UTFPR UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ **Câmpus Pato Branco**

Disciplina: **Programação Orientada a Objetos**
Turma: **Engenharia de Computação - 4^o Período**
Prof^a. Luciene de Oliveira Marin
lucienemarin@utfpr.edu.br

1^a Lista de exercícios - Revisão da Linguagem C

Estruturas de repetição

1. Sabe-se que o número Neperiano `e = 2.7182818...` (que é um número irracional) pode ser calculado pela soma dos valores de uma série infinita, como mostrado abaixo:

$$e = \sum_{n=0}^{\infty} \frac{1}{n!} = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} \ldots +$$

   Fazer um programa em linguagem C que calcule este número `(e)` considerando apenas as 15 (quinze) primeiras parcelas.

Estruturas de repetição e decisão

2. Faça um programa em linguagem C para informar quantos algarismos possui qualquer número inteiro digitado pelo usuário. Além disso, o programa também deve informar quantos algarismos são múltiplos de 2 e quantos algarismos são múltiplos de 3. Por exemplo:

```
Número Informado pelo usuário:  25648
Resultado Informado pelo programa:  5 algarismo(s)
Quantidade de algarismo(s) múltiplo(s) de 2:  4
Quantidade de algarismo(s) múltiplo(s) de 3:  1
```

3. Faça um programa que lê um número inteiro e positivo n e em seguida produz e mostra uma matriz $M_{n \times n}$ semelhante à vista abaixo.

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ -1 & 0 & 1 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & -1 & 0 & 1 \\ -1 & -1 & -1 & -1 & 0 \end{pmatrix}$$

4. Uma das maneiras de se conseguir a raiz quadrada de um número é subtrair deste os ímpares consecutivos a partir de 1, até que o resultado da subtração seja menor ou igual a zero. 0 número de vezes que forem realizadas as subtrações consecutivas é a raiz quadrada exata do número (resultado 0) ou aproximada do número (resultado negativo). Por exemplo:

```
a)
Raiz de 16 = 4
16 - 1 = 15
15 - 3 = 12
12 - 5 = 7
7 - 7 = 0
Total de 4 subtrações.
```

```
b)
Raiz de 24 ≅ 5
24 - 1 = 23
23 - 3 = 20
20 - 5 = 15
15 - 7 = 8
8 - 9 = -1
Total de 5 subtrações.
```

Faça um programa em linguagem C que dado um número inteiro mostre sua raiz quadrada e se essa raiz é exata ou aproximada, isto é, para os exemplos acima, as saídas deverão ser da seguinte forma, respectivamente:

```
a) Raiz quadrada exata = 4
b) Raiz quadrada aproximada = 5
```

Vetores - strings

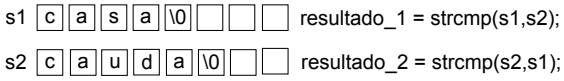
5. Faça um programa que leia uma frase qualquer que apresente apenas caracteres alfanuméricos. No final, o programa deve exibir a quantidade de vezes que cada caracter aparece na frase. Obs.: você pode estabelecer qualquer caracter numérico para sinalizar os caracteres alfanuméricos que já foram contados durante o processo.

   Exemplo:

   entrada: Batatinha quando nasce se esparrama pelo chao.
   saída: b ocorre 1 vez(es)
   a ocorre 9 vez(es)
   t ocorre 2 vez(es)
   i ocorre 1 vez(es)
   n ocorre 3 vez(es)
   h ocorre 2 vez(es)
   q ocorre 1 vez(es)
   u ocorre 1 vez(es)
   d ocorre 1 vez(es)
   o ocorre 3 vez(es)
   s ocorre 3 vez(es)
   c ocorre 2 vez(es)
   e ocorre 4 vez(es)
   p ocorre 2 vez(es)
   i ocorre 1 vez(es)
   r ocorre 2 vez(es)
   m ocorre 1 vez(es)
   . ocorre 1 vez(es)

6. Na biblioteca `<string.h>` da linguagem C existe uma função chamada `strcmp(s1,s2)` que compara duas *strings* `s1` e `s2`. Se as duas strings possuem exatamente o mesmo conteúdo a função `strcmp` retorna 0, se as strings forem diferentes a função pode retornar 1 ou -1 segundo as seguintes regras. Retorna:

$$\begin{cases} 1, & \text{se o segundo parâmetro (no exemplo } \texttt{s2}\text{) é menor do que primeiro parâmetro} \\ -1, & \text{se o segundo parâmetro é maior do que o primeiro parâmetro;} \\ 0, & \text{se o primeiro e o segundo parâmetro forem iguais} \end{cases}$$



Escreva um programa que faça a mesma coisa que a função `strcmp(s1,s2)` faz. Duas strings devem ser lidas, mas por simplicidade, deverá ser impresso na tela apenas uma mensagem informando se as strings são iguais ou diferentes.

Funções - strings

7. Escreva uma função de protótipo

   ```
   void strins(char s[], char ch, int pos);
   ```

   que insira o caracter **ch** na posição **pos** da string **s**.

8. Escreva uma função de protótipo

   ```
   void strinss(char s1[], char s2[], int pos);
   ```

   que insira a string **s2** em **s1** na posição **pos**. Utilize a função do exercício anterior.

9. Escreva uma função de protótipo

   ```
   void left(char origem[], char dest[], int n);
   ```

   que copie os **n** primeiros caracteres da string **origem** na string **dest**. Utilize a função `strncpy()`.

10. Escreva uma função de protótipo

    ```
    void right(char origem[], char dest[], int n);
    ```

    que copie os **n** últimos caracteres da string **origem** na string **dest**. Utilize a função `strncpy()`.