```c
// Se DEV C++ // F9 -> Compilar     F10 -> Executar         F11 -> Compilar e Executar
// Se VsCode  // F6 -> Compilar e Executar

// Instalação VsCode -> https://code.visualstudio.com/download
// Extensão Microsoft " C/C++ IntelliSense" e " C/C++ Compile Run"
// Instalação MinGW  -> https://code.visualstudio.com/docs/languages/cpp

/* Printf com acentuação no VSCODE
    1° No canto inferior direito do VSCODE (Clicar em UTF-8)
    2° Clicar em "Reopen With Encoding"
    3° Trocar para "Western (ISO 8859-1) iso 88591"
*/

//BIBLIOTECAS
#include <stdio.h>                                  // Funções de Entrada
#include <stdlib.h>                                 // Funções Padrão
#include <locale.h>                                 // Função de texto setlocale()

// CORPO DO PROGRAMA
int main(){ ou void main() {                        //Inicio

    setlocale(LC_ALL, "Portuguese");                // Permite utilizar caracteres especiais como ex: á, ç, é, dentre outros.

    /*DECLARAÇÃO DE VARIÁVEIS*/

    tipo nome;

    /*|   VARIÁVEIS        |N° DE BITS| FORMATO |
      | void               |          |         |
      | Char               |     8    |    %c   |
      | Unsigned Char      |     8    |    %c   |
      | Int                |    32    | %d / %i |
      | Unsigned int       |    32    |    %u   |
      | Short int          |    16    |    %hi  |
      | Unsigned short int |    16    |    %hu  |
      | Long int           |    32    |    %li  |
      | Unsigned long int  |    32    |    %lu  |
      | Float              |    32    |    %f   |
      | Double             |    64    |    %lf  |
      | Long Double        |    96    |
    */

    //Ex:
    int x1, y1, result1;                            // Declarando as Variáveis globais do tipo Int
    int x = 1, y = 1, result = 0;                   // Declaração de Variáveis globais atribuindo Valores iniciais
    float result2;                                  // Variável Global do tipo float

    //Obs: Ex:  " i-- " -> i = i - 1 (Decrementa) ou " i++ " -> i = i + 1 (Incrementa)

    // ENTRADA OU LEITURA DE DADOS
    scanf("%d", &x1);                               // Ler Variável x1
    scanf("%d", &y1);                               // ler Variável y1
    scanf("%d %d", &x1, &y1);                       // ler Variável x1 e y1
    gets(x1);

    //BLOCOS DE COMANDOS
    result = x + y;                                 // OBS: " = " -> Receber ou atribuir  " == " -> igual
    result1 = x1 + y1;
    float result2 = (result1 / 3);

    // ESTRUTURAS DE DECISÃO " if " ou " if else "
    if(/*Condição 1*/){                             // se
        // Blocos de Comandos
        }else if(/*condição 2*/){                   // senão se
            // Blocos de Comandos
            }else{                                  // senão
                // Blocos de Comandos
            }

```

```c
    // if e else sem {}
    if(/*Condição 1*/)
        // Um Comando
        else
            // Um Comando

    // ESTRUTURAS DE REPETIÇÃO " WHILE "

    while(/*Condição*/) {                          // Enquanto a condição for verdadeira o while vai rodar
        // Bloco de Comandos
    }

    // ESTRUTURAS DE REPETIÇÃO " FOR "

    for (i = 0; i > 0; i++ /*i--*/){               // for (var = inicio; condição; incremento)
        // Bloco de Comandos
        break;                                     // Caso necessário
    }

    //Switch case
    int operador;
    printf("Qual operador deseja escolher?\n");
    scanf("%d", &operdador); //
    switch(operador){
        case 1:
               // Bloco de Comandos
               break;
        case 2:
               // Bloco de Comandos
               break;
        case 3:
               // Bloco de Comandos
               break;
        default:                                   // Caso nenhuma das opções seja escolhida
               // Bloco de Comandos
    }

    // SAÍDA OU ESCRITA DE DADOS
    printf("Calculadora de soma: \n");             // Escrever  (\n -> Pular linha)
    printf("O valor da soma é: %d ", result1);     // Escrever
    printf("O valor da soma é: %f ", result2);     // Escrever
    printf("O valor da soma é: %.2f ", result2);   // Escrever
    printf("cont = %d|soma = %d|ValoraSomar = %d\n", cont, soma, ValoraSomar); // Passo a passo

    //COMANDO DE PARADA DO SISTEMA
    system("pause");

    //UTILIZAR VARIÁVEL TIPO Char
    char a;
    printf ( "Entre com um caractere : " ) ;
    scanf ( "%c" , &a ) ;
    printf ( "\nCaractere Digitado: %c\n" , a ) ;
    printf ( "\nCaractere Digitado : %c (Codigo ASCII = %d)\n" , a , a);

    //CRIAR VETOR
    /* Um vetor é uma seqüência de vários valores do mesmo tipo, armazenados
    seqüencialmente na memória, e fazendo uso de um mesmo nome de variável
    para acessar esses valores. Um vetor também pode ser entendido logicamente
    como uma lista de elementos de um mesmo tipo. */

    tipo variável[indice];

    int indice = 10;
    int vetor[indice];

    scanf("%d", &vetor[indice] );
    printf("%d", vetor[indice]);

    int indice;
```

```c
    int vetor[100];
    ...
    for (indice = 0; indice < 100; indice++) {
        // Bloco de comandos
    }

    printf("%d", vetor[indice]);
    scanf("%d", &valores[indice] );

    // PONTEIROS
    /* Um ponteiro é uma variável que contém um endereço de memória
        Existem dois operadores especiais para ponteiros: * e
        & -> É um operador unário que devolve o endereço na memória do seu operando;
        * -> É um operador unário que devolve o valor da variável localizada no
        endereço;

    Exemplo:

    int Numero, Valor, *Ponteiro;
    Numero = 10;
    Ponteiro = &Numero;     // Ponteiro recebe o endereço na memória da variável
    Numero
    Valor = *Ponteiro;      // Valor recebe o valor da variável localizada no endereço

    */

    tipo *Nome;
    int *p = NULL;          // Pode ser inicializado com o valor nulo
    (*Ponteiro)++;          // Incrementa o Conteúdo da Variável apontada pelo
    Ponteiro

    // CRIAR E CHAMAR FUNÇÃO
    void teste(int b) //tipo_da_funcao  NomeDaFuncao (Lista_de_Parametros)
    {
        // corpo da função
    }

    int teste(int b){

        // Bloco de comandos
        return(tipo_da_funcao a)
    }

    int main(){

        TESTE = teste();

    }

} //FIM

```