Linguagens de Programação I

Tema # 4

Operadores em C

Susana M Iglesias

FUNÇÕES ENTRADA-SAÍDA I/O

- ***printf()***, utilizada para enviar dados ao dispositivo de saída padrão (*stdout*),
- ***scanf()***, utilizada para ler dados do dispositivo de entrada padrão (*stdin*),
- Sintaxe:

```
printf(string_de_controle, lista_de_argumentos);

scanf(string_de_controle, lista_de_argumentos);
```

FUNÇÕES ENTRADA-SAÍDA I/O

- Exemplos:

```
int i; char ch; float fl;
printf("Escreve inteiro: %d\n",i);
scanf("%d", &i);
printf("Escreve caractere: %c\n",ch);
scanf("%c", &ch);
printf("Escreve caractere: %f\n",fl);
scanf("%f", &fl);
```

FUNÇÕES ENTRADA-SAÍDA I/O

- Comandos de formato:

Código	Significado
`%d` ou `%i`	Inteiro decimal
`%c`	Caractere
`%f`	Ponto flutuante decimal
`%e`	Ponto flutuante notação científica
`%o`	número em octal
`%x`	número em hexadecimal

- **Atividade**, faça uma pesquisa sobre os comandos de formatação das funções *printf()* e *scanf()*. [C Completo e Total, 212-228]

OPERADORES

- Tipos de Operadores:

1. atribuição,
2. aritméticos,
3. relacionais,
4. lógicos

OPERADOR DE ATRIBUIÇÃO

- Operador (=) , pode ser utilizado dentro de qualquer expressão válida da linguagem,
- Sintaxe: **`identificador_de_variavel = expressão`**;

OPERADOR DE ATRIBUIÇÃO

- O destino, ou *lvalue* tem que ser uma variável ou um ponteiro,
- O *rvalue* pode ser uma variável ou qualquer expressão válida da linguagem,
- Conversão de tipos: refere-se à situação em que variáveis de um tipo são misturadas com variáveis de outro tipo.
- A maioria das linguagens proíbem conversão automática de tipos.

OPERADOR DE ATRIBUIÇÃO

- O C, permite conversão automática de tipos, nenhuma mensagem de erro ou aviso tem lugar quando uma conversão de tipos acontece.
- **Regra de conversão de tipos**: num comando de atribuição o valor do *rvalue* é convertido no tipo do *lvalue*, antes de fazer a atribuição.

OPERADOR DE ATRIBUIÇÃO

- Exemplo de conversão de tipos:

```
int x;
char ch;
float f;
```

```
void func1(){
   ...
   ch = x;
   x = f;
   f = ch;
   f = x;
   ...}
```

- As conversões de tipos apenas mudam a forma em que o valor é representado, em ocasiões pode ocorrer perda de informação.
- O C permite atribuições múltiplas em um único comando. Ex:

```
x = y = z = 0
```

OPERADORES ARITMETICOS

- Os operadores +, - , * e / funcionam em C com seu significado matemático,
- eles são classificados como operadores binários,
- o operador – pode ser unário,
- Quando / é aplicado a inteiro ou caractere qualquer resto é truncado, operação de divisão inteira.
- Outro operador é %, devolve o resto da divisão inteira, é um operador binário, ambos operandos devem ser inteiros.

OPERADORES ARITMETICOS

- O C inclui dois operadores que geralmente não aparecem na notação matemática,
- operador de incremento ++
- operador de decremento --
- `x = x + 1` é o mesmo que `x++`
- os operadores decremento é incremento são unários
- o operador incremento(decremento) pode aparecer antes o depois do operando,

```
a = 100;        a = 100;
b = ++x;        b = ++x;
```

- os operadores ++ e -- permitem fazer as operações mais eficientemente.

OPERADORES ARITMETICOS

- O C calcula as operações aritméticas em uma seqüência determinada pelas regras de precedência,
- os parêntesis tem na linguagem C o mesmo significado que em expressões algébricas,
- os parêntesis tem maior precedência que os outros operadores,
- quando dos operadores com igual precedência estão no mesmo nível, a expressão é calculada da esquerda a direita,

OPERADORES ARITMETICOS

- Precedência de operadores

Operador	Ordem de precedência
`()`	[1], se houver parêntesis aninhados calcula-se primeiro o parêntesis mais interno
`++ ou --`	[2]
`*, / ou %`	[3] no caso de vários operadores são calculados da esquerda a direita
`+ ou -`	[4] no caso de vários operadores são calculados da esquerda a direita

OPERADORES ARITMETICOS

- Exemplos:

```
a = 1; b = 2; c = 3; d = 4; e = -1

a + b * c

(a - b) * d

c % b + e * c

((a+b) * c - e + d / b) % b

b - (a + 3) * e - d % b + a / b
```

ESTRUTURA DE SELEÇÃO

- Para tomar decisões em um programa utilizamos a estrutura *if-then-else*
- Sintaxe

```
if (expressão){bloco A} else {bloco B}
```

- Se expressão for verdadeiro o bloco A é executado, senão bloco B é executado.
- A estrutura de seleção junto aos operadores lógicos e relacionais, permite executar operações (ou caminhos) diferentes durante a execução do programa.

OPERADORES RELACIONAIS

- operador relacional, refere-se às relações que os valores podem ter uns com os outros.

Operador	Significado
>	maior que
>=	maior ou igual que
<	menor
<=	menor ou igual que
==	igual a
!=	diferente de

OPERADORES LÓGICOS

- operador lógico, refere-se às maneiras que relações podem ser conectadas.

Operador	Significado
&&	AND
| |	OR
!	NOT

- verdadeiro qualquer valor diferente de zero,
- falso é zero,
- as expressões que utilizam operadores lógicos e relacionais devolvem zero para falso e 1 para verdadeiro.

OPERADORES LÓGICOS

- Tabela da verdade de operadores lógicos

a	b	a && b	a || b	!a
0	0	0	0	1
0	1	0	1	1
1	1	1	1	0
1	0	0	1	0

- Os operadores lógicos e relacionais tem menor precedência que os operadores aritméticos.
- Precedência:

```
!
>   >=   <  <=
==  !=
&&
||
```

LÓGICOS E RELACIONAIS

- Exemplos:

```
a = 0; b = 1; c = 1; d = 5;

a > 0

(d >= b) && (d < 50)

((a == 0) && !c)

((b == c) || (c > -1)) && ((a != 0) || (c <= 0))
```

PALAVRAS RESERVADAS

- As palavras reservadas da linguagem C são:

auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

(cinza – estudadas, brancas – por estudar)

O TERCEIRO PROGRAMA EM C

```
/* Imprime mensagem de aprovado(desaprovado) */
#include <stdio.h>
#include <stdlib.h>
int main(){
  float nota;
  printf("Entre com a nota do aluno: ");
  scanf("%f", &nota);
  if (nota>=5.0)
   printf("Aluno aprovado :-) \n\a");
  else
   printf("Aluno desaprovado :-( \n\a");
  system("PAUSE");
  return 0;
}
```