



Linguagem de Programação

Introdução ao C e a Comandos de decisão

Prof. Francisco Glaubos

Tipos de dados

- A linguagem C oferece cinco tipos de dados básicos:

Tipo	Espaço	Escala
<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.

```
/* 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 compostos 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

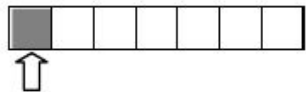


Figura 1.2 – O bit de sinal numa variável do tipo *char*

- 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

```
/* variáveis de tipos modificados*/  
unsigned char tecla, opcao;  
long int x,y,z;
```

- 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

Função scanf()

- permite que um valor seja lido do teclado e armazenado em uma variável
- sintaxe: `scanf("formatação", arg1, arg2, ..., argn);`

```
/* lendo dados com a função scanf() */
```

```
int idade;
```

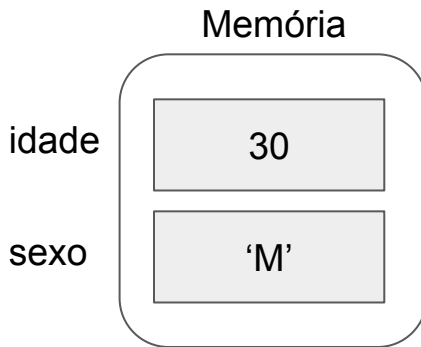
```
char sexo;
```

```
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

Função **scanf()**

- Nomes de variáveis como `idade` e `sexo`, correspondem a locais na memória do computador
- No exemplo anterior:
 - `scanf("%d %c", &idade, &sexo);`
 - suponha que o usuário entre com os valores **30** e **M**
 - o computador irá colocar cada valor em seu respectivo local:



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

```
/* exibindo dados com a função printf*/
```

```
#define pi 3.1415
```

```
int main() {
```

```
    double raio, perim;
```

```
    printf("Qual a medida do raio? \n");
```

```
    scanf("%lf", &raio);
```

```
    perim = 2*pi*raio;
```

```
    printf("O perimetro da circunferencia é: %lf", perim);
```

```
    return 0; }
```

- '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 1:** Dada uma temperatura em graus Fahrenheit, informe o valor correspondente em graus Celsius. *[Dica: $C = (F - 32) * (5 / 9)$].*
- **Exercício 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 um valor lógico: “zero representa falso e qualquer outro valor representa verdade”.
 - Por exemplo, os valores 5, -3, 1.2 e 'a' são verdadeiros, enquanto 0 e 4-4 são falsos.
- Para gerar valores lógicos, usamos operadores relacionais
 - que quando usados para uma comparação, retornam '0' se a mesma for falsa e '1' se verdadeira

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

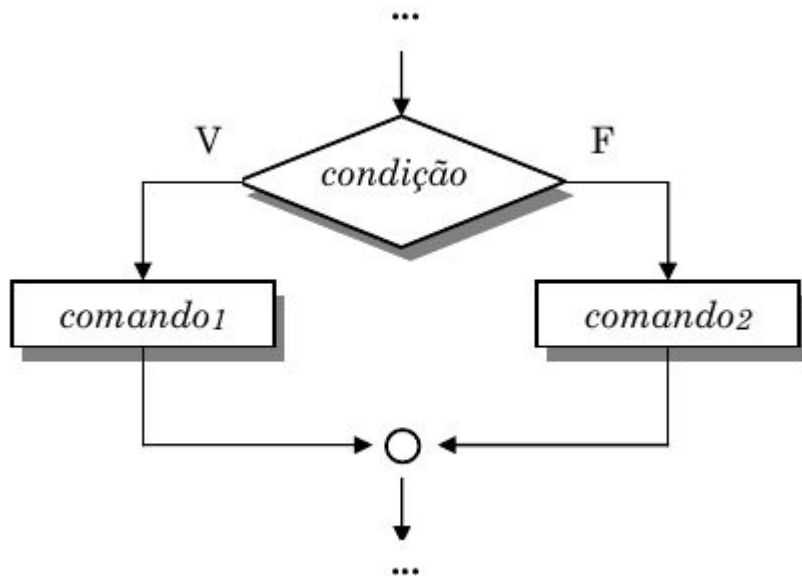
Operadores lógicos

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

Decisão simples

- A estrutura condicional (ou decisão simples), serve para escolher 1 entre 2 comandos alternativos



