Centro Universitário Serra dos Órgãos - UNIFESO
Direção Acadêmica das Ciências Humanas e Tecnológicas - DACHT
Curso de Bacharelado em Ciência da Computação
Disciplina: Laboratório de Fundamentos em TIC
Professor: Gabriel Resende Machado

LISTA DE EXERCÍCIOS 1 - PARTE DA AV1

Valor: 30 pontos

Instruções para entrega (necessário seguir à risca):

- Para cada exercício, será necessário produzir dois arquivos: **(1) fluxograma da solução** e **(2) arquivo de extensão .c** referente ao código-fonte do programa escrito em C;

- Para elaborar o fluxograma, utilize a plataforma online *draw.io*. Para salvar o fluxograma em um arquivo, pressione CTRL + SHIFT + A, vá em *Ficheiro*, *Exportar Como*, *JPEG*. Na caixa de diálogo, marcar o *checkbox Selecionar Somente* e, em *Appearance*, marcar *Light*. Em seguida, clique em *Exportar*, nomeie o arquivo como ***fluxograma_exercicio_1.jpg*** (substituir o numeral '1' pelo número correspondente do exercício), e em *Where,* selecione a opção *Descarregar,* para salvar o fluxograma no seu computador;

- Os arquivos de código fonte **.c** devem ser salvos como ***exercicio_1.c*** (substituir o numeral '1' pelo número correspondente do exercício);

- Reunir os arquivos .jpg (fluxogramas) e .c (códigos fonte), **referentes a cada exercício**, em pastas distintas. Cada pasta terá o nome ***solucoes_exercicio_1*** (substituir o numeral '1' pelo número correspondente do exercício). Como são 12 exercícios, haverá 12 pastas contendo ambos os arquivos .c e .jpg;

- Para realizar a entrega, compacte as pastas em formato .zip, juntamente com um arquivo chamado ***integrantes.txt***. Neste arquivo, devem constar os nomes completos dos integrantes do grupo, juntamente com suas respectivas matrículas. Enviar o arquivo zipado via plataforma do Canvas. **Se atentar ao código da turma para envio:**

    - **FLEXCOMP1.8:** Turma do Campus Quinta do Paraíso;

    - **FLEXCOMP01AB:** Turma do Campus Sede.

- **O trabalho deve ser feito em grupo de, no máximo, 3 pessoas;**

- **Prazo de entrega: impreterivelmente 06/05/2023.**

1) Desenvolva um programa em C que leia do terminal um valor de ponto flutuante em graus Celsius (ºC, ex.: 25.5), e converta esse valor de entrada para as escalas de temperatura Fahrenheit (ºF) e Kelvin (K). O programa deve exibir o valor original lido, como também os valores convertidos, com precisão de duas casas decimais, da seguinte forma:

```
25.50 graus Celsius é igual a:
77.90 graus Fahrenheit;
298.65 Kelvin.
```

2) Desenvolva um programa em C que apresente ao usuário um menu com as seguintes opções:

```
Digite uma das opcoes abaixo:
1- Transformar KPH para MPH;
2- Transformar MPH para KPH.
```

Caso o usuário informe a Opção **1** (como valor inteiro), um valor de ponto flutuante representando uma medida em quilômetros por hora (KPH) deverá ser lido e convertido para milhas por hora (MPH). Caso a Opção **2** seja escolhida, um valor de ponto flutuante representando uma medida em milhas por hora (MPH) deverá ser lido e convertido para quilômetros por hora (KPH). Caso nenhuma das opções disponíveis seja fornecida pelo usuário, o programa deverá encerrar. Exiba os valores lidos e convertidos seguindo a orientação do Exercício 1.

3) Desenvolva um programa em C que permita ao usuário informar um número inteiro N > 0. Em seguida, o usuário deverá digitar N números inteiros, que deverão ser armazenados em um *array*. Por fim, o programa deverá ordenar o *array* em ordem crescente e exibi-lo ao usuário. Os números deverão ser exibidos em uma mesma linha, com a separação de um espaço entre eles.

