Comandos de Repetição
(ILP-010)

Prof. Dr. Silvio do Lago Pereira

Departamento de Tecnologia da Informação

Faculdade de Tecnologia de São Paulo

Comandos de repetição

Operador de atribuição aritmético

é um notação concisa que combina uma atribuição com uma operação aritmética.

- **Forma estendida**: *variável* **=** *variável operador expressão*
- **Forma concisa**: *variável operador***=** *expressão*

Forma estendida	Forma compacta
`x = x+y`	`x += y`
`x = x-y`	`x -= y`
`x = x*y`	`x *= y`
`x = x/y`	`x /= y`
`x = x%y`	`x %= y`

Operadores de atribuição aritméticos são ideais para a manipulação de **variáveis acumuladoras**!

Comandos de repetição

Operador de incremento/decremento

é um notação ainda mais concisa que combina uma atribuição com uma operação de soma ou subtração **unitária** (uma única unidade a mais ou a menos).

Forma estendida	Forma posfixa	Forma prefixa
`x = x+1`	`x++`	`++x`
`x = x-1`	`x--`	`--x`

Note que:

- A forma **posfixa** altera a variável **depois** que seu valor é usado na expressão.
- A forma **prefixa** altera a variável **antes** que seu valor seja usado na expressão.
- Quando usadas **isoladamente**, as formas posfixa e prefixa são **equivalentes**.

Operadores de atribuição aritméticos são ideais para a manipulação de **variáveis contadoras**!

Comandos de repetição

Exemplo 1. Uso isolado de incremento/decremento [1ª versão]

```
#include <stdio.h>
int main(void) {
    int x=5, y=5;
    ++x;
    y--;
    printf("x=%d y=%d\n",x,y); // x=6 y=4
    return 0;
}
```

Exemplo 2. Uso isolado de incremento/decremento [2ª versão]

```
#include <stdio.h>
int main(void) {
    int x=5, y=5;
    x++;
    --y;
    printf("x=%d y=%d\n",x,y); // x=6 y=4
    return 0;
}
```

Comandos de repetição

Exemplo 3. Uso de incremento/decremento em expressão [1ª versão]

```
#include <stdio.h>
int main(void) {
    int x=5, y, z;
    y = ++x + 2;
    z = x-- + 2;
    printf("x=%d y=%d z=%d\n",x,y,z); // x=5 y=8 z=8
    return 0;
}
```

Exemplo 4. Uso de incremento/decremento em expressão [2ª versão]

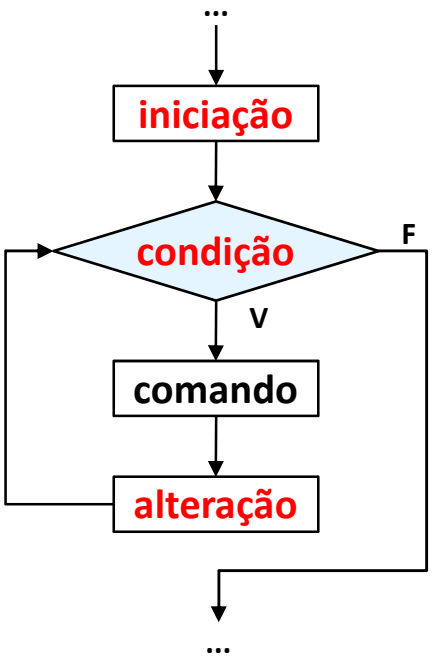
```
#include <stdio.h>
int main(void) {
    int x=5, y, z;
    y = x++ + 2;
    z = --x + 2;
    printf("x=%d y=%d z=%d\n",x,y,z); // x=5 y=7 z=7
    return 0;
}
```

Comandos de repetição

O comando `for`

repete um comando, um número **determinado** de vezes.

```
for(iniciação; condição; alteração) comando;
```



Exemplo 5. O comando `for`

Exibir uma contagem progressiva de **1** até **9**.

```
#include <stdio.h>
int main(void) {
    int i;
    for(i=1; i<=9; i++)
        printf("%d\n",i);
    return 0;
}
```

O comando **`for`** repete um comando **enquanto** sua condição for verdadeira!

Comandos de repetição

Exemplo 6. Tabuada

Dado um número **n** (entre **1** e **10**), exiba a sua tabuada.

```
#include <stdio.h>
int main(void) {
    int n;
    printf("Numero? ");
    scanf("%d",&n);
    for(int i=1; i<=10; i++)
        printf("%d x %2d = %3d\n",n,i,n*i);
    return 0;
}
```

Note que a **variável contadora** pode ser declarada dentro do próprio comando **for**!

Comandos de repetição

Exercício 1. Contagem regressiva

Dado um número natural **n**, exiba uma contagem regressiva de **n** até **0**.

