PUC Minas

Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática (ICEI)
Engenharia de Computação / Engenharia de Software
Disciplina: Algoritmos e Estruturas de Dados I

Lista de Exercícios 5

Instruções e dicas

A) As questões são individuais. Em caso de cópias de trabalho a pontuação será zero para os autores originais e copiadores. Não serão aceitas justificativas como: “Fizemos o trabalho juntos, por isso estão idênticos”.

B) Documente cada uma das funções/procedimentos por meio comentários.

A seguir um exemplo de uma função que recebe um número n por parâmetro calcula e retorna o seu cubo de n.

```
/*
Descrição: a função tem por objetivo calcula o cubo de n.
Entrada: n (inteiro)
Saída: inteiro
*/
int cubo(int n) {
  return n*n*n;
}
```

C) Veja exemplo de código para leitura de frase (*string* longa com espaços) e armazenamento em vetor de char.

```
char frase[60];
scanf(" %[^\n]", &frase); // opção para leitura sem problema com espaços
fflush(stdin);
printf("%s", frase);
```

D) Caso necessite, veja um exemplo de uso da Função *rand* para geração de número aleatórios entre 0 e 99.

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void){
        // comando para gerar numeros aleatorios diferentes
        srand(time(0));

        printf("\nDez numeros aleatorios entre 0 e 99: ");
        for(int i=0; i<10; i++)
                printf(" %i", rand() % 100);
        return 0;
}
```

Parte 1 - Questões para serem entregues no Canvas

Resolva os exercícios a seguir e entregue pelo CANVAS. Cada exercício deve conter um arquivo no **formato .C.** O uso apropriado de funções procedimentos serão considerados na avaliação.

1. Escreva um programa que leia da entrada padrão 5 números reais, que devem ser armazenados em um arranjo. Em seguida, gere outro arranjo, cujos valores correspondem ao dobro dos respectivos elementos do primeiro arranjo.

2. Implemente um procedimento *preencheNotas* que receba e preencha um vetor com as notas de uma turma de 10 alunos. Faça um procedimento *calculaMedia* que receba um vetor preenchido com as notas, calcule a média da turma e conte quantos alunos obtiveram nota acima da média. Esse procedimento deve exibir a média e o resultado da contagem. Faça um programa (main) que declare e preencha as devidas o vetor com valores digitados e acione os procedimentos.

```c
void preencheNotas(int tam, int vetor[]);
void calculaMedia(int tam, int vetor[]);
```

3. Implemente um procedimento preencheValores que preencha 2 vetores X e Y, com 10 elementos cada com valores **aleatórios** entre 10 e 20. Implemente um procedimento que receba os dois vetores previamente preenchidos e gere um novo vetor Z com os elementos desses 2 vetores intercalados de tal forma que nas posições ímpares do novo vetor estejam os elementos do primeiro vetor e nas posições pares os elementos do segundo vetor recebido por parâmetro. Faça um procedimento que receba e exiba o conteúdo de um vetor. Implemente um programa que faça as devidas declarações e acione os módulos para exemplificar o seu uso.

```c
void preencheValores(int x[], int y[], int length);
void intercala(int x[], int y[], int length, int z[]);
```

4. Peça ao usuário que crie uma senha. Utilize um para armazenar a senha. Valide se a senha atende aos critérios de segurança, como ter pelo menos 8 caracteres, conter pelo menos uma letra maiúscula, uma letra minúscula, um número e um caractere especial. Imprima se a senha é válida ou não.

5. Crie uma função que recebe um vetor de caracter por parâmetro que representa um endereço de e-mail para validar se o e-mail é válido. A função deverá retornar a posição do símbolo '@'. Se o símbolo não estiver presente, então a função deverá retornar –1.

Implemente um *main* que peça ao usuário que insira um endereço de e-mail e depois chame a função passando esse e-mail por parâmetro. Imprima se o endereço de e-mail é válido ou não.

6. Crie um procedimento que recebe um vetor de caracteres. Esse procedimento deverá contar o número de palavras que são repetidas no texto e imprima a contagem de cada palavra repetida juntamente com sua frequência. Implemente o mais que solicita ao usuário o texto e chame o procedimento passando o vetor de caracteres.

7. Implemente a função abaixo, que receba como parâmetro uma string e converte, individualmente, cada caractere para maiúsculo. A função deverá retornar a string convertida para maiúsculo. A string a ser retornada deverá ser criada utilizando o comando malloc.

