

Curso de C

Controle de Execução

Controle de Execução

Objetivos:

- Aprender a:
 - Interromper
 - Reiniciar
 - Avançar e retroceder para pontos arbitrários

Controle de Execução

Roteiro:

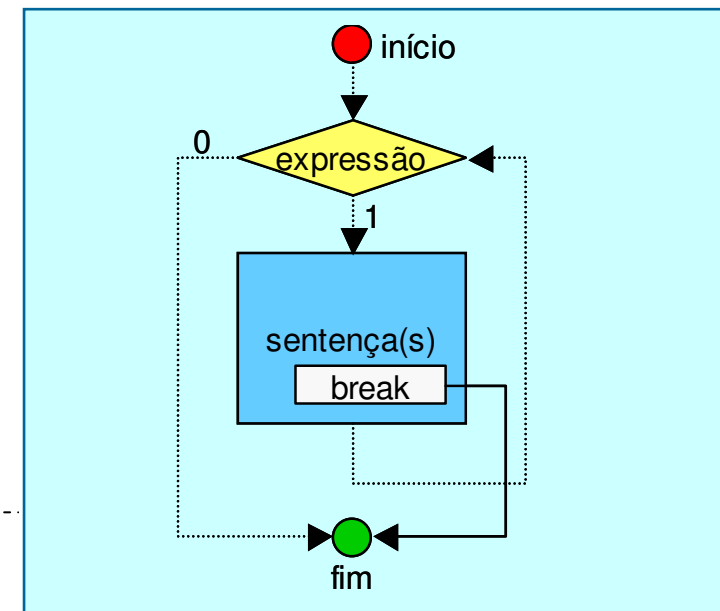
- Comando `break`
- Comando `continue`
- Comando `goto`

break

Objetivo do **break**:

- Cancelar execução:
 - **for** / **while** / **do...while**
- Comportamento:
 - Termina imediatamente o bloco
 - Não executa restante do bloco
 - Continua logo após o bloco
- Exemplos:
 - Terminar uma busca
 - Situações de erro
 - Evitar repetições

```
while (expressão) {  
    sentenças(s);  
    if (condição) {  
        break;  
    }  
    sentenças(s);  
}
```

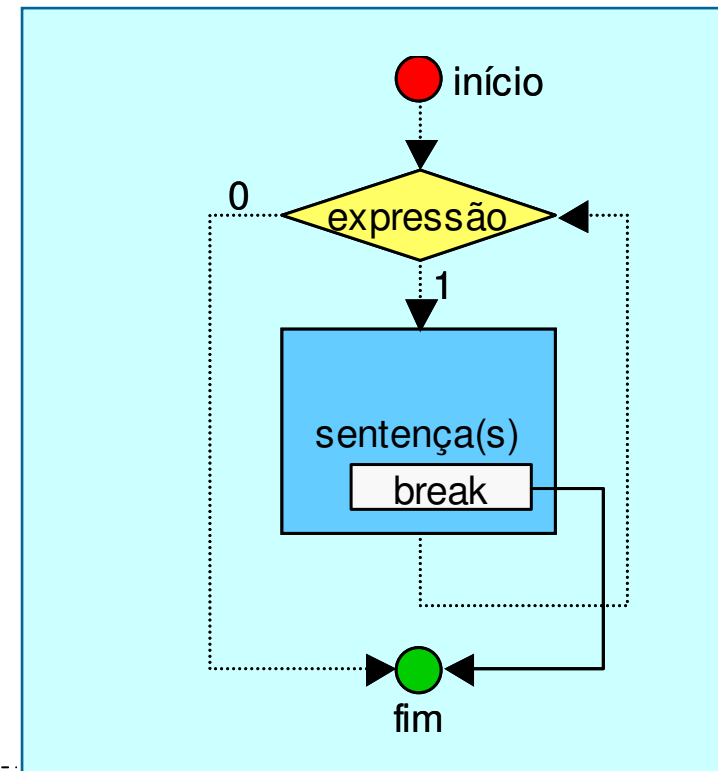


break

Sintaxe break com while

Sintaxe:

```
while (expressão) {  
    sentenças(s);  
    if (condição) {  
        break;  
    }  
    sentenças(s);  
}
```