4) Desenvolva um programa em C que permita ao usuário informar um número inteiro N > 0. Em seguida, o usuário deverá digitar N números inteiros, que deverão ser armazenados em um *array*. Após, o programa deverá calcular a mediana dos valores do *array*. Para calcular corretamente a mediana, é preciso verificar se (i) o *array* está ordenado de forma crescente (utilize a solução do Exercício 3); e (ii) se N é par ou ímpar. Ao final, exiba uma mensagem no terminal informando a mediana referente ao *array* digitado pelo usuário.

5) Desenvolva um programa em C que permita ao usuário informar um número inteiro N > 0. Em seguida, o usuário deverá digitar N números inteiros, que deverão ser armazenados em um *array*. A seguir, o programa deverá contar a ocorrência de cada

elemento presente no *array*. Como exemplo, tome a seguinte entrada: `N = 10, array = [2, 3, 5, 4, 2, 7, 2, 1, 0, 3]`. O programa deverá exibir a seguinte saída:

```
Ocorrencias de cada elemento de array:

2: 3 ocorrencias
3: 2 ocorrencias
5: 1 ocorrencias
4: 1 ocorrencias
7: 1 ocorrencias
1: 1 ocorrencias
0: 1 ocorrencias
```

6) Desenvolva um programa em C que permita ao usuário informar um número inteiro N >= 0. Em seguida, transforme o número N em binário. **Dica 1:** O programa deve ser capaz de converter qualquer número inteiro do tipo `int` (4 bytes). Por isso, declare um *array* com 32 posições. **Dica 2**: a solução baseada em recursão é mais simples. Por fim, exiba uma mensagem de saída para o usuário seguindo o formato a seguir, tomando como exemplo N = 20. Após a exibição da mensagem, o programa deverá solicitar novamente que o usuário digite um novo valor para N. **O programa só deve ser encerrado quando o usuário atribuir N < 0.**

```
Decimal (base 10): 20
Binario (base 2): 10100
```

7) Desenvolva um programa em C que receba duas *strings* `str1` e `str2` como entrada (compostas por caracteres pertencentes à tabela ASCII) e diga se ambas são anagramas, *i.e.* se possuem a mesma ocorrência de caracteres. Ao final, exiba uma mensagem de acordo com o formato abaixo (cada linha representa um caso de teste para `str1` e `str2`):

```
‘listen’ e ‘silent’ sao anagramas.
‘bola’ e ‘lado’ nao sao anagramas.
```

8) Desenvolva um programa em C que receba uma *string* como entrada e verifique se ela é um palíndromo, *i.e.* se é igual à *string* de entrada quando lida de trás para a frente (ex.: ARARA). Ao final, exiba uma mensagem de acordo com o formato abaixo (cada linha representa um caso de teste):

```
‘arara’ eh palindromo.
‘bola’ nao eh palindromo.
```

9) Desenvolva um programa em C que receba uma *string* como entrada contendo 0 ou mais caracteres ASCII e informe quantas palavras essa *string* possui. Define-se como *palavra* quaisquer grupos de caracteres **separados por um ou mais espaços**. Ao final,

o programa deve exibir uma mensagem que informe a quantidade de palavras encontradas. Exemplo de *string* de entrada: "`Isso aqui eh um teste _+`". Saída esperada:

```
Foram encontradas 6 palavras.
```

10) Elabore um programa em C que implemente a Cifra de César. Devem ser recebidos como entradas três argumentos: (i) uma *string* formada por caracteres ASCII, chamada `mensagem`, (ii) um valor inteiro chamado `pulo` e (iii) um caractere chamado `direcao`, que pode assumir dois valores: 'E' para esquerda ou 'D' para direita. O programa deverá codificar a mensagem ao trocar cada caractere da mensagem pelo seu respectivo caractere localizado na posição `mensagem[i] + pulo`, caso `direcao` seja 'D' ou `mensagem[i] - pulo`, caso `direcao` seja 'E'. **Considere o alfabeto compreendendo os valores da tabela ASCII pertencentes ao intervalo [33, 126]**. **Os espaços não devem ser codificados**. Obs.: atente-se aos casos de fronteira: caso a codificação ASCII de um caractere for 126, pulo for igual a 3 e direção igual a 'D', o caractere escolhido será o 35 e vice-versa. Ao final, o programa deve exibir a mensagem criptografada. Exemplo de execução:

