Aula 03 – Tipos de Dados.

Algoritmos e Lógica de Programação

Definição

Na vida real, todas as vezes que precisamos guardar algo temos que usar recipientes.

Eles possuem as mais variadas formas e tamanhos. Isto porque cada item que queremos guardar necessita um recipiente apropriado.

Logicamente que eles não são feitos para uso específico, mas para algumas famílias.

Alguns são quadrados para se guardar frios ou qualquer tipo de salgadinho, outros para guardar ovos e ainda outros para líquidos.

Definição

Na linguagem C, nós usamos “recipientes” apropriados para os tipos de dados que queremos guardar na memória, e a linguagem não aceita o que não foi criado especificamente para o tipo de dado.

Em programação nós chamamos estes “recipientes” de variáveis, e o formato que o dado necessita ser guardado chamamos de tipo de dado.

Em C temos os seguintes tipos de dados: char, int, float e double

Variáveis

Sempre que desejamos guardar um dado qualquer devemos criar uma variável, associada a um tipo de dado.

Uma variável nada mais é que uma posição de memória reservada para armazenarmos informação.

Uma variável deve sempre ser definida antes de ser usada.

Ela deve ser de um tipo específico.

Definição de Variáveis

Para definirmos uma variável demos fazer da seguinte forma:

```
Tipo variavel1, variavel2;
```

Exemplo:

```
int i;
char letra;
float pi, raio;
double peso;
```

Nome das Variáveis

Para se definir uma variável precisamos dar um nome coerente para ela e seguir algumas regras:

- O nome de uma variável pode ser composto pelas letras do alfabeto (maiúsculas ou minúsculas), pelos números e pelo caractere underscore (_)
- O primeiro caractere do nome da variável não pode ser um número
- Maiúsculas e minúsculas são consideradas diferentes
- O nome de uma variável não pode ser uma palavra reservada em C
- Não podemos usar acentos

Nome das Variáveis

Mais alguns cuidados a seguir, mas não são regras:

- O nome de uma variável deve ser descritivo daquilo que ela armazena
- O nome de uma variável não dever ser todo em maiúsculas. Para os programadores o uso de nomes em maiúsculas define uma constante
- Caso uma variável tenha mais de uma palavra temos que usar o underscore ou Camel Case para separa-las:

  nome_do_aluno ou nomeDoAluno

Atribuição

Quando uma variável é criada, o compilador reserva uma série de bytes para ela. A quantidade de bytes é determinada por vários fatores como: arquitetura do hardware, arquitetura do SO (Sistema Operacional), versão do SO e versão do compilador.

Quando isto ocorre, a variável tem como conteúdo o que havia naquela posição de memória (lixo), portanto temos que inicia-la, ou seja, atribuir um valor a ela.

Para isso usamos:

```
variável = valor;
```

Atribuição

Exemplo:

Vamos criar uma variável chamada num, do tipo inteiro e depois colocar o número 17 nela:

```
int num;

num = 17;
```

Podemos ainda fazer assim:

```
int num = 17;
```

Atribuição

Você pode ainda criar mais que uma variável ao mesmo tempo:

```
int num = 17, b, x=2, c;
```

Podemos também colocar o mesmo valor em mais de uma variável:

```
int a, b, c, d;

a = b = c = d = 5;
```

Inteiros

Quando criamos uma variável do tipo inteiro (int), dizemos ao compilador para criar uma variável capaz de armazenar os números naturais positivos e negativos.

Podemos fazer as seguintes operações:

Operação	Descrição	Exemplo	Resultado
+	Soma	10 + 3	13
-	Subtração	7 – 2	5
*	Multiplicação	8 * 2	16
/	Divisão	20 / 2	10
%	Módulo (Resto da divisão inteira)	5 % 2	1

Inteiros

Não precisamos detalhar a soma, a subtração e a multiplicação, mas sim a divisão e o módulo.

Imagine que queremos dividir 5 por 2 (usando inteiros).

Você deve imaginar que teremos o resultado 2,5, mas isto não é verdade. Como usamos inteiros o resultado seria 2 com resto 1:

5	2
1	2

Precedência de operadores

Em C, temos que respeitar uma certa ordem para o cálculo de fórmulas:

1. Multiplicação e divisão
2. Soma e subtração

Para rompermos com isso usamos os parênteses.

Exemplo:

```
x = (5 + 5) / 2;
```

Imprimindo Inteiros

No programa ao lado vemos a declaração do inteiro e sua inicialização. Logo em seguida, temos um printf com algumas coisas estranhas...

```
#include <stdio.h>


main()
{
    int num = 200;
    printf("num = %d\n", num);
}
```

Imprimindo Inteiros

O comando printf vem de “print” + “formato”.

Logo, ele não só imprime textos na tela mas também variáveis em formatos específicos.

Ele funciona assim:

Colocamos em seu texto alguns códigos indicando como queremos imprimir uma variável.

```
#include <stdio.h>


main()
{
    int num = 200;
    printf("num = %d\n", num);
}
```

Imprimindo Inteiros

O %d é um destes códigos e ele quer dizer que a variável será impressa em decimal.

