez seu programa só encerrará quando a opção digitada for 0, e você decidiu incrementar sua calculadora colocando a opção de potência.

     Seu programa receberá dois **números inteiros**, e depois a opção correspondente a cada operação, apoś o resultado ser impresso na tela, você deverá receber a opção novamente (até que a opção seja 0). As opções são enumeradas do seguinte modo:

     0. sair do programa;
     1. soma dos dois números;
     2. subtração do primeiro pelo segundo;
     3. subtração do segundo pelo primeiro;
     4. multiplicação dos dois números;
     5. divisão do primeiro pelo segundo;
     6. divisão do segundo pelo primeiro;

7. quociente inteiro da divisão do primeiro pelo segundo;
8. quociente inteiro da divisão do segundo pelo primeiro;
9. resto da divisão do primeiro pelo segundo;
10. resto da divisão do segundo pelo primeiro;
11. primeiro número elevado pelo segundo (utilize o resultado com long int);
12. segundo número elevado pelo primeiro, (utilize o resultado com long int) ;

Complete o arquivo `L02EX01.c`

Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	**Saída**
`3` (`Primeiro número`)	
`2` (`Segundo número`)	
`1` (`opção de soma`)	
	`5` (`resultado da soma`)
`5` (`opção de divisão`)	
	`1.5` (`resultado da divisão`)
`11` (`opção de potência`)	
	`9` (`resultado da potência`)
`0` (`sair do programa`)	
`7` (`Primeiro número`)	
`9` (`Segundo número`)	
`15` (`Opção de inválida`)	
	`Opção Inválida!`
`4` (`Opção de multiplicação`)	
	`63` (`resultado da multiplicação`)
`0` ((`sair do programa`))	

Detalhes

(a) Utilize um long int como resultado das opções 11 e 12;

(b) Utilize a mensagem pré definida de Opção Inválida! quando a opção selecionada não estiver no intervalo de 0 a 12 e na tentativa de divisão por 0.

2. A conjectura de Collatz afirma que todo número natural, quando aplicado a esta conjectura, formará uma sequência e o último número será sempre igual à 1. Faça um programa que imprima a sequência obtida a partir de um **número natural** n, tal que $0 < n < 1.000$, cuja sequência respeite a seguinte função:

$$f(n) = \begin{cases} n/2, & \text{se } n \text{ par} \\ 3n+1, & \text{se } n \text{ ímpar} \end{cases}$$

O programa só deverá ser finalizado quando o usuário digitar o número zero.

O primeiro número da sequência sempre deverá ser o número n informado pelo usuário. Após cada valor da sequência, imprima um espaço em branco (␣). Após a impressão da sequência, insira uma quebra de linha (¶).

Complete o arquivo `L03EX02.c`

Exemplo de E/S:

Entrada	**Saída**
5	
	5 16 8 4 2 1␣¶
12	
	12 6 3 10 5 16 8 4 2 1␣¶
0	

3. Pelo calendário gregoriano, instituído em 1582 pelo Papa Gregório XIII, os anos bissextos são aqueles cujos valores são divisíveis por 4, exceto os anos que são divisíveis por 100 e não por 400. Por exemplo, os anos 1600 e 2000 são anos bissextos enquanto que os anos 1700, 1800 e 1900 não são. Sabendo disso, faça um programa que imprima na tela todos os anos bissextos contidos em um intervalo informado pelo usuário e o número de anos bissextos desse mesmo intervalo.

Complete o arquivo `L03EX02.c`

Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos).

Entrada	**Saída**
1980 (Primeiro ano do intervalo)	
2000 (Segundo ano do intervalo)	
	1980 1984 1988 1992 1996 2000␣¶
	6 (Quantidade de anos bissextos)
1950 (Primeiro ano do intervalo)	
1980 (Segundo ano do intervalo)	
	1952 1956 1960 1964 1968 1972 1976 1980␣¶
	8 (Quantidade de anos bissextos)
0 (Primeiro ano do intervalo)	
0 (Segundo ano do intervalo)	

Detalhes

(a) Nos casos de testes, as entradas numéricas inteiras são sempre informadas corretamente e, portanto, não é necessário fazer qualquer tipo de tratamento para essas variáveis.

(b) Deve-se colocar um espaço entre cada ano bissexto, e pular uma linha entre a lista com os anos bissextos e a quantidade de anos bissextos.

(c) O programa deve solicitar uma nova entrada de dados até que o usuário digite o intervalo "0 0", então o programa é finalizado.

(d) O programa deve mostrar uma mensagem de erro caso o usuário entre com um intervalo inválido (primeiro ano maior do que o segundo ano).

