

# Aula 1

# Tipos de dados

- A linguagem C oferece cinco tipos de dados básicos:

<b>Tipo</b>	<b>Espaço</b>	<b>Escala</b>
<i>char</i>	1 byte	-128 a +127
<i>int</i>	2 bytes	-32768 a +32767
<i>float</i>	4 bytes	3.4e-38 a 3.4e+38
<i>double</i>	8 bytes	1.7e-308 a 1.7e+308
<i>void</i>	<i>nenhum</i>	<i>nenhuma</i>

- O computador é somente capaz de manipular números
  - Portanto, cada valor de variável é representada por um número da tabela ASCII. Inclusive char, que varia de 0 a 127
  - A tabela ASCII padrão, possui apenas valores positivos. Mas extensões dessa tabela, podem possuir valores negativos

# Declaração de uma variável

- A declaração de uma variável consiste em um tipo e um identificador
  - O tipo determina o espaço de memória que deverá ser alocado para ela
  - e o identificador permitirá que ela seja referenciada no restante do programa.

## *Exemplo 1.2.* Declaração de variáveis.

```
char tecla, opcao;  
int x, y, z;  
float comissao, desconto, salario;
```



*Todo identificador deve iniciar-se com letra (maiúscula ou minúscula) e ser composto exclusivamente por letras, dígitos e sublinhas.*

# Tipos de dados modificados

- Além dos tipos básicos, C oferece também alguns tipos de dados modificados:

Tipo	Espaço	Escala
<i>unsigned char</i>	1 byte	0 a 255
<i>unsigned int</i>	2 bytes	0 a 65535
<i>long int</i>	4 bytes	-2 147 483 648 a +2 147 483 647

- O bit mais à esquerda em *char* ou *int* é chamado de *bit de sinal*
  - utilizado pelo computador para indicar se o valor ASCII é positivo ou negativo

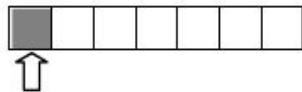


Figura 1.2 – O bit de sinal numa variável do tipo *char*

- Com **unsigned** informamos ao computador, que os valores serão positivos
  - portanto, ganhamos mais 1 bit para representar valores e a escala de valores dobra

# Tipos de dados modificados

**Exemplo 1.3.** Algumas variáveis de tipos modificados.

```
unsigned char contador;  
unsigned int a, b, c;  
long int tam_arquivo;
```

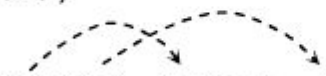
☞ *Os modificadores podem prefixar apenas os tipos char e int. A única exceção feita é long float, que equivale ao tipo double e por isso é raramente utilizado*

# Entrada e saída: função **scanf()**

- permite que um valor seja lido do teclado e armazenado em uma variável
- sintaxe: `scanf("formatação", arg1, arg 2, ..., argn);`

*Exemplo 1.4.* Lendo dados com a função `scanf()`.

```
int idade;  
char sexo;  
...  
scanf("%d %c", &idade, &sexo);
```



Especificador	Representa
%c	um único caracter
%o, %d, %x	um número inteiro em octal, decimal ou hexadecimal
%u	um número inteiro em base decimal sem sinal
%ld	um número inteiro longo em base decimal
%f, %lf	um número real de precisão simples ou dupla
%s	uma cadeia de caracteres ( <i>string</i> )
%%	um único sinal de porcentagem

# Entrada e saída: função **printf()**

- permite exibir informações formatadas na tela
- sintaxe: *printf("formatação", arg1, arg2, ..., argn);*
  - semelhante à scanf, porém com a lista de argumentos contendo os valores e não endereço das variáveis

**Exemplo 1.5.** Exibindo dados com a função *printf()*.

```
#include <stdio.h>
#define PI 3.1415
main() {
    double raio, perim;
    getch();
    printf("\n Qual a medida do raio?");
    scanf("%lf", &raio);
    perim = 2*PI*raio;
    printf("\n O perímetro da circunferência é %lf", perim);
    getch();
}
```

- '\n' usado para saltar uma linha. [enter]

# Operadores aritméticos

Operador	Resultado
+	soma de dois números quaisquer
-	diferença entre dois números quaisquer
*	produto de dois números quaisquer
/	quociente da divisão de dois números
%	resto da divisão de dois número inteiros

- Exercício 2.1: Dada uma temperatura em graus Fahrenheit, informe o valor correspondente em graus Celsius. [Dica:  $C = (F - 32) * (5 / 9)$ ].
- Exercício 2.2. Dadas as medidas dos catetos de um triângulo retângulo, informe a medida da hipotenusa. [Dica: para calcular a raiz quadrada use a função `sqrt()`, definida na biblioteca `math.h`].



# Operadores relacionais

- Não existe um tipo específico para a representação de valores lógicos

- Entretanto, qualquer valor pode ser interpretado como verdadeiro e qualquer outro valor representa verdade”.
- Por exemplo, os valores 5, -3, 1.2 e 'a' são verdade

- Para gerar valores lógicos, usamos operadores relacionais que quando usados para uma comparação, retornam ‘0’ se a mesma for falsa e ‘1’ se verdadeira

**Exemplo 2.1.** Operadores relacionais e valores lógicos

```
...  
printf("%d %d", 5<6, 6>5);  
...
```

A saída produzida pela instrução será 1 0.