Logo após a string devemos colocar as variáveis usando vírgula.

```
#include <stdio.h>


main()
{
    int num = 200;
    printf("num = %d\n", num);
}
```

Impressão de Inteiros

Veja outro exemplo mais complexo e diga o que será impresso:

```
#include <stdio.h>

main()

{

    int idade = 25;

    int ano = 2019;

    printf("Estamos no ano %d e Jose tem %d anos\n", ano,
idade);

    printf("Em %d ele tera %d\n", ano+1, idade+1);

}
```

Leitura de Inteiros

Veja o programa abaixo:

```
#include <stdio.h>
main()
{
    int n1;
    int n2;
    printf("Digite o primeiro numero: ");
    scanf("%d", &n1);
    printf("Digite o segundo numero: ");
    scanf("%d", &n2);
    printf("O resultado de %d + %d = %d", n1, n2, n1 + n2);
}
```

Leitura de Inteiros

Ao lado temos um pedaço do programa para estudarmos o que acontece:

O printf somente emite a mensagem para que o usuário saiba o que deve fazer.

O scanf faz a leitura da variável através do teclado.

```
printf("Digite o primeiro numero: ");

scanf("%d", &n1);
```

Leitura de Inteiros

A parte destacada é muito semelhante ao que vimos no printf.

O que você acha que esta parte faz?

Ela indica qual o tipo de dado eu farei a leitura.

```
printf("Digite o primeiro numero: ");

scanf("%d", &n1);
```

Leitura de Inteiros

Na segunda parte do comando temos a variável que desejamos carregar com um & (e comercial) antes.

Ainda é cedo para entendermos plenamente o porquê disso, mas lembre-se sempre de colocar o & antes da variável quando usar scanf.

```
printf("Digite o primeiro numero: ");

scanf("%d", &n1);
```

Números Reais: float e double

As variáveis do tipo float e double são usadas para armazenar números que possuem parte fracionária.

A diferença entre elas é que geralmente um float ocupa 4 bytes e um double 8. Isto faz com que o double possa armazenar um número muito maior que o float.

Para fazermos a impressão e leitura de números reais usamos %f

Números Reais: float e double

Exemplo:

```
#include <stdio.h>


main()
{
    float n1;
    float n2;
    float soma;
    float media;
    printf("Digite o valor de n1: ");
    scanf("%f", &n1);
    printf("Digite o valor de n2: ");
    scanf("%f", &n2);
    soma = n1 + n2;
    media = soma / 2;
    printf("A media eh: %f", media);
}
```

Caracteres: char

As variáveis do tipo char são capazes de armazenar um único caractere.

Este caractere não é uma string e deve estar dentro de aspas simples.

Exemplo:

```
char letra = 'A';
```

Caracteres: char

Temos duas maneiras de fazer a leitura de caracteres: com scanf e com getchar

Para usar o scanf temos que ter o código %c

Quando temos mais do que um scanf no programa e temos uma leitura de char podemos ter um problema com relação ao buffer de teclado. Para evitar isto usamos:

```
scanf(" %c", &letra);
```

Exemplo de scanf x getchar

```c
#include <stdio.h>

main()
{
    char letra;
    printf("Digite uma letra: ");
    scanf(" %c", &letra);
    printf("Voce digitou %c", letra);
}
```

```c
#include <stdio.h>

main()
{
    char letra;
    printf("Digite uma letra: ");
    letra = getchar();
    printf("Voce digitou %c", letra);
}
```

Exercícios

1. Faça um programa que leia dois inteiros e mostre na tela o resultado das operações tradicionais (+, -, * e /)
2. Faça um programa que peça para o usuário digitar uma data e imprima no formato dd/mm/aaaa.

Exercícios

3. Indique quais declarações estão corretas:
   a) y int;
   b) int ;
   c) integer x;
   d) inta, b;
   e) float f,g, c;
   f) char ch1 = ch2 = ‘A’;
   g) char ch1 = ‘A’, ch2 = ‘A’;

Exercícios

4. Uma variável inteira, quando declarada, é sempre inicializada com:
    a. 0 (zero)
    b. 1 (um)
    c. Um valor aleatório
    d. Um valor negativo

Exercícios

5. Indique os nomes corretos de variáveis:
    a. Valor
    b. &xvar
    c. dez%
    d. a + b
    e. _Saldo
    f. main
    g. a10
    h. 10a
    i. PRECO
    j. saldo_da_conta
    k. saldoDaConta
    l. for
    m. For

Exercícios

6. Faça um programa que leia 4 números e imprima a soma deles.
7. Faça um programa que leia 4 números e imprima a média deles.
8. Faça um programa que cria variáveis, entre com os dados e imprima o resultado, para armazenar: idade, altura e gênero.
9. Faça um programa que peça o ano de nascimento da pessoa, o ano atual e imprima a idade dela.
10. Sabendo que a conversão de Fahrenheit para Celsius se dá pela fórmula f – 32 * 5 / 9, faça um programa que converta uma temperatura de Fahrenheit para Celsius.