```
char* capitalizeString(char *vetor, int tamanho);
```

Ilustre o funcionamento da função criando um main que solicita ao usuário o texto e chama a função. O main deverá imprimir o resultado da função.

Parte 2 - Questões que não precisam ser entregues (vetor numéricos)

1. Escreva um programa para preencher um vetor com 20 vinte valores inteiros (os valores podem ser lidos do teclado ou gerados automaticamente). Em seguida, o sistema deve solicitar ao usuário um valor, que deve ser pesquisado no vetor. Imprima as posições do vetor que armazena o valor informado.

2. Escreva um programa que preencha um vetor de tamanho 100 com os 100 primeiros números naturais que não são múltiplos de 6 e que não terminam com 6.
Atenção: todas as 100 posições do vetor devem ser preenchidas.

3. Escreva um algoritmo que:
(a) Crie um arranjo de 5 elementos inteiros e o preencha de números
(b) Procure a posição do menor elemento deste arranjo
(c) Troque o menor elemento com elemento da primeira posição
(d) Imprima os elementos do arranjo

Cada um dos itens acima devem ser implementados em funções ou procedimentos separados que

recebem o vetor por parâmetro.

4. Escreva um programa que receba do usuário dois vetores, A e B, com 10 números inteiros cada. Crie um novo vetor, C, calculando C = A - B. Mostre na tela os dados do vetor C.

5. Implemente um procedimento *preencheValores* que preencha um vetor X de 10 elementos. Na sequência, faça um procedimento *copiaNegativos* que receba um vetor preenchido, teste e copie todos os valores negativos deste vetor para um novo vetor, sem deixar elementos vazios entre os valores copiados. O vetor contendo números negativos deve conter até 10 elementos - após o último número negativo (caso não existam 10 número negativos) o vetor deve conter o número 0. Faça um programa (main) que acione os procedimentos e imprima o vetor de números negativos, sem imprimir o valor zero.

