Instituto Federal de Educação Ciência e Tecnologia do RN - IFRN

Exercícios

001 Escreva um programa que leia um número inteiro n, verifique se o mesmo é positivo e mostre todos os números de 0 até n, um número por linha. Caso o número não seja positivo o programa deve mostrar a palavra `NEGATIVO`, em uma única linha.

002 Escreva um programa que leia um número inteiro n e mostre todos os números múltiplos de n de 0 até 1.000. Cada número deve ser mostrado em uma linha. Nada mais deve ser mostrado.

003 Escreva um programa que leia um número inteiro n e mostre todos os múltiplos de 3 maiores ou iguais a 0 e menores ou iguais a n.

004 Escreva um programa que leia dois números n e m e mostre todos os números de n até 1 que são múltiplos de m, em ordem decrescente.

005 Escreva um programa que leia dois números inteiros n e m e mostre a soma dos números de 0 até n que sejam múltiplos de m. Se o n for negativo o programa deve mostrar 0.

006 Escreva um programa que leia um número inteiro n e calcule o seguinte somatório:

$$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n-1} + \frac{1}{n}$$

O programa deve mostrar o valor do somatório em uma única linha, com 4 casas decimais.

007 Escreva um programa que leia um número inteiro n e mostre os números de 0 até n que são múltiplos de 3 ou de 5, um número por linha em ordem crescente. Se o n for negativo o programa deve mostrar a palavra `Negativo` e nada mais.

008 Escreva um programa que leia dois números inteiros digitados pelo usuário e imprima a soma de todos os números existentes entre eles incluindo eles mesmos. O programa só deve mostrar a soma, nada mais. Considere que o primeiro pode ser menor que o segundo e vice-versa.

009 Escreva um programa que leia três números inteiros a, r e n da entrada e que imprima os n primeiros termos de uma P.A.. Considere que a é o primeiro termo e r a razão desta P.A. Cada número deve ser impresso em uma linha.

010 Escreva um programa que leia um número inteiro n e depois mais n números inteiros, armazenando-os em um *array*. O programa deve então ler um número inteiro e contar quantas vezes o número aparece no *array* e em quais índices.

Exemplo: Considere um *array* de 10 elementos `[1,4,2,5,3,4,6,4,1,2]` e o número a ser procurado sendo o 4. O programa deve mostrar, em uma linha, o número 3, já que o 4 aparece 3 vezes, e nas linhas subsequentes os índices em que ele aparece. A saída do programa fica como abaixo:

```
3
1
5
7
```

011 Escreva um programa que leia um número inteiro positivo n e calcule o seguinte somatório:

$$\frac{1}{1} - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} - \frac{1}{6} + \cdots + \frac{1}{n}$$

O programa deve mostrar o valor do somatório em uma única linha, com 4 casas decimais.

012 Escreva um programa que leia um número natural n e mostre todos os divisores de n, começando com o 1 e terminando com n, em ordem crescente.

013 Escreva um programa que leia um número natural n e informe se ele é primo. Se n for primo o programa escreve o número seguido da palavra `primo`, como no exemplo abaixo:

```
11 primo
```

se o número não for primo o programa mostra o número seguido da palavra `composto` e abaixo a lista de divisores, com exceção do 1 e dele mesmo, como no exemplo a abaixo:

```
10 composto
2
5
```

014 Escreva um programa que leia dois números inteiros e mostre o MDC (Máximo Divisor Comum) dos dois números. O programa só deve mostrar o MDC e nada mais. O programa deve verificar se os dois números são inteiros positivos e, caso contrário, o programa deve mostrar apenas o número 0 na saída.

015 Escreva um programa que leia um número n e calcule o n-ésimo número de Fibonacci

(http://pt.wikipedia.org/wiki/Número_de_Fibonacci). Um número de Fibonacci é definido através de uma sequencia, onde cada número é a soma dos dois anteriores. Os dois primeiros números da sequencia são o 1 e o 1. O terceiro número é obtido a partir da soma entre 1 e 1, sendo 2. O quarto termo é a soma de 1 e 2, que é 3. Abaixo segue uma tabela que mostra os 8 primeiros números da sequencia.

Seq	Num
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21

O programa deve, para uma entrada `Seq`, mostrar `Num`. Considere que a o número lido sempre será um inteiro maior do que 0.

