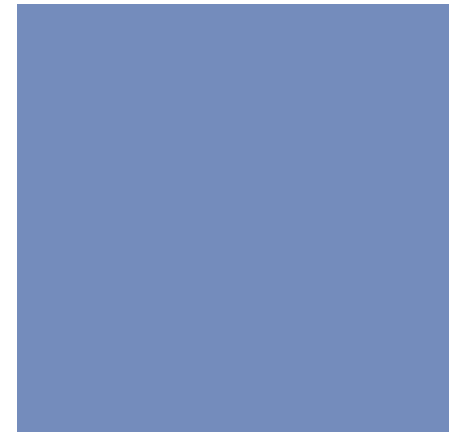
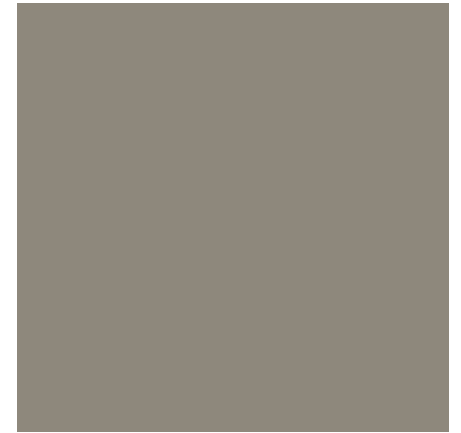




# Estruturas de Repetição III

booleano  
continue



INF110 – Programação I

Prof. Alcione/André Gustavo  
DPI/UFV – 2020/1



## + Erros comuns em loops

- Ex: Imprimir os números de 1 a 10
- Erros comuns
  - Esquecer de inicializar o contador
  - Errar a condição de parada do contador
  - Esquecer de incrementar o contador



# Estrutura de repetição `for`



- Sintaxe

- `for` ( inicialização ; condição ; incremento )  
comando;



# Estrutura de repetição `for`



- Sintaxe

- `for` ( inicialização ; condição ; incremento )  
comando;

- Equivalente a

- inicialização;

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



# Estrutura de repetição `for`

## ■ Sintaxe

- `for ( inicialização ; condição ; incremento )`  
    comando;

Ficam todos numa linha,  
evitando os erros comuns

## ■ Equivalente a

- inicialização;  
    `while (condição) {`  
        comando;  
        incremento;  
    `}`

# + Exemplo

- Ler um natural N e escrever todos os naturais de 1 a N, inclusive
- Ler um natural N e escrever todos os naturais de N até 1

```
int main() {  
    int i;  
    int N;  
  
    cin >> N;  
  
    for(i=1; i<=N; i++)  
        cout << i << " ";  
  
    return 0;  
}
```

```
int main() {  
    int i;  
    int N;  
  
    cin >> N;  
  
    for(i=N; i>=1; i--)  
        cout << i << " ";  
  
    return 0;  
}
```

# + Exemplo

- Ler dois inteiros A e B e imprimir todos os inteiros de A até B

```
int main() {  
    int i;  
    int A, B;  
  
    cout << "Contar de quanto ate quanto? ";  
    cin >> A >> B;  
  
    if (A<B)  
        for(i=A; i<=B; i++)  
            cout << i << " ";  
    else  
        for(i=A; i>=B; i--)  
            cout << i << " ";  
  
    return 0;  
}
```



# Exemplo



- Ler dois números inteiros A e B e verificar se entre A e B existe algum número divisível por 30
  - Se existir imprimir SIM
  - Senão, não imprimir nada





# Exemplo

- Ler dois números inteiros A e B e verificar se entre A e B existe algum número divisível por 30
  - Se existir imprimir SIM
  - Senão, não imprimir nada

- Obs.: note o uso do `break`

```
int main() {  
    int a, b;  
    int i;  
  
    cin >> a >> b;  
  
    for(i = a; i <= b; i++)  
        if (i % 30 == 0) {  
            cout << "SIM\n";  
            break;  
        }  
  
    return 0;  
}
```