```
        void preencheValores(int vetor[], int length);
void copiaNegativos(int vetor[], int length, int vetorNeg[]);
```

6. Escreva um programa que preencha e imprima na tela um vetor de tamanho 50, cujos valores são dados pela seguinte fórmula: vet[i] = (i + (3 x i)) % (i+1)

onde i corresponde à posição do elemento no vetor.

7. Escreva um programa que leia da entrada padrão 5 números reais, que devem ser armazenados em um arranjo. Em seguida, gere outro arranjo, cujos valores correspondem ao dobro dos respectivos elementos do primeiro arranjo.

8. Escreva um vetor de inteiros A com 10 elementos. Em seguida, leia um vetor de inteiros B, também com 10 elementos. Durante a leitura do vetor B, se o elemento na posição i for igual ao elemento na mesma posição do vetor A, rejeitar o valor e solicitar a entrada de um novo valor. A rejeição deve ocorrer até que o valor seja digitado um valor válido. Após a leitura dos vetores A e B, criar um algoritmo que calcule e imprima a diferença entre cada um dos elementos.

9. Escreva um programa que armazene as idades dos alunos que estão presentes em uma aula da disciplina de Algoritmos e Estrutura de Dados I. Considere que o vetor possa conter até 60 registros. Sabe-se que, em uma dada aula, alguns alunos podem ter faltado. Com isso, leia elementos até que seja digitado um valor 0 ou enquanto a quantidade de alunos for inferior à capacidade do vetor. Imprima:

- Idade de todos os alunos presentes na aula (não imprimir idades não cadastradas).
- Idade de todos os alunos com idade superior à média.

Parte 3 - Questões que não precisam ser entregues (vetor de caracter/strings)

1. Solicite ao usuário que insira uma string. Conte o número de vogais na string utilizando um vetor para armazenar as vogais ('a', 'e', 'i', 'o', 'u') e percorrendo a string fornecida. Imprima o número total de vogais encontradas.

2. Crie um vetor de strings. Peça ao usuário que insira várias strings. Ordene as strings em ordem alfabética utilizando qualquer algoritmo de ordenação (por exemplo, bubble sort). Imprima as strings ordenadas.

3. Peça ao usuário que insira duas strings. Utilize vetores para armazenar as duas strings. Concatene as duas strings em uma terceira. Imprima a string resultante.

4. Solicite ao usuário que insira uma frase. Utilize vetores para armazenar as palavras da frase. Inverta a ordem das palavras na frase (sem inverter a ordem das letras em cada palavra). Imprima a frase com as palavras invertidas.

5. Peça ao usuário que insira uma palavra ou frase. Utilize vetores para armazenar a palavra ou frase. Verifique se a palavra ou frase é um palíndromo (ou seja, se pode ser lida da mesma forma de trás para frente e vice-versa). Imprima se a palavra ou frase é um palíndromo ou não.

6. Solicite ao usuário que insira duas palavras. Utilize vetores para armazenar as palavras. Verifique se as duas palavras são anagramas uma da outra (ou seja, se contêm as mesmas letras, independentemente da ordem). Imprima se as palavras são anagramas ou não.

7. Peça ao usuário que crie uma senha. Utilize vetores para armazenar a senha. Valide se a senha atende aos critérios de segurança, como ter pelo menos 8 caracteres, conter pelo menos uma letra maiúscula, uma letra minúscula, um número e um caractere especial. Imprima se a senha é válida ou não.

8. Solicite ao usuário que insira uma frase e duas palavras: uma palavra a ser buscada na frase e outra palavra para substituir a palavra encontrada. Utilize vetores para armazenar a frase e as palavras. Busque a palavra na frase e substitua todas as ocorrências pela segunda palavra. Imprima a frase resultante após a substituição.

9. Escreva um programa que leia uma palavra e a imprima de trás-para-frente.

10. Escreva um programa que retorne o número de vogais e de consoantes em uma string. Desconsiderar caracteres numéricos, espaços e pontuação.

11. Radical é o elemento que contém o significado básico em uma palavra e do qual é possível constituir uma família de palavras. Em um verbo regular, o radical é a parte invariante, que dá origem a todas as conjugações. Escreva um programa que leia um verbo regular terminado em "AR", obtenha o radical e retorne a conjugação nos seguintes tempos verbais:

    1. Presente do indicativo;
    2. Pretérito perfeito do indicativo;
    3. Futuro de presente do indicativo.

12. Escreva um programa que conte a frequência de cada caractere em uma string fornecida pelo usuário. Utilize um vetor para armazenar as contagens de cada caractere (assumindo que estamos lidando apenas com caracteres ASCII). Imprima a frequência de cada caractere na string.

13. Implemente um programa que ordene um vetor de strings em ordem crescente de tamanho das palavras. Utilize o algoritmo de ordenação de sua preferência (por exemplo, bubble sort, insertion sort). Imprima as palavras ordenadas.

14. Escreva um programa que conte o número de palavras em um texto fornecido pelo usuário. Utilize um vetor de caracteres para armazenar o texto. Considere uma palavra como sendo uma sequência contígua de caracteres alfabéticos.

15. Crie um programa que valide se um número de CPF fornecido pelo usuário é válido ou não. Utilize um vetor de caracteres para armazenar o CPF. Implemente o algoritmo de validação do CPF, considerando os dígitos verificadores.

16. Implemente um programa que conte o número de dígitos em um número inteiro fornecido pelo usuário. Utilize um vetor de caracteres para armazenar o número como uma string. Conte o número de caracteres na string para determinar a quantidade de dígitos.

17. Desenvolva um programa que gere senhas aleatórias de acordo com critérios específicos, como comprimento e tipos de caracteres permitidos. Utilize vetores de caracteres para armazenar as senhas geradas. Permita que o usuário especifique o comprimento da senha e os tipos de caracteres (letras maiúsculas, minúsculas, números, caracteres especiais) a serem incluídos.

18. Implemente um programa que criptografe uma mensagem fornecida pelo usuário utilizando a cifra de César. Utilize vetores de caracteres para armazenar a mensagem original e a mensagem criptografada. Permita que o usuário especifique o deslocamento (chave) a ser utilizado na criptografia.