4. Heron de Alexandria foi um matemático de destaque que desenvolveu vários algoritmos e fórmulas para simplificar cálculos matemáticos. Uma de suas maiores contribuições é o Método de Heron para o cálculo de raiz quadrada. Tomando o valor inicial de k como 1, gera-se um novo valor de k de acordo com a fórmula:

$$k_{i+1} = \frac{k_i + \frac{n}{k_i}}{2}$$

À medida que o processo se repete, os novos valores de k se aproximam cada vez mais da raiz quadrada de n. Faça um programa que receba dois valores inteiros e imprima as raízes quadradas de todos os números inteiros do primeiro até o segundo usando o Método de Heron.

Complete o arquivo `L03EX03.c`

Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	**Saída**
`10 12`	
	`3.16` (`raiz de 10`)
	`3.32` (`raiz de 11`)
	`3.46` (`raiz de 12`)
`9 5`	
	`2.24` (`raiz de 5`)
	`2.45` (`raiz de 6`)
	`2.65` (`raiz de 7`)
	`2.83` (`raiz de 8`)
	`3.00` (`raiz de 9`)
`15 15`	
	`3.87` (`raiz de 15`)

Detalhes

(a) Use 10 repetições da fórmula de Heron para alcançar um valor suficientemente preciso.

(b) Caso o segundo número seja maior que o primeiro, a entrada será considerada válida, porém a impressão das raízes deverá ser feita do menor valor até o maior.

(c) Imprima uma raiz por linha e utilize duas casas decimais para apresentar o valor.

5. Você, em seu tempo livre, decide se divertir com um jogo de uma renomada série que além do aspecto aventura, apresenta uma série de puzzles para resolver. Em um dos desafios para conseguir um item opcional (porém extremamente necessário à sua coleção de conquistas pessoais), a sua missão é simples, ao receber um número você deve inverter a ordem de seus algarismos. Por exemplo, se você receber "123" deve responder "321".

   No entanto, como é necessário realizar esse feito 50 vezes seguidas, sem errar, com temporizador, você, caro programador, decide fazer um programa que possa fazer essa conferência em tempo e a fim de evitar um erro que levará a outra cansativa tentativa.

   Seu programa deve iniciar uma sessão e imprimir uma mensagem de iniciação. A partir daí, deve receber um número (N 0 <= N <= 9.000.000.000.000.000.000) e então imprimir a sua versão com números invertidos linha por linha. Seu programa deve encerrar ao receber o número -1 e, então, imprimir uma mensagem de encerramento.
   Para trabalhar com números bem maiores, utilize variável do tipo **long long int**.

   Complete o arquivo `L03EX05.c`

   Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	**Saída**
	`Iniciando a sessão...`
`456` (n)	
	`654` (`Algarismos invertidos`)
`9382164` (n)	
	`4612839` (`Algarismos invertidos`)
`-1`	
	`Encerrando a sessão...`

   Detalhes

   (a) Ao iniciar e encerrar o programa, não se esqueça de usar mensagens já pré-definidas em seu código.

   (b) A saída de um número invertido deverá ser realizada seguida de uma quebra de linha (`\n`).

   (c) Atente-se que a opção de saída (-1) não gera uma linha vazia com quebra de linha (`\n`) (Basta olhar o exemplo)

6. Astheobaldo cansado de tentar ficar monstrão, desiste da academia e resolve se dedicar aos estudos. Enquanto estudava matemática, ele se deparou com o **Teorema Fundamental da Aritmética** que afirma que todos os números inteiros positivos maiores que 1 podem ser decompostos num produto de números primos, sendo esta decomposição única a menos de permutações dos fatores.

Astheobaldo ficou intrigado com este Teorema, ele insiste em acreditar que isso não é válido. Você, astuto, resolve fazer um programa em C, para mostrar ao seu querido amigo o quanto ele está errado.

Seu programa deve receber um inteiro (int) positivo maior que 1, e imprimir na tela a decomposição do número em fatores primos. Após cada valor da sequência, imprima um espaço em branco (␣). Após a impressão do último, insira uma quebra de linha (¶).

Complete o arquivo `L03EX06.c`

Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	**Saída**
`1239` (`número`)	
	`3 7 59¶` (`fatoração em primos`)
`45` (`número`)	
	`3 3 5¶` (`fatoração em primos`)

Detalhes

(a) Não é necessário fazer qualquer verificação sobre a entrada;

(b) Após cada número da sequência há um espaço em branco, exceto o último número, que deve ter uma quebra de linha.

- **Cuidados**

1. **Erros de compilação:** nota **zero** no exercício
2. **Tentativa de fraude:** nota **zero na média** para todos os envolvidos.