# Exemplo



- Ler dois números inteiros A e B e verificar se entre A e B existe algum número divisível por 30
  - Se existir imprimir SIM
  - Senão, não imprimir nada
- Variação:
  - Se existir, imprimir SIM
  - Senão, imprimir NÃO



# Exemplo

- Ler dois números inteiros A e B e verificar se entre A e B existe algum número divisível por 30

- Se existir imprimir SIM
- Senão, não imprimir nada

- Variação:

- Se existir, imprimir SIM
- Senão, imprimir NÃO

- Obs.: note o uso de flag booleano

```
int main() {  
    int a, b;  
    int i;  
    bool tem = false;  
  
    cin >> a >> b;  
  
    for(i = a; i <= b; i++)  
        if (i % 30 == 0)  
            tem = true;  
  
    if (tem)  
        cout << "SIM\n";  
    else  
        cout << "NAO\n";  
  
    return 0;  
}
```



# Exemplo



- Ler uma lista de 10 notas e informar a média
  - As notas são inteiros no intervalo  $[0,100]$
- Informar também quantos aprovados ( $\text{nota} \geq 60$ )
- Informar se houve alguma nota 100



```
#include <iostream>
using namespace std;

int main() {
    int nota, total;
    int contap = 0;           //contador de aprovados
    bool nota100 = false;    //booleano para nota 100
    int i;                   //contador de notas

    total = 0;

    for(i=1; i<=10; i++) {
        cin >> nota;
        total += nota;
        if (nota >= 60)
            contap++;
        if (nota == 100)
            nota100 = true;
    }

    cout << "Media = " << total/10.0 << endl;
    cout << "Aprovados = " << contap << endl;
    if (nota100)
        cout << "Houve nota 100\n";
    else
        cout << "Nao houve nota 100\n";

    return 0;
}
```



# Exemplo



- Ler uma lista de 10 notas e informar a média
  - As notas são inteiros no intervalo  $[0,100]$
- Informar também quantos aprovados ( $\text{nota} \geq 60$ )
- Informar se houve alguma nota 100
- Informar ainda o valor da maior e da menor nota



```
int main() {
    int nota, total;
    int contap = 0;           //contador de aprovados
    bool nota100 = false;    //booleano para nota 100
    int i;                   //contador de notas
    int maior = 0;           //guardara a maior nota
    int menor = 100;         //guardara a menor nota

    total = 0;

    for(i=1; i<=10; i++) {
        cin >> nota;
        total += nota;
        if (nota >= 60)
            contap++;
        if (nota == 100)
            nota100 = true;
        if (nota > maior)
            maior = nota;
        if (nota < menor)
            menor = nota;
    }

    cout << "Media = " << total/10.0 << endl;
    cout << "Aprovados = " << contap << endl;
    if (nota100)
        cout << "Houve nota 100\n";
    else
        cout << "Nao houve nota 100\n";
    cout << "Maior nota = " << maior << endl;
    cout << "Menor nota = " << menor << endl;

    return 0;
}
```



# Exercício #1



- Ler um número natural  $N$  e imprimir todos os pares de 1 a  $N$
- Ler dois números naturais  $A$  e  $B$  e imprimir os pares entre eles (inclusive eles, se forem pares)





## Exercício #2



- Ler um número natural  $N$  e imprimir todos os seus divisores



## Exercício #3



- Ler um número natural  $N$  e informar se ele é primo ou não
  - Um número primo só tem como divisores 1 e o próprio número



## Exercício #4



- Ler dois números inteiros A e B e calcular o somatório dos números inteiros de A até B (inclusive A e B)