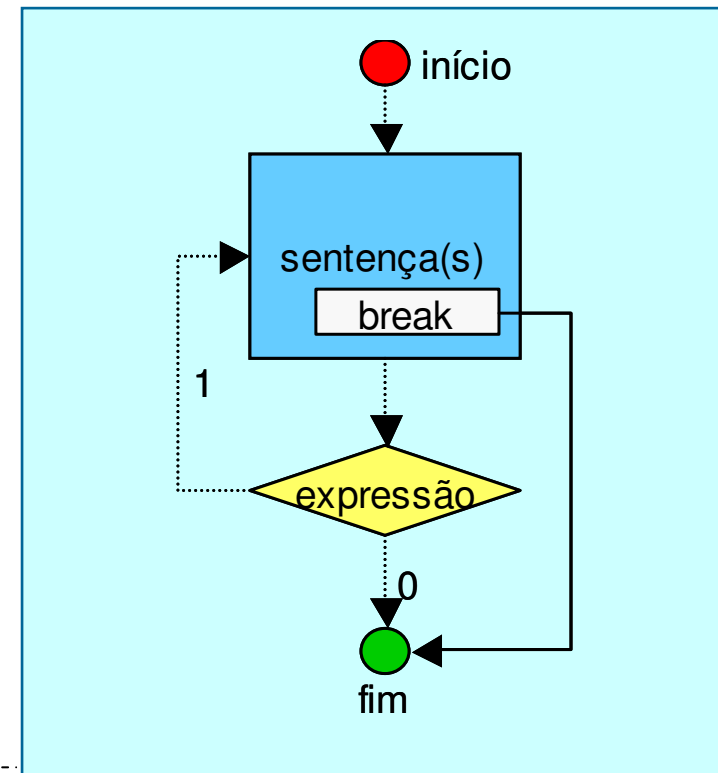


break

Sintaxe break com do...while

Sintaxe:

```
do {  
    sentenças(s);  
    if (condição) {  
        break;  
    }  
    sentenças(s);  
} while (expressão);
```

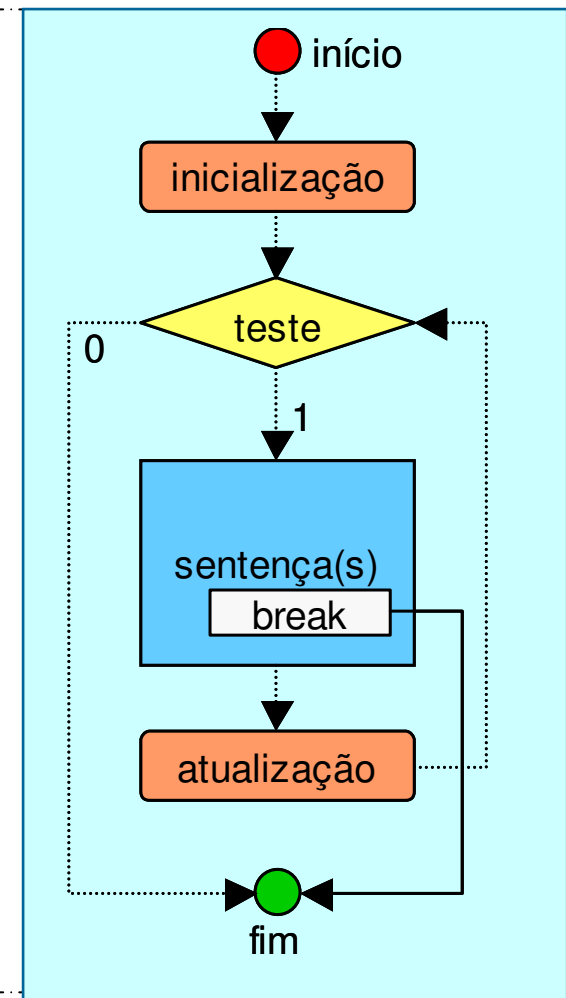


break

Sintaxe break com for ()

Sintaxe:

```
for (inicialização;  
    teste;  
    atualização) {  
    sentenças (s);  
    if (condição) {  
        break;  
    }  
    sentenças (s);  
}
```