016 Escreva um programa que leia dois números inteiros e mostre o MMC (Mínimo Múltiplo Comum) dos números. Nada mais deve ser mostrado pelo programa.

017 Um determinado material radioativo perde metade de sua massa a cada x segundos. Escreva um programa que leia o tempo x de perda de massa e a massa inicial, em gramas, de um material. O programa deve então calcular em quanto tempo a massa do material fica menor do que $0,5$ gramas. O programa deve mostrar a massa final, com 3 casas decimais em uma linha e o tempo gasto para o material chegar a essa massa. O tempo deve ser mostrado na forma `HH:MM:SS`.

018 Um determinado material radioativo perde metade de sua massa a cada 55 segundos, se massa for maior do que um determinado valor x. Se a massa for menor do que x, o material perde $1/4$ de sua massa a cada 40 segundos. Escreva um programa que leia a massa inicial, em gramas de um determinado material e o valor x em que o fator de perda muda da metade para um quarto e calcule em quanto tempo a massa do material fica menor do que 1. O programa deve mostrar o tempo total, em segundos, considereando apenas a parte inteira, em que a mass do material fica menor do que 1 grama.

019 Escreva um programa que leia um número inteiro e realize a sua decomposição em números primos. O programa deve mostrar, em uma única linha a forma decomposta desse número.

Exemplo: Considere o número 100, cuja forma decomposta é $2^2 \times 5^2$, que é o mesmo que $2 \times 2 \times 5 \times 5$. O programa deve mostrar em uma única linha:

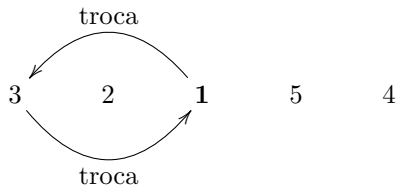
```
2x2x5x5
```

020 Um técnica simples de ordenação de números em um *array*, apesar de ineficiente, é conhecida como método da seleção.

A técnica consiste em procurar em um *array* o menor elemento e troca-lo de posição com o elemento localizado no índice 0. Com isso há uma garantia que o menor elemento encontra-se já no seu lugar, considerando uma ordenação crescente. O próximo passo é encontrar o menor elemento entre os que estão entre os índice 1 e $n-1$ e efetuar a troca com o elemento do índice 1. Agora temos que os dois primeiros elementos estão nos seus devidos lugares. Repetimos a operação para colocar o elemento menor entre os restantes no índice 2, e assim sucessivamente para todos os índices do *array*. Ao final teremos o *array* ordenado.

Exemplo: considere o *aray* `[3,2,1,5,4]`

O menor elemento é o número 1, localizado no índice 2 do *array*. Uma vez identificado o elemento e seu índice, realizamos uma troca entre o elemento de índice 0 e o menor elemento, no índice 2.



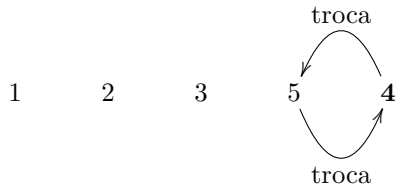
Após a troca, teremos uma nova ordem de elemento, sabendo que o menor de todos está no primeiro índice (índice 0) do *array*.

1 2 3 5 4

A técnica considera agora apenas os índices após o 0. O menor elemento localizado entre os índices restantes (1..4) é o 2, que já está localizado no índice 1. O mesmo acontece com o índice 2, que contém o menor elemento, o número 3, entre os índices 2..4.

1 **2** 3 5 4

O menor elemento localizado entre os índices 3 e 4 está localizado no índice 4, sendo necessario efetuar a troca.



Após essa operação, o *array* restante contém apenas um elemento, que está no índice 4 do nosso *array*. Sendo este o maior elemento. O *array* está ordenado.

Escreva um programa que implemente esta técnica de ordenação e mostre todas as trocas realizadas durante o processamento. O programa deve ler um número inteiro com a quantidade de elementos a ordenari. Depois lê n números inteiros e ordena-os usando esta técnoca. A cada necessidade de troca o programa deve mostrar os índices a serem trocados e os elementos contidos nos índices. Ao final o programa mostra o *array* ordenado.

Considerando um *array* de 5 elementos $a = [3, 2, 1, 5, 4]$ o programa deve mostrar a seguinte saída:

```
0,2 (3,1)
3,4 (5,4)
1
2
3
4
5
```