Operador relacional	Resultado
$x = y$	verdade se $x$ for igual a $y$
$x \neq y$	verdade se $x$ for diferente de $y$
$x < y$	verdade se $x$ for menor que $y$
$x > y$	verdade se $x$ for maior que $y$
$x \leq y$	verdade se $x$ for menor ou igual a $y$
$x \geq y$	verdade se $x$ for maior ou igual a $y$

- Exemplo:

```
...  
printf("%d %d", 5<6, 6<5);  
...
```

A saída produzida pela instrução será 1 0

# Operadores lógicos

Operador lógico	Resultado
$! x$	<i>verdade se e só se <math>x</math> for falso</i>
$x \&\& y$	<i>verdade se e só se <math>x</math> e <math>y</math> forem verdade</i>
$x    y$	<i>verdade se e só se <math>x</math> ou <math>y</math> for verdade</i>

- Numa expressão contendo operadores aritméticos, relacionais e lógicos, a avaliação é efetuada na seguinte ordem:
  - primeiro avaliam-se todos os operadores aritméticos;
  - em seguida, avaliam-se os operadores relacionais;
  - só então, avaliam-se os operadores lógicos.

# Comandos de decisão

- servem para escolher um entre dois comandos
- em C, é codificado da seguinte maneira: *if( condição ) comando<sub>1</sub>; else comando<sub>2</sub>;*

**Exemplo 2.2.** O uso de decisão simples.

```
#include <stdio.h>

main() {
    float a, b, m;
    printf("\n Informe as duas notas obtidas: ");
    scanf("%f %f", &a, &b);
    m = (a+b)/2;
    if( m >= 7.0 ) printf("\n Aprovado");
    else printf("\n Reprovado");
}
```

- Como há mais de um comando, no exemplo 2.3, faz-se necessário o uso de chaves '{' '}' para abrir e fechar o bloco de comandos

**Exemplo 2.3.** O uso de blocos de instruções.

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

main() {
    float a, b, m;
    clrscr();
    printf("\n Informe as duas notas obtidas: ");
    scanf("%f %f", &a, &b);
    m = (a+b)/2;
    if( m >= 7.0 ) {
        textcolor(BLUE);
        cprintf("\n Aprovado");
    }
    else {
        textcolor(RED);
        cprintf("\n Reprovado");
    }
    getch();
}
```

- *cprint() e textcolor() são para mostrar texto colorido*

# Operador condicional

- proporciona uma maneira mais compacta para decisões simples
- sintaxe: *condição ? expressão1 : expressão2;*

*Exemplo 2.5.* O uso do operador condicional.

```
...  
abs = n>0 ? n : -n;  
...
```

- A instrução acima atribui à variável **abs** o valor absoluto da variável **n**. A expressão **n>0** é avaliada: se for verdadeira, **abs** recebe o próprio valor de **n**; caso contrário, **abs** recebe o valor de **n** com o sinal invertido.

# Exercício

Desenvolver um algoritmo que leia um número inteiro e verifique se o número é divisível por 5 e por 3 ao mesmo tempo.

# Exercício

Desenvolver um algoritmo para ler um número “x” e calcular e imprimir o valor de “y” de acordo com as condições abaixo:

- $y = x$  , se  $x < 1$
- $y = 0$  , se  $x = 1$
- $y = x^2$  , se  $x > 1$

# Exercício

Fazer um programa em C que dado três valores A, B e C, verificar se eles formam um triângulo. Formando triângulo, dizer se é triângulo equilátero, isósceles ou escaleno.

- Condição para ser triângulo: a soma do comprimento de dois lados deve ser maior (ou igual) ao comprimento do terceiro lado.
- Tipos de triângulos:
  - triângulo equilátero: todos os lados são iguais
  - triângulo isósceles: dois lados iguais
  - triângulo escaleno: todos os lados são diferentes

# Exercício

Faça um programa em C que identifique se a raiz quadrada de um dado número inteiro  $X$  é inteira, ou seja, se  $X$  é um número quadrado perfeito.



# Exercício

*Cada character é representado por um byte, e de acordo com o alfabeto ASCII um byte entre 00000000 e 01111111.*

*No computador, cada caractere está representado em uma faixa de valores, que para nós pode ser visualizada como uma faixa de inteiros.*

Escreva um programa em C para checar se um character é uma letra, dígito ou caractere especial.

DICA: façam o seguinte teste: `printf("%d", 'a');`

# Decisão múltipla

- O C oferece uma estrutura de decisão múltipla para precisamos escolher uma entre várias alternativas previamente definidas, por exemplo, em um menu.

- sintaxe: 

```
switch( expressão ) {  
    case constante1 : comando1; break;  
    case constante2 : comando2; break;  
    ...  
    case constanten : comandon; break;  
    default          : comando;  
}
```



O caso default é opcional e, embora seja geralmente posicionado no final do bloco switch, ele pode aparecer em qualquer posição entre os case's especificados.

# Decisão múltipla: exemplo com “erro”

*Exemplo 2.8.* O uso da estrutura de decisão múltipla com vazamentos.

```
#include <stdio.h>

main() {
    int n;

    printf("\n Digite um número: ");
    scanf("%d", &n);

    switch( n ) {
        case 1: putchar('A'); break;
        case 3: putchar('B');
        case 4: putchar('C'); break;
        default: printf('*');
        case 5: putchar('D');
    }
    putchar('.');
}
```

<i>n</i>	Saída
1	A.
2	*D.
3	BC.
4	C.
5	D.

# Exercício

Usando o ***switch***, crie uma calculadora simples (operações de +, -, \*, e /). O usuário digita uma expressão na forma valor1 oper valor2