Lista de Exercícios 1

Ponteiros

1. Defina ponteiro, qual a sua importância? Cite um exemplo da sua aplicação.
2. Diferencie memória alocada estaticamente de memória alocada dinamicamente.
3. Comente sobre as funções MALLOC e FREE.
4. Elaborar um programa que leia dois valores inteiros (A e B). Em seguida faça uma função que retorne a soma do dobro dos dois números lidos. A função deverá armazenar o dobro de A na própria variável A e o dobro de B na própria variável B.
5. Faça um programa que leia três valores inteiros e chame uma função que receba estes 3 valores de entrada e retorne eles ordenados, ou seja, o menor valor na primeira variável, o segundo menor valor na variável do meio, e o maior valor na última variável. A função deve retornar o valor 1 se os três valores forem iguais e 0 se existirem valores diferentes. Exibir os valores ordenados na tela.
6. Implemente uma função que calcule a área da superfície e o volume de uma esfera de raio R. Essa função deve obedecer ao protótipo:

   void calc_esfera(float R, float *area, float *volume)

Funções

1. Escreva uma função que gera um triângulo lateral de altura 2*n-1 e n largura. Por exemplo, a saída para n = 4 seria:

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

2. Escreva uma função que gera um triângulo de altura e lados n e base 2*n-1. Por exemplo, a saída para n = 6 seria:

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

3. Crie uma função que compara duas strings e que retorna se elas são iguais ou diferentes.
4. Implemente a função a qual recebe duas strings, str1 e str2, e concatena a string apontada por str2 à string apontada por str1.
5. Implemente a função a qual recebe duas strings, str1 e str2, e um valor inteiro positivo N . A função concatena não mais que N caracteres da string apontada por str2 à string apontada por str1 e termina str1 com NULL.
6. Faça uma função que dado um caractere qualquer retorne o mesmo caractere sempre em maiúsculo.

Structs

1. Escreva um trecho de código para fazer a criação dos novos tipos de dados conforme solicitado abaixo:
   - Horário: composto de hora, minutos e segundos.
   - Data: composto de dia, mês e ano.
   - Compromisso: composto de uma Data, Horário e texto que descreve o compromisso.
2. Crie uma estrutura representando os alunos de um determinado curso. A estrutura deve conter a matrícula do aluno, nome, nota da primeira prova, nota da segunda prova e nota da terceira prova.
   (a) Permita ao usuário entrar com os dados de 5 alunos.
   (b) Encontre o aluno com maior nota da primeira prova.
   (c) Encontre o aluno com maior média geral.
   (d) Encontre o aluno com menor média geral
   (e) Para cada aluno diga se ele foi aprovado ou reprovado, considerando o valor 6 para aprovação
3. Faça um programa que leia os dados de 10 alunos (Nome, matricula, Média Final), armazenando em um vetor. Uma vez lidos os dados, divida estes dados em 2 novos vetores, o vetor dos aprovados e o vetor dos reprovados, considerando a média mínima para a aprovação como sendo 5.0. Exibir na tela os dados do vetor de aprovados, seguido dos dados do vetor de reprovados.

4. Fazer um programa para simular uma agenda de telefones. Para cada pessoa devem-se ter os seguintes dados: Nome, E-mail, Endereço (contendo campos para Rua, número, complemento, bairro, cep, cidade, estado, país). Telefone (contendo campo para DDD e número)
   (a) Definir a estrutura acima.
   (b) Declarar a variável agenda (vetor) com capacidade de agendar até 100 nomes.
   (c) Definir um bloco de instruções busca por primeiro nome: Imprime os dados da pessoa com esse nome (se tiver mais de uma pessoa, imprime para todas).
   (d) Definir um bloco de instruções busca por mês de aniversário: Imprime os dados de todas as pessoas que fazem aniversário nesse mês.
   (e) Definir um bloco de instruções insere pessoa: Insere por ordem alfabética de nome.
   (f) Definir um bloco de instruções retira pessoa: Retira todos os dados dessa pessoa e desloca todos os elementos seguintes do vetor para a posição anterior.

Listas Encadeadas

1. **Referente ao trecho de código a seguir:**

```
1   Elemento *NameFuncao(Lista *lt){
2       Elemento *aux = lt->primeiro;
3       if(aux == NULL){
4           return aux;
5       }else{
6           if(aux->proximo == NULL){
7               lt->primeiro = NULL;
8               lt->ultimo = NULL;
9           }else{
10              lt->primeiro = lt->primeiro->proximo;
11              lt->primeiro->anterior = NULL;
12          }
13          aux->anterior = aux->proximo = NULL;
14          lt->num--;
15          return aux;
16      }
17  }
```

Figura 1: Função desconhecida

   1. **A função implementa qual funcionalidade de Lista? Dê um nome para a função.**
   2. **O Código refere-se a uma função de lista simplesmente ou duplamente encadeada?**
   3. **Comente sobre os testes que ocorrem nas linhas 3,6,9.**
   4. **Comente sobre a linha de código 10.**
   5. **As linhas 10 e 11 podem ter sua ordem troca. Comente.**

2. **Referente ao trecho de código a seguir:**

```
1   void nomefunção(Lista lt){
2       elemento *aux = lt.ultimo;
3       if(aux == NULL)
4           printf("\n LISTA VAZIA!");
5       else{
6           printf("\n Início da Lista!");
7           while(aux != NULL){
8               mostraElemento(*aux);
9               //insira o comando ausente
10          }
11          printf("\n Fim da Lista DE!");
12      }
13  }
```

Figura 2: Função desconhecida

   1. **A função implementa qual funcionalidade de Lista? Dê um nome para a função.**
   2. **O Código refere-se a uma função de lista simplesmente ou duplamente encadeada? Comente.**
   3. **Defina a linha de código ausente (linha 9), para o código processar.**

```c
typedef struct numero
{
    int id;
    struct numero *anterior;
    struct numero *proximo;
}Num;


typedef struct LDE
{
    Num *primeiro;
    Num *ultimo; //facultativo
    int qtd;
}LDEN;
```

Figura 3: Lista Números

3. **Considere as estruturas (Figura 3) para uma lista de números inteiros positivos. Implemente as funções insereInício e removeInício. Execute algumas inserções e remoções.**
4. **Considere as estruturas (Figura 3) para uma lista de números inteiros positivos. Implemente as funções insereOrdenado e removePosicao. A função insere ordenado sempre insere o elemento mantendo a ordem crescente do valores. A função remove na posição remove o elemento escolhido pelo usuário, com base no seu valor.**
5. **Faça uma função que retorne quantos números pares existem na lista.**
6. **Considere uma lista contendo números inteiros positivos. Faça uma função que retorne uma nova lista contendo apenas os números pares da lista (cópia).**
7. **Considere uma lista contendo números inteiros positivos. Faça uma função que retorne a média da lista.**
8. **Considere uma lista contendo números inteiros positivos. Faça uma função que retorne uma nova lista como os elementos em ordem inversa a lista original. Mostre as listas.**