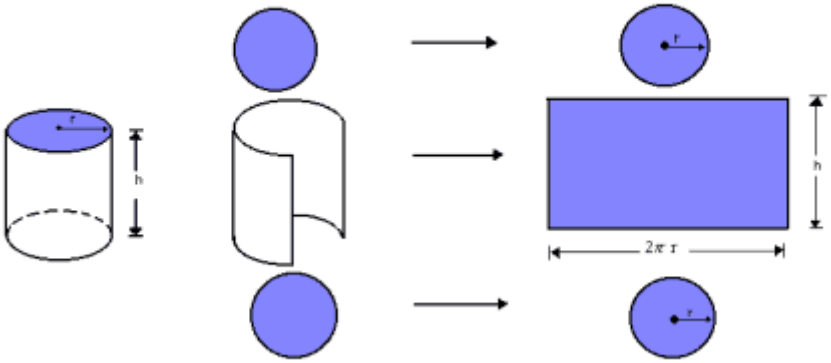
Universidade Federal do Ceará
Laboratório de Programação
Prof. Rafael Ivo

Lista de exercícios 1

Obs: Todas as questões devem ser resolvidas utilizando a linguagem C padrão ANSI.

1) Escreva um programa que leia a altura e o raio da base de um cilindro circular reto e escreva as seguintes informações: área lateral, área da base e volume.
Obs: Defina o valor de π = 3,14159265359 como uma constante declarada fora da função **`main`**.



$$Comprimento\,da\,circunferência = 2\pi \times raio$$
$$Área\,da\,circunferência = \pi \times raio^2$$
$$Volume\,do\,cilindro = Área\,da\,base \times Altura$$

2) Escreva um programa que leia os coeficientes A, B e C de uma equação $Ax^2+Bx+C=0$ e calcule o valor do discriminante delta e as raízes da equação.

3) Escreva um programa que leia os coeficientes do sistema de equações lineares mostrado abaixo e retorne os valores de x e y que satisfazem ambas as equações do sistema.

$$\begin{cases} Ax+By=C \\ Dx+Ey=Z \end{cases}$$

4) Utilize o comando **`sizeof`** para exibir o tamanho em bytes de todos os tipos e suas variações implementados pela linguagem C.

5) Crie uma função gerador de senha. O papel dessa função é retornar um número inteiro cada vez que é chamada, em que o número é igual ao sucessor do número retornado pela última vez. ***Obs:*** Não é permitido utilizar variáveis globais.

6) Sem utilizar a biblioteca **`string.h`**, crie uma função que receba duas strings A e B e retorne os seguintes valores:

- número 1 → caso a string A deva vir depois da string B em uma ordem alfabética
- número -1 → caso a string A deva vir antes da string B em ordem alfabética
- número 0 → caso a string A seja igual a string B

Obs: Caso uma string seja um subconjunto da outra como: A = "Rafael" e B = "Rafael Ivo", a string menor deve vir antes.

7) Ao criar uma interface gráfica de janelas de um sistema operacional, um programador decide armazenar 8 opções de preferência do usuário. Essas preferências são opções de ligado / desligado.

1. Sombra
2. Transparência
3. Borda
4. Encaixar borda na tela
5. Animação ao mover
6. Animação ao minimizar
7. Animação ao fechar
8. Foco automático

Esse programador define que todas as opções devem estar ligadas por padrão, mas cria funções para ligar e desligar cada opção. Construa um programa com duas funções:

- Função que liga ou desliga uma determinada opção
- Função que exibe as opções ligadas em um determinado momento

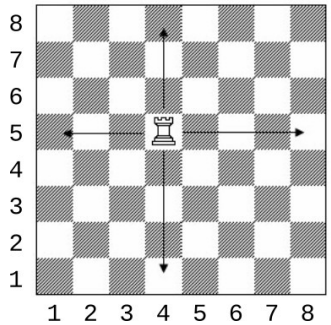
A papel da função **`main`** é apenas ficar em um laço redirecionando para as demais funções até que o usuário deseje terminar o programa.
Obs:

- Utilize apenas 1 byte de memória para armazenar todas as opções.
- Utilize constantes através do comando **`enum`** para acessar as opções.

8) Construa uma função que receba uma string e retorne quantas vogais há na string. Ex: "Universidade" → 6 vogais.
Obs: Construa sua função de duas formas diferentes: usando a estrutura **`if-else-if`** e usando o comando **`switch`**.

9) Escreva uma função que escreva uma string em ordem reversa. Ex: "linguagem" → "megaugnil". Obs: Não é permitido usar a biblioteca **`string.h`**.

10) No xadrez, a torre se move em linha reta horizontalmente e verticalmente pelo número de casas não ocupadas, até atingir o final do tabuleiro ou ser bloqueado por outra peça, como mostrado na figura ao lado. Construa um programa que receba as coordenadas de uma torre em um ***tabuleiro vazio*** e retorne a lista das posições onde a torre pode se mover. Por exemplo, no caso da figura:




```
Coordenada inicial: (4,5)
Posições possíveis:
(4,1),(4,2),(4,3),(4,4),(4,6),(4,7),(4,8),(1,5),(2,5),(3,5),(5,5),(6,5),(7,5),(8,5)
```

11) Escreva um programa que leia um número inteiro N positivo e desenhe formas geométricas usando asteriscos, como mostrados abaixo para o caso de N = 6.

`*`
`**`
`***`
`****`
`*****`
`******`
	`******`
	`*****`
	`****`
	`***`
	`**`
	`*`
		`*`
		` *`
		`  *`
		`   *`
		`    *`
		`     *`

`     *`
`    *`
`   *`
`  *`
` *`
`*`
	`******`
	` *****`
	`  ****`
	`   ***`
	`    **`
	`     *`
		`     *`
		`    **`
		`   ***`
		`  ****`
		` *****`
		`******`

`* * * * * *`
` * * * * * *`
`  * * * * * *`
`   * * * * * *`
`    * * * * * *`
`     * * * * * *`
	`* * * * * *`
	` * * * * *`
	`  * * * *`
	`   * * *`
	`    * *`
	`     *`
		`*          *`
		` *        *`
		`  *      *`
		`   *    *`
		`    * *`
		`     *`

12) Estudiosos já formularam diversas séries matemáticas que, se levadas ao infinito, podem calcular π de forma precisa em inúmeras casas decimais. Algumas delas são tão complexas que só conseguem ser analisadas por supercomputadores. A de Gregory-Leibniz, por sua vez, é uma das mais simples. Para isso, usa-se a seguinte fórmula:

$$\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \frac{4}{13} - \frac{4}{15} \ \cdots$$

Somas e subtrações são alternadas com frações de numerador 4 e denominador com números ímpares em sequência. Quanto mais longe for, mais perto este resultado será de π. Escreva um programa que leia do usuário uma quantidade positiva N de parcelas da sequência de Gregory-Leibniz e retorne a aproximação do valor de π.

13) Escreva um programa para determinar se um determinado número é primo ou não.

14) Escreva um programa que cheque se um número inteiro positivo é perfeito. O número é dito perfeito quando ele é igual a soma dos seus divisores.
Ex:

```
Número: 56
Divisores: 1, 2, 4, 7, 8, 14, 28
Soma dos divisores: 64
Conclusão: Número 56 não é perfeito.
```

15) Escreva um programa que leia em uma variável int um número binário e o converta para um número decimal.
Ex:

```
Número binário: 1010100
Número decimal: 84
```

Obs: Calcule o resultado usando o laço **`for`**.