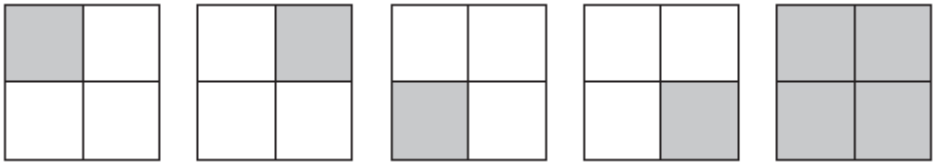
```
Digite a mensagem a ser criptografada: Essa mensagem eh muito
importante!

Digite o valor do pulo: 5

Digite a direcao da codificacao (E/D): D

Mensagem criptografada: Jxxf rjsxfljr jm rznyt nrutwyfsyj&
```

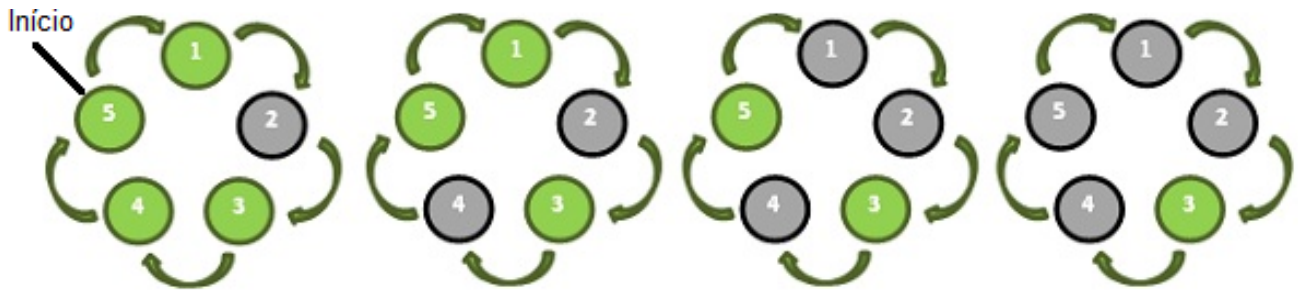
11) Richard Feynman era um físico americano muito famoso e ganhador do Prêmio Nobel de Física. Recentemente, foram encontrados alguns papéis e notas que acredita-se terem pertencido a Feynman. Entre diversas anotações, havia um guardanapo onde ele escreveu um simples desafio: "quantos quadrados diferentes existem em um quadriculado de N x N quadrados?". No mesmo guardanapo havia um desenho, que está reproduzido abaixo, mostrando que para N = 2, a resposta é 5.



Implemente um programa em C que leia um número inteiro N >= 0 e retorne quantos quadrados diferentes existem em um quadriculado de N x N quadrados. O programa deve ser encerrado somente quando N = 0. Exemplos de entradas e saídas abaixo.

Exemplo de Entrada	Exemplo de Saída
2 1 8 0	5 1 204

12) O problema de Josephus é assim conhecido por causa da lenda de Flavius Josephus, um historiador judeu que viveu no século 1. Segundo o relato de Josephus do cerco de Yodfat, ele e seus companheiros (40 soldados) foram presos em uma caverna, cuja saída foi bloqueada pelos romanos. Eles preferiram suicidar-se a serem capturados, e decidiram que iriam formar um círculo e começar a matar-se pulando de três em três. Josephus afirma que, por sorte ou talvez pela mão de Deus, ele permaneceu por último e preferiu entregar-se aos romanos a suicidar-se. Elabore um programa em C que leia um inteiro representando a quantidade de casos de teste. Para cada caso de teste, retorne a posição do sobrevivente, após ler dois números inteiros: (i) `numero` e (ii) `salto`. O número `numero` representa a quantidade de pessoas no círculo, numeradas de 1 até `numero`. O número `salto` representa o tamanho do salto de um homem até o próximo homem que será morto. A figura a seguir ilustra um exemplo onde `numero = 5` e `salto = 2`.



Neste exemplo o elemento que restará após as eliminações é 3. A seguir, há mais exemplos de entradas e saídas.

Exemplo de Entrada	Exemplo de Saída
3 5 2 6 3 1234 233	3 1 25