Exercício 2. Cores

O programa a seguir deveria exibir cada número de cor em sua cor correspondente; porém, ele não está fazendo isso. Faça a correção necessária.

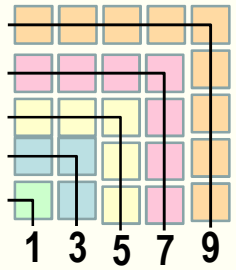
```
#include <stdio.h>
#include <conio.h>
int main(void) {
    int c;
    for(c=0; c<=15; c++)
        _textcolor(c);
        printf("Cor %d\n",c);
    return 0;
}
```

Comandos de repetição

Exercício 3. Quadrados perfeitos

O **quadrado** de um número natural **n** é igual à **soma** dos **n** primeiros ímpares consecutivos. Com base nessa ideia, crie um programa que, dado um número natural **n**, calcula e exibe o quadrado de **n**.

Por exemplo: $5^2 = 1 + 3 + 5 + 7 + 9 = 25$



Exercício 4. Potência

A **potência** de um número real **x** elevado a um número natural positivo **n** é igual ao **produto** de **n** fatores iguais a **x** (por definição, todo número elevado a **0** é **1**). Dados um número real **x** e um número natural **n**, calcule e exiba a potência x^n.

Exercício 5. Fatorial

O **fatorial** de um número natural **n** é igual ao **produto** dos **n** primeiros naturais positivos (por definição, o fatorial de **0** é **1**). Dado um número natural **n**, calcule e exiba o seu fatorial.

Exercício 6. Termial

O **termial** de um número natural **n** é igual à **soma** dos **n** primeiros naturais positivos (por definição, o termial de **0** é **0**). Dado um número natural **n**, calcule e exiba o seu termial.

Comandos de repetição

Exemplo 7. Tabela ASCII

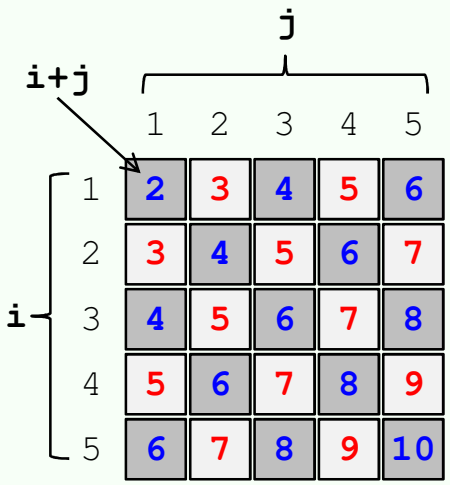
```
#include <stdio.h>
#include <conio.h>

int main(void) {
    for(int c=0; c<=255; c++) {
        printf("%3d => %c\n",c,c);
        if( c%16==15 ) {
            printf("\nPressione enter...");
            _getch();
            _clrscr();
        }
    }
    return 0;
}
```

Note que podemos usar comandos de seleção dentro de um comando de repetição!

Comandos de repetição

Exemplo 8. Tabuleiro de xadrez

```
#include <stdio.h>
#include <conio.h>
int main(void) {
    int n;
    printf("Tamanho? ");
    scanf("%d",&n);
    for(int i=1; i<=n; i++) {
        for(int j=1; j<=n; j++) {
            _textcolor((i+j)%2==0 ? 8 : 15);
            printf("%c%c",219,219);
        }
        putchar('\n');
    }
    return 0;
}
```



Note que podemos usar comandos de repetição dentro de outros comandos de repetição!

Comandos de repetição

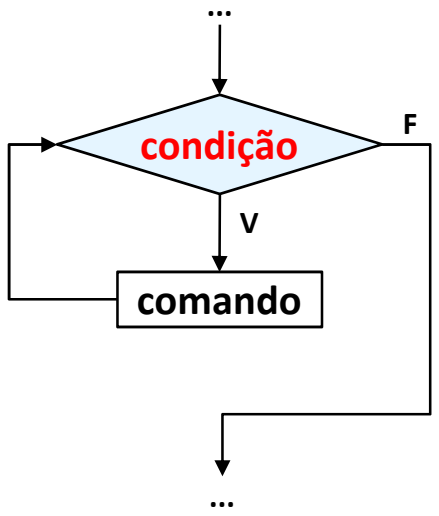
O comando `while`

repete um comando, um número **indeterminado** de vezes (zero ou mais vezes).

