**Ponteiro para função**

1. Encontre os erros do seguinte programa.

#include <stdio.h>

#include <conio.h>

void funcA(int A)

{

printf("\n Valor de A dentro da funcao: %d\n", A);

}

void funcB(int B)

{

printf("\n Valor de B dentro da funcao: %d\n", B);

}

int main(int argc, char \*\*argv)

{

int valor = 100;

char tecla;

void (\*pointer)(int);

printf("\n Digite A ou B para entrar na funcao correspondente: ");

do{

tecla = getch();

} while(tecla != 'A' && tecla != 'a' && tecla != 'B' && tecla != 'b');

if (tecla == 'A' || tecla == 'a')

\*pointer = &funcA;

else

if (tecla == 'B' || tecla == 'b')

\*pointer = &funcB;

pointer(valor);

return(0);

}

2. Faça uma função com o protótipo indicado na sequência, que seja capaz de calcular as quatro funções matemáticas simples, por meio do uso de ponteiros para funções.

float calculo(float, float, float (\*)(float, float));

Exemplo:   
 calculo(3.0, 4, adicao);   
 Obs.: sua função chamará uma função adicao para para calcular a soma de 3.0 + 4.

3. Construa um programa que chame, por meio de um ponteiro de função, uma função capaz de calcular se um número é primo ou não. Essa sua função deve retornar 1 caso o valor seja primo e 0, caso contrário.

int (\*primo)(int);

4. Construa um programa que chame, por meio de um ponteiro de função, uma função que compara dois valores inteiros passados como parâmetros e imprime uma mensagem indicando qual dos valores é maior.

void (int, int, void (\*)(int));

5. Faça um programa que permita chamar duas funções, **fibonacci** e **fatorial**, por meio de ponteiro para função.

6. Faça um programa que chame uma função, que recebe como parâmetros dois valores inteiros e um ponteiro para uma das funções **max** ou **min**. Além disso, essa sua função deve imprimir o maior e o menor valor passados como parâmetro.

7. Faça uma função que recebe um valor inteiro e um ponteiro para outra função, esta capaz de verificar se o valor é positivo ou negativo. Por fim, sua função deve retornar um valor booleano indicando o resultado da análise do valor passado como parâmetro.

8. Faça um programa que possua duas funções, onde uma imprime um quadrado qualquer e outra que apague esse mesmo quadrado. Seu programa deve apresentar um menu ao usuário indicando que operação ele deseja realizar e, uma vez escolhida a operação, seu programa deve chamar a função adequada por meio de um ponteiro de função.

9. Faça um programa que permita passar um ponteiro para uma função e uma *string* com o nome de um arquivo somente para leitura. Caso seja possível abrir o arquivo, seu programa deve apresentar uma mensagem indicando esse fato e, caso contrário, não. Observação: todas as chamadas a funções devem ser por meio de ponteiros de função.

10. Faça um programa que chame uma função por meio de um ponteiro para essa função, onde essa função é capaz de receber uma *string* como parâmetro e transforme as letras minúsculas em maiúsculas e as maiúsculas em minúsculas.

11. Faça um programa que receba uma *string* e verifica se essa *string* é palíndrome. Uma vez definida se a string é ou não palíndrome, seu programa deve chamar uma função, por meio de um ponteiro de função, que imprima uma mensagem com o resultado da verificação. **Observação**: uma *string* palíndrome é aquela *string* que lendo em qualquer sentido é igual. Exemplo: SUBI NO ONIBUS.

12. Faça um programa que leia dois valores inteiros e, por meio de ponteiro para função, chame duas funções para calcula o MDC e o MMC desses dois valores. Observação: essas funções devem retornar o valor do MDC e do MMC para a função que o chamou.

13. Faça um programa que permita modificar os valores de passadas pelo usuário de graus Celsius para Fahrenheit e vice-versa de acordo com a opção desejada pelo usuário. As funções devem ser chamadas por meio de um ponteiro para função e retornar um valor em ponto flutuante para a função que o chamou.

14. Escreva um programa que abra um arquivo qualquer. Uma vez aberto o arquivo, seu programa deve chamar duas funções por meio de um ponteiro para função. A primeira deve imprimir na tela o conteúdo do arquivo e a segunda deve inserir em outro arquivo todo o conteúdo do arquivo aberto inicialmente. Obs.: deve ser passado o nome do arquivo que será lido e um ponteiro para a função desejada.

15. Escreva um programa que possua duas funções chamadas por meio de um ponteiro para função. Uma capaz de gerar um vetor a partir de uma matriz e outra capaz de imprimir esse vetor.

16. Escreva um programa que possua duas funções chamadas por meio de um ponteiro para função. Uma é capaz de imprimir na saída padrão uma quantidade de inteiros em forma de matriz, e outra que imprima em uma matriz de inteiros, a tabuada de 1 a 10.

17. Escreva um programa que possua duas funções passadas por meio de ponteiro para função, onde uma função deve imprimir todos os valores da tabela ASCII e outra que imprima o valor decimal correspondente a tecla digitada pelo usuário.

18. Encontre os erros do seguinte código e o demonstre na forma correta:

#include<stdio.h> /\* printf(), scanf() \*/

#define TAMANHO 6

void ler (int v[], int tam);

void imprimir (int v[], int tam);

void inverter (int v[], int tam);

void (\*vetor\_funcoes[3])(int \*, int) = {ler, imprimir, inverter};

int main(int argc, char \*\*argv)

{

int vetor[TAMANHO];

typedef enum {ler=1, imprimir, inverter} Menu;

Menu opcao;

do{

printf("\n\n1 - ler\n2 - imprimir\n3 - inverter\nOutro valor - Sair\n\nDigite uma opcao: ");

scanf("%i", (int \*) &opcao);

if(opcao >= ler && opcao <= inverter)

(vetor\_funcoes[opcao-1])(&vetor, TAMANHO);

} while(opcao >= ler && opcao <= inverter);

return(0);

}

void ler (int v[], int tam)

{

int i;

for (i=0; i<tam; i++)

{

printf("%d = ", i);

scanf("%d", &v[i]);

}

}

void imprimir (int v[], int tam)

{

int i;

for (i=0; i<tam; i++)

printf("%d = %d\n", i, v[i]);

}

void inverter (int v[], int tam)

{

int i, temp;

for (i=0; i<tam/2; i++)

{

temp = v[i];

v[i] = v[tam-i-1];

v[tam-i-1] = temp;

}

}