Estrutura padrão:

#include <stdio.h> aqui ficam as bibliotecas

int main(){

#aqui fica o codigo

}

Escrita e leitura:

printf(""); ← vai exibir para o usuário tudo que estiver entre os ""

scanf("%tipo\_da\_variavel",&variavel); ← deixa o usuário salvar um valor em certa variável (!!!NÃO ESQUECER O "&"!!!)

Tipos de variáveis:

int = a; ← para números inteiros scanf("%d",&a);

float = a; ← para números decimais scanf("%f",&a);

string = a[];← cadeia de caracteres scanf("%s",&a);

char = a; ← caracter simples scanf("%c",&a);

Operadores aritméticos:

+ = adição

- = subtração

\* = multiplicação

/ = divisão

% = sobra (( x % 2) == 0) ← testa se um valor x dividido por dois sobra 0 {!!! apenas para inteiros!!!}

Operadores relacionais:

== ( a == b ← verifica se a é igual a b)

!= (a != b ← verifica se a é diferente de b)

< (a < b ← verifica se a é menor que b)

<= (a <= b ← verifica se a é menor ou igual a b)

> (a > b ← verifica se a é maior que b)

>= (a >= b ← verifica se a é maior ou igual a b)

Operadores lógicos:

&& ← "E" logico ex: a=1,b=1,c=1, (a==b && b==c) verifica se A é igual a B e tambem verifica se B é igual a C

|| ← "OU" logico ex: a=1,b=1,c=0 (a==b || b==c) verifica se A é igual a B ou se B é igual a C

! ← "NÃO" logico ex: a=1 (!(a == 1)) a condição dentro dos parenteses internos retorna verdade porem o ! do parenteses externo faz retornar um valor falso

Operadores de atribuição:

= ← faz uma variavel receber o valor após o "=" ex: int a; (A não tem um valor definido) a = 1; (A agora vale 1)

+= ← a=20, a += 5; equivalente a a = a + 5, agora a vale 25

-= ← a=20, a -= 3; equivalente a a = a - 3, agora a vale 17

\*= ← a=20, a \*= 2; equivalente a a = a \* 2, agora a vale 20

/= ← a=20, a /= 2; equivalente a a = a / 4, agora a vale 10

Incremento/Decremento:

++ ← a=20, a++; equivalente a a = a + 1, agora a vale 21

-- ← a=20, a--; equivalente a a = a - 1, agora a vale 19

Estrutura condicional:

a = 0

if(a > 0){ verifica "SE" a é maior que 0, caso sua instrução tenha apenas uma linha, não precisa de {}

#instrução ← aqui você diz o que deve ser feito caso o if seja verdadeiro

}else if (a < 0){ ← verifica "SE" a é menor que 0, caso a verificação anterior seja falsa

#instrução

}else { ← se nenhuma das condições anteriores forem verdadeiras, ele será utilizado

#instrução

}

Switch Case:

switch (expressão) {

case valor1:

// Código a ser executado se a expressão for igual a valor1

break;

case valor2:

// Código a ser executado se a expressão for igual a valor2

break;

// ... pode ter mais casos

default:

// Código a ser executado se nenhum dos casos acima for verdadeiro

break;

}

Laços de repetição:

while (condição) {

// código

}

for (inicialização; condição; incremento) {

// código

}

do {

// Código a ser executado

} while (condição); // A condição é verificada após a execução do bloco

Bibliotecas que podem ser uteis:

<locale.h> ← para manipulação de locais (configurações regionais)

setlocale(LC\_ALL, ""); ← define o local para a aplicação, agora pode usar acentuação sem dar erro

<math.h> ← para funções matemáticas

pow(base, expoente) ← calcula a potência (base^expoente)

sqrt(x) ← calcula a raiz quadrada de x

fabs(x) ← valor absoluto de x

<stdlib.h> ← para funções de propósito geral

rand() ← gera um número aleatório

srand(tempo) ← inicializa o gerador de números aleatórios com uma semente

atoi(string) ← converte uma string para um inteiro

atof(string) ← converte uma string para um float

system("cls") ou system("clear") ← limpa tudo que foi exibido anteriormente

<unistd.h> ← para operações de sistema POSIX

sleep(segundos) ← suspende a execução por um número de segundos

Extras:

\n ← pula para a próxima linha

\t ← insere uma tabulação (espaço maior)

\\ ← insere uma barra invertida (\)

\' ← insere uma aspa simples (')

\" ← insere uma aspa dupla (")

%% ← insere o símbolo de porcentagem (%)

printf("\033[H\033[J"); ← Limpa a tela

// ← faz comentário na linha

/\* \*/ ← faz comentários em varias linhas