Exemplo 9. O comando `while`

Dado um número natural positivo, exiba os seus dígitos.

```
#include <stdio.h>
int main(void) {
    int n;
    printf("Numero? ");
    scanf("%d",&n);
    while( n>0 ) {
        printf("%d\n",n%10);
        n /= 10;
    }
    return 0;
}
```



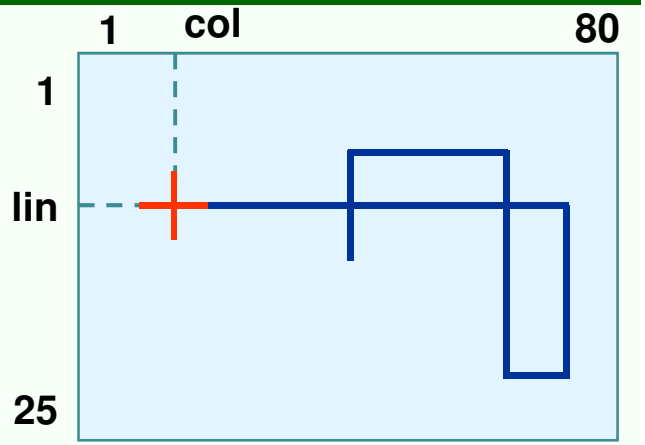

```
while( condição )
    comando;
```

O comando **`while`** testa sua condição **antes** de executar o comando a ser repetido!

Comandos de repetição

Exemplo 10. Rastros

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

int main(void) {
    int col=40, lin=12;
    while( 1 ) {
        _gotoxy(col,lin);
        _putch(219);
        switch( toupper(_getch()) ) {
            case 'N': if( lin> 1 ) lin--; break;
            case 'S': if( lin<24 ) lin++; break;
            case 'L': if( col<80 ) col++; break;
            case 'O': if( col> 1 ) col--; break;
            case 'F': exit(0);
        }
    }
    return 0;
}
```



Comandos de repetição

Exercício 7. Dígito verificador

Numa agência bancária, as contas são identificadas por números de seis dígitos mais um **dígito verificador** (vide exemplo). Dado um número natural **`n`**, exiba o número de conta correspondente.

Exemplo:

Seja **`n`** = **`7314`**.

1º Calcular a soma **`s`** dos dígitos de **`n`** (ou seja, **`s`** = **`7+3+1+4`** = **`15`**).

2º Calcular o resto **`r`** da divisão de **`s`** por **`10`** (ou seja, **`r`** = **`5`**).

Número de conta: **`007314-5`**

Exercício 8. Rastro

Adicione as seguintes opções no programa que desenha rastros:

- **`R`**: ativa/desativa o rastro (quando desativado, o cursor deve se mover sem deixar rastro).
- **`C`**: seleciona uma nova cor para o rastro (alternando de uma cor para a próxima, ciclicamente).

Exercício 9. Teclado

A função **`_kbhit()`**, declarada em **`conio.h`**, devolve verdade apenas quando alguma tecla é pressionada. Usando essa função, e o comando **`while`**, crie um programa que exibe a palavra **`TESTE`** enquanto nenhuma tecla for pressionada.

Comandos de repetição

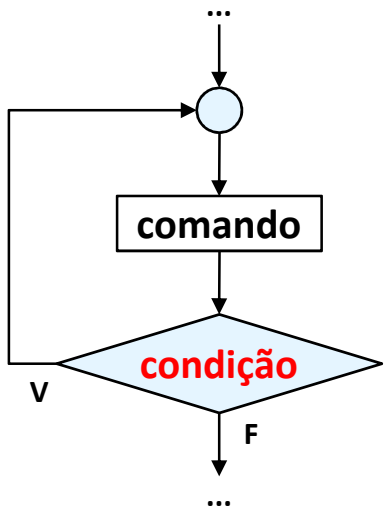
O comando `do-while`

repete um comando, um número **indeterminado** de vezes (uma ou mais vezes).

Exemplo 11. O comando `do-while`

Exibir a soma de uma sequência de números terminada com **0**.

```
#include <stdio.h>
int main(void) {
    int s=0, n;
    do {
       printf("Numero? ");
       scanf("%d",&n);
       s += n;
    } while( n!=0 );
    printf("Soma = %d\n",s);
    return 0;
}
```




