

RESUMO DE FLUXOGRAMA E VARIÁVEIS

Primeiramente, vamos nos informar da diferença de algoritmo e programação, eles se completam, porém não é a mesma coisa já que algoritmos é finita, sequencial e indica os passos para criar um programa, já a programação, como devem saber, é digital, além disso, é feita através de instruções e contém a compilação. Portanto um equilibra e da assistência a sua maneira ao outro.







O fluxograma é uma das formas de montar uma estrutura algorítmica contendo nível de detalhamento adequado para montar ou informar sobre o que o programa irá executar.

A muitas imagens na internet de como um fluxograma funciona e de como ele é organizado, já que cada figura tem seu significado.

Por exemplo:

- Um círculo ou duas linhas paralelas que ao final delas tem uma curva que liga elas: são para indicar que o fluxograma tem seu início ou fim no lugar onde for colocada.
- → : isso é uma seta de sequência, ela que indica onde o fluxograma vai fluindo até que chegue ao fim. Pode ser apontada para qualquer lado.
- Um retângulo: Apresenta quais são os comandos básicos do seu código, como conta de divisão, multiplicação, soma, subtração, entre outras.
- Um retângulo com a parte de baixo como uma onda: é o que a saída de dados, por exemplo, printf. É o conteúdo que vai ser imprimido na tela quando for executado.
- Um paralelogramo: é a entrada de dados, no caso da linguagem C o scanf. No código ele é o que armazena os dados que a pessoa transmitiu para o computador.
- Losango: tem duas funcionalidades distintas, uma delas é ponto de decisão, se é falso ou verdadeiro, ou se a resposta para seguir é sim ou não, nesse caso dependendo do que a pessoa escolher, leva a caminhos diferente por um tempo, no código é o “if else”. Já a outra função dessa figura é com laço de repetição, ou seja, desvio condicional. No código tem três possibilidades, vai depender de qual é melhor ser usado no momento, eles são: “for”, “do-while” e “while”.

Uma imagem para melhor entender as figuras citadas anteriormente.

	Símbolo usado para indicar início e fim
	Símbolo usado para indicar sentido da sequência.
	Símbolo utilizado para indicar comandos básicos
	Símbolo utilizado para indicar saída (impressão de dados)
	Símbolo utilizado para indicar entrada (leitura de dados)
	Símbolo utilizado para indicar pontos de decisão/desvio condicional

Já as variáveis, são importantes em um código, por conta de serem elas a armazenarem informações durante a execução de um programa.

Cada variável tem sua “espécie”, como se elas fossem armazenadas como num supermercado, cada setor tem seu semelhante e iguais. Cada um tem seu espaço de memória. As mais usadas são:

- int: esse tipo de variável armazena números inteiros, exemplo: 1, 2, 3, 4, 10;
- float: armazena números quebrados, exemplo: 1.5 , 25.70;
- char: armazena caracteres, exemplo: Oi, Hello;

Essas são três **tipos** de variáveis.

A tabela que está abaixo mostra todas os tipos e espaço de memória que aguenta armazenar e suas respectivas porcentagens:

Data type	Size (in bytes)	Range	Format Specifier
short int	2	-32,768 to 32,767	%hd
unsigned short int	2	0 to 65,535	%hu
unsigned int	4	0 to 4,294,967,295	%u
int	4	-2,147,483,648 to 2,147,483,647	%d
long int	4	-2,147,483,648 to 2,147,483,647	%ld
unsigned long int	4	0 to 4,294,967,295	%lu
long long int	8	-(2 ⁶³) to (2 ⁶³)-1	%lld
unsigned long long int	8	0 to 18,446,744,073,709,551,615	%llu
signed char	1	-128 to +127	%c
unsigned char	1	0 to 255	%c
float	4		%f
double	8		%lf
long double	16		%LF

Uma variável tem que ser acompanhada pelo seu **identificador**, por exemplo:

int idade;

float valor;

char nome;

Vale ressaltar para evitar problemas futuros no código, coloca o identificador de uma variável começando com números, como, 12idades. E outro conselho válido, nunca usar em um identificador de uma variável, caracteres especiais, como: calçado, área. Ou seja, pode ter área como um nome para a variável, poder tem que tirar o acento, ficando dessa forma “area”.

Por conseguinte, na programação em C o sinal de igual (=) na matemática, quer dizer que uma certa variável está recebendo o valor da outra variável ou um comando básico, como:

<variável> = <variável> / <operação>

mediana = soma / 2;

OPERADORES ARITIMÉTICOS BÁSICOS

- + : soma, é o mesmo símbolo da matemática;
- - : subtração, também é o mesmo da matemática;
- * : multiplicação;
- / : divisão.

OUTROS COMANDOS BÁSICOS EM C

- ESCREVER AO USUÁRIO EM C:

```
printf(<string_formatada>, [<variáveis>]);
```

O printf é um comando que imprime a mensagem que o programador definiu para ser mostrada para o consumidor do produto. Irei mostrar exemplos para que fique mais fácil entendimento.

- printf("Olá mundo!");
- printf("Sua média final é %f", nota_final);
- printf("Colocação: %i - %ld", posição, valor);

O printf sempre é acompanhado de parênteses, ponto e vírgula e aspas duplas, já que aspas simples e duplas tem diferença no código. No entanto, pode usar essa função de outra forma além de pedir dados ou passar uma mensagem, pode informar o dado que foi adquirido, como do segundo e terceiro exemplo.

As %i, %f e %ld, corresponde com seu tipo de variável, como: %d ou %i são para int, %f para float e %ld, essa é outro tipo de variável chamada long int.

Consequentemente, quando quer informar quanto deu o resultado de uma média da turma ou nesse gênero, basta colocar a % do respectivo identificador e depois das aspas inserir uma vírgula e colocar o identificador, encerrando com o parêntese e o ponto e vírgula.

-LER A INFORMAÇÃO QUE O USUÁRIO DIGITA NO TECLADO:

```
scanf([<formatos>], [<variáveis>]);
```

O scanf é usado para obtenção dos dados, como por exemplo, usa o printf comentado anteriormente para informar o usuário o que precisa ser inserido, com o scanf obtém e armazena o dado fornecido.

Exemplos:

- `scanf("%f", ¬a_parcial);`
- `scanf("%i\n%d", &posição, &valor);`

O scanf também é sempre acompanhado por parênteses, ponto e vírgula e aspas duplas. Porém a diferença é que dentro das aspas duplas, só contém a(s) % que está querendo obter, e o “&” comercial (&) tem que sempre estar acompanhado do identificador da variável, exceto para o char.

ESTRUTURA BÁSICA EM C:

```
#include <stdio.h>
```

```
int main() {
```

```
return 0;
```

```
}
```

O código fica entre o `int main()` e o `return 0` a princípio.

ALGUMAS INFORMAÇÕES INTERESSANTES NÃO COMENTADA ANTERIORMENTE:

`//`: comentário que o código ignora (só na linha que foi inserido), são bons para explicar o que acontece no código.

`/*comentário*/`: funciona igual ao anterior, porém vai da onde foi colocado `/*` até o que fecha.

`\n`: pular linha no código.

`%.2f`: usada na impressão de um dado, como no segundo exemplo do `printf`. Serve para ter somente duas casas decimais depois da vírgula, quando for do tipo `float`. A `%.3f` é com três casas depois da vírgula e assim por diante.