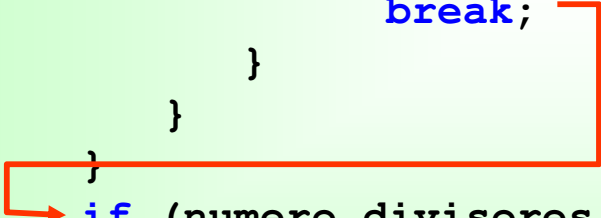


break

```
int main(int argc, char *argv[]) {
    int numero, divisor, resto, numero_divisores;

    printf("Digite o numero: ");
    scanf("%d", &numero);

    numero_divisores = 0;
    for (divisor = 1; divisor <= numero; divisor++) {
        resto = numero % divisor;
        if (resto == 0) {
            numero_divisores++;
            if (numero_divisores >= 3) {
                break;
            }
        }
    }
    if (numero_divisores == 2) {
        printf("O número %d é primo!\n", numero);
    }
    return 0;
}
```



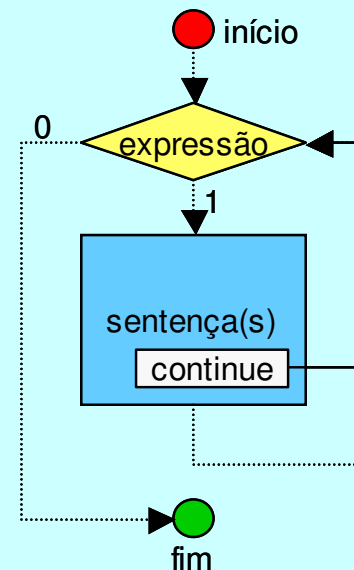
ControleExecucao\Divisores03\Divisores03.vcproj

continue

Objetivo do `continue`:

- Reiniciar execução:
 - `for` / `while` / `do...while`
- Comportamento:
 - Reinicia o bloco
 - Não executa resto do bloco
- Exemplos:
 - Pular valores inválidos
 - Evitar processamento

```
while (expressão) {  
    sentenças(s);  
    if (condição) {  
        continue;  
    }  
    sentenças(s);  
}
```

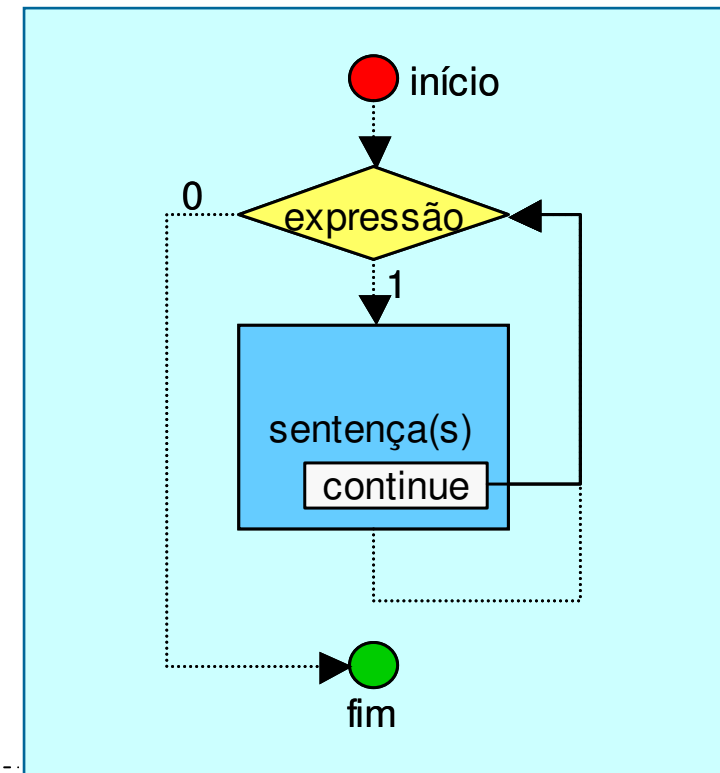


continue

Sintaxe continue com while

Sintaxe:

```
while (expressão) {  
    sentenças(s);  
    if (condição) {  
        continue;  
    }  
    sentenças(s);  
}
```