- Em C, é codificado da seguinte maneira:

```
if (condição) comando1; else comando2;
```

Decisão simples

```
// O uso de decisão simples.  
#include <stdio.h>  
  
void main() {  
    float a, b, m;  
    printf("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");  
}
```

Decisão simples

- *Como há mais de um comando, faz-se necessário o uso de chaves '{' '}' para abrir e fechar o bloco de comandos*
- *cprint() e textcolor() são para mostrar texto colorido*

```
//O uso de blocos de instruções.  
#include <stdio.h>  
#include <conio.h>  
main() {  
    float a, b, m;  
    printf("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");  
    }  
}
```

Operador condicional

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

```
//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

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

- Condição para ser triângulo: a soma do comprimento de dois lados deve ser maior que o 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 constante_1 : comando_1; break;  
    case constante_2 : comando_2; break;  
    ...  
    case constante_n : comando_n; 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 **cases** especificados

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main(){
    int n;
    printf("\n Digite um número: ");
    scanf("%d", &n);
    switch( n ) {
        case 1: printf("A"); break;
        case 3: printf("B");
        case 4: printf("C"); break;
        default: printf("*");
        case 5: printf("D");    }
    printf("."); }
```

- Qual a saída para os seguintes valores de n?

N	Saída
1	
2	
3	
4	
5	

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main(){  
    int n;  
    printf("\n Digite um número: ");  
    scanf("%d", &n);  
    switch( n ) {  
        case 1: printf("A"); break;  
        case 3: printf("B");  
        case 4: printf("C"); break;  
        default: printf("*");  
        case 5: printf("D");    }  
    printf("."); }  

```

- Qual a saída para os seguintes valores de n?

N	Saída
1	A.
2	
3	
4	
5	

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main() {  
    int n;  
    printf("\n Digite um número: ");  
    scanf("%d", &n);  
    switch( n ) {  
        case 1: printf("A"); break;  
        case 3: printf("B");  
        case 4: printf("C"); break;  
        default: printf("*");  
        case 5: printf("D");    }  
    printf("."); }  
}
```

- Qual a saída para os seguintes valores de n?

N	Saída
1	A.
2	*D.
3	
4	
5	

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main() {  
    int n;  
    printf("\n Digite um número: ");  
    scanf("%d", &n);  
    switch( n ) {  
        case 1: printf("A"); break;  
        case 3: printf("B");  
        case 4: printf("C"); break;  
        default: printf("*");  
        case 5: printf("D");    }  
    printf("."); }  
}
```

- Qual a saída para os seguintes valores de n?

N	Saída
1	A.
2	*D.
3	BC.
4	
5	

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main(){
    int n;
    printf("\n Digite um número: ");
    scanf("%d", &n);
    switch( n ) {
        case 1: printf("A"); break;
        case 3: printf("B");
        case 4: printf("C"); break;
        default: printf("*");
        case 5: printf("D");    }
    printf("."); }
```

- Qual a saída para os seguintes valores de n?

N	Saída
1	A.
2	*D.
3	BC.
4	C.
5	

Decisão múltipla: exemplo com “erro”

//O uso da estrutura de decisão múltipla com vazamentos.

```
void main() {  
    int n;  
    printf("\n Digite um número: ");  
    scanf("%d", &n);  
    switch( n ) {  
        case 1: printf("A"); break;  
        case 3: printf("B");  
        case 4: printf("C"); break;  
        default: printf("*");  
        case 5: printf("D");    }  
    printf("."); }  
}
```

- Qual a saída para os seguintes valores de n?

N	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, e o seu programa deve mostrar o resultado da expressão.