```
do {
    comando;
} while( condição );
```

O comando **`do-while`** testa sua condição **após** executar o comando a ser repetido!

Comandos de repetição

Exemplo 12. Jogo de adivinhação

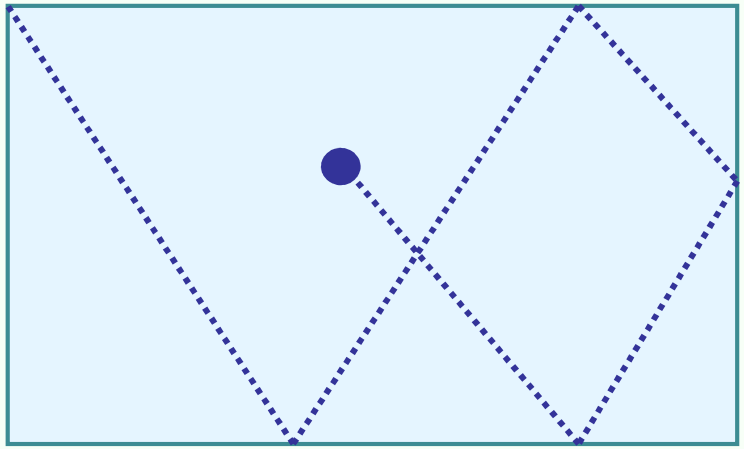
Sortear um número e verificar se o usuário consegue descobrir qual foi o número sorteado.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main(void) {
    srand(time(NULL));
    int c, n = rand()%7 + 1;
    do {
       printf("Chute entre 1 e 7: ");
       scanf("%d",&c);
       if( c>n ) puts("Muito alto!");
       else if( c<n ) puts("Muito baixo!");
       else puts("Voce acertou!");
    } while( n!=c );
    return 0;
}
```

As funções **srand()** e **rand()**, declaradas em **stdlib.h**, geram números aleatórios!

Comandos de repetição

Exemplo 13. Pingue-pongue



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

int main(void) {
    int x=1, y=1, dx=-1, dy=-1;
    do {
        _gotoxy(x,y);
        printf("O\b");
        _sleep(1); // pausa em segundos
        printf(".");
        if( x==1 || x==80 ) dx = -dx;
        if( y==1 || y==24 ) dy = -dy;
        x += dx;
        y += dy;
    } while( !_kbhit() );
    return 0;
}
```

Comandos de repetição

Exercício 10. Pingue-pongue

Altere o programa do pingue-pongue, de modo que a cor do rastro mude aleatoriamente cada vez que a direção do movimento da bolinha for modificada. Use números aleatórios.

Exercício 11. Máximo e mínimo de uma sequência

Dada uma sequência de números naturais (cujo último número é **0**), informe quais são os itens máximo e mínimo nessa sequência.

Exercício 12. Consistência de entrada

Dado um número real não negativo, informe a sua raiz quadrada. O programa deve rejeitar a entrada, enquanto não for digitado um número real não negativo.

Exercício 13. Caixa eletrônico

Simule o funcionamento de um caixa eletrônico, que oferece as seguintes opções ao cliente: **1** – *depósito*, **2** – *saque*, **3** – *saldo* e **4** – *sair*. Suponha que o saldo inicial do cliente é de **R$ 1000,00** e que ele não pode ficar negativo (se o usuário tentar efetuar um saque maior que o saldo corrente, a operação não deve ser efetuada e o usuário deve ser informado).

Comandos de repetição

O comando `break`

pode ser usado para interromper a execução de um comando de repetição.

Exemplo 14. Primalidade

Verificar se um dado número natural maior que **1** é **primo**.

```
#include <stdio.h>
int main(void) {
    int n, d;
    printf("Numero? ");
    scanf("%d",&n);
    for(d=2; d<=n-1; d++)
        if( n%d == 0 ) break;
    if( d==n ) puts("E primo!");
    else puts("Nao e primo!");
    return 0;
}
```

Um número natural maior que **1** é primo se e só se ele for divisível apenas por **1** e por ele mesmo.

Teste de primalidade:

```
7 % 2 == 1
7 % 3 == 1
7 % 4 == 3
7 % 5 == 2
7 % 6 == 1
```

∴ 7 é um número primo

```
9 % 2 == 1
9 % 3 == 0
```

∴ 9 não é um número primo

Existem algoritmos para teste de primalidade que são muito mais eficientes do que este!

Comandos de repetição

Exercício 14. Teste de primalidade

Um teste de primalidade um pouco mais eficiente considera o fato de que, *se o número não tem divisores menores ou iguais à sua raiz quadrada (arredondada para cima), então ele é primo*. Com base nisso, altere o programa anterior. Use as funções **`sqrt()`** e **`ceil()`**, de **`math.h`**.

Exemplos:

- Para ***n*** = **11**, temos a raiz **3.32**. Então, precisamos variar o divisor de **2** até no máximo **4**.

  11 % **2** == 1
  11 % **3** == 2
  11 % **4** == 3
  Portanto, 11 é um número primo!

- Para ***n*** = **1001**, temos a raiz **31.64**. Então, precisamos variar o divisor de **2** até no máximo **32**.

  1001 % **2** == 1
  1001 % **3** == 2
  1001 % **4** == 1
  1001 % **5** == 1
  1001 % **6** == 5
  1001 % **7** == 0
  Portanto, 1001 não é um número primo!

Fim