# + Estrutura de repetição (com condição pós-fixada)

## ■ Sintaxe

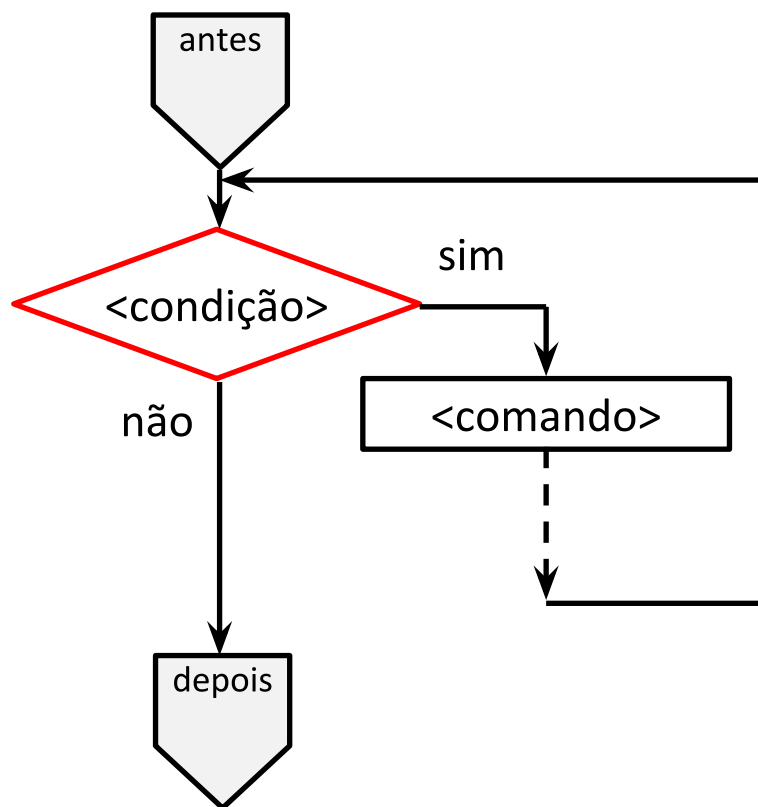
```
do {  
    bloco de comandos;  
while {condição};
```

## ■ Funcionamento

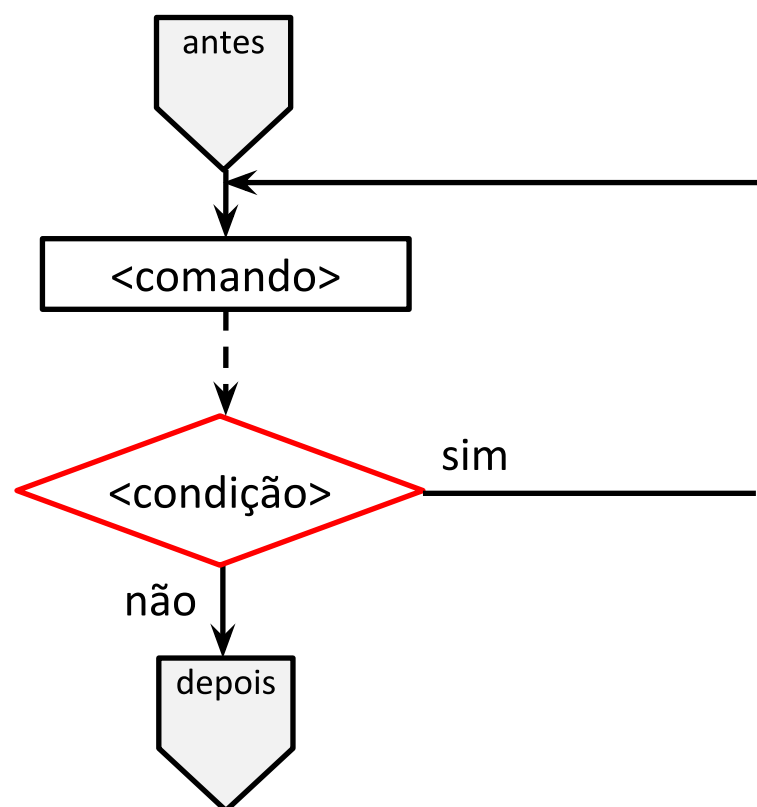
- Executa o bloco de comandos e depois efetua o teste condicional
- Caso a condição seja verdadeira, executa-se o bloco de comandos novamente e efetua o teste condicional
- Caso a condição seja falsa, o fluxo segue para a primeira instrução seguinte (após a condição)

# + while x do-while

## ■ while



## ■ do-while



# + while x do-while

## ■ Diferença

- `while` testa a condição **antes** de executar o bloco
- `do-while` testa a condição **depois** de executar o bloco

## ■ Semelhança

- Ambos repetem o bloco se a condição for verdadeira
- Ambos interrompem a repetição se a condição for falsa

+

Qual a saída?

```
int cont = 0;
while (cont < 10) {
    cout << cont;
    cont++;
}
```

```
int cont = 0;
do {
    cout << cont;
    cont ++;
} while (cont < 10);
```

```
int cont;
for(cont = 0; cont < 10; cont++)
    cout << cont;
```

## + Qual a saída?

```
int cont = 0;
while (cont < 10) {
    cout << cont;
}
```

```
int cont = 0;
do {
    cout << cont;
} while (cont < 10);
```

- Loop infinito nos dois casos





## Qual a saída?

```
int n = 3;
while (n != 0) {
    cout << "abc" << endl;
    n = 0;
    cout << "def" << endl;
}
cout << "ghi" << endl;
```

```
int n = 3;
do {
    cout << "abc" << endl;
    n = 0;
    cout << "def" << endl;
} while (n != 0);
cout << "ghi" << endl;
```



## Qual a saída?

```
int n = 3;
while (n == 0) {
    cout << "abc" << endl;
    cout << "def" << endl;
    n--;
}
cout << "ghi" << endl;
```

```
int n = 3;
do {
    cout << "abc" << endl;
    cout << "def" << endl;
    n--;
} while (n == 0);
cout << "ghi" << endl;
```

## + Instrução `continue`

- Interrompe imediatamente a iteração atual e passa à próxima



# + break x continue

- Ambos ignoram (não executam) o restante dos comandos do bloco
- `continue` volta imediatamente para a **condição** do loop
- `break` vai imediatamente para **fora** do loop



# Exemplo



- Ler uma lista de valores (terminada por 9999) e informar a raiz quadrada de cada valor (ignorar os negativos)

```
int main() {  
  
    int n;  
  
    do {  
        cin >> n;  
        if (n==9999) break;           //interrompe o do-while  
        if (n<0) continue;           //volta ao inicio do do-while  
        cout << sqrt(n) << endl;  
    } while(true);  
  
    return 0;  
}
```



# Exemplo

- Ler uma lista de valores (terminada por 0) e informar a média dos valores lidos (ignorar os negativos)

```
int main() {
    int valor;
    int total = 0;
    int cont = 0;

    do {
        cin >> valor;
        if (valor==0) break;    //interrompe o do-while
        if (valor<0) continue; //volta ao inicio do do-while
        total += valor;
        cont++;
    } while (true);

    if (cont==0)
        cout << "Nenhum valor para tirar media\n";
    else
        cout << "Media = " << (double)total/cont << endl;

    return 0;
}
```

# + Tratamento de entrada

- Estruturas de repetição podem ser usadas para tratar entrada
  - Solicitar novamente valores que foram entrados incorretos
  - Ou ignorar valores incorretos

```
int a;

cout << "Digite um valor entre 1 e 10: ";
cin >> a;
while(a<1 || a>10) {
    cout << "O valor deve estar entre 1 e 10, digite novamente: ";
    cin >> a;
}

cout << "Voce digitou " << a << endl;
```



# Exercício #1



- Escrever o alfabeto maiúsculo
  - ABCDE ... XYZ
- Escrever o alfabeto minúsculo de trás pra frente
  - zyx ... edcba





## Exercício #2



- Ler um número inteiro e escrevê-lo de trás pra frente
- Exemplos
  - $1234 \rightarrow 4321$
  - $51000 \rightarrow 00015$



## Exercício #3



- Fazer um programa para um jogo de adivinhação
  - Sortear um número qualquer de 0 a 9
  - Ler palpites do usuário até que ele descubra o número
  - Informar quantas tentativas foram feitas
- Ao final perguntar se quer jogar de novo (S/N)