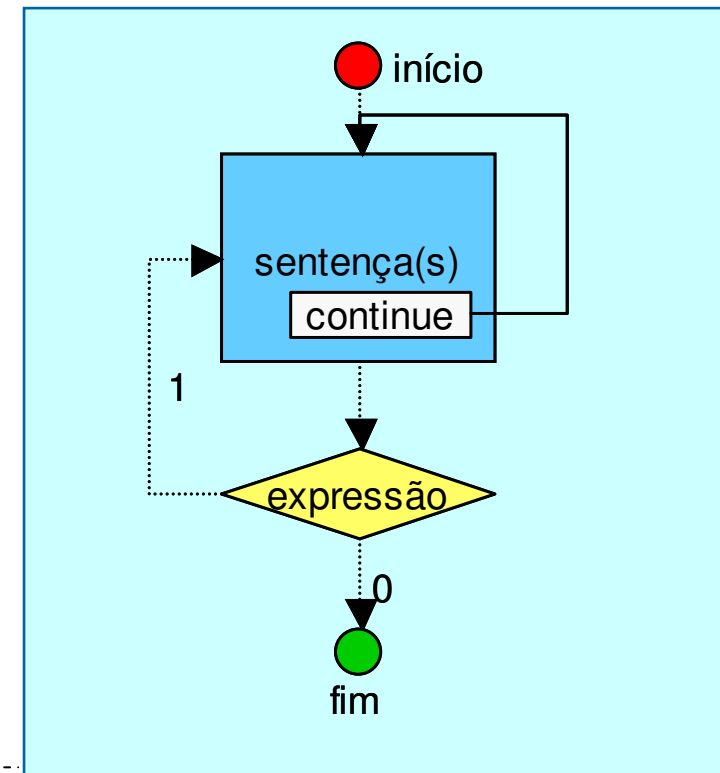


continue

Sintaxe continue com do...while

Sintaxe:

```
do {  
    → sentenças (s) ;  
    if (condição) {  
        continue; →  
    }  
    sentenças (s) ;  
} while (expressão) ;
```



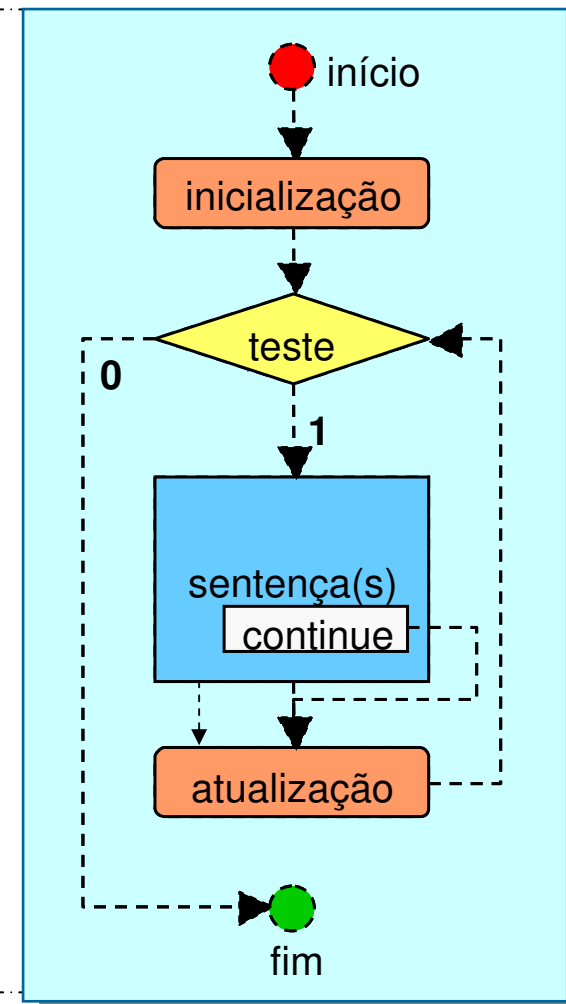
continue

Sintaxe continue com for

Sintaxe:

```
for (inicialização;  
    teste;  
    atualização) {  
    sentenças (s);  
    if (condição) {  
        continue;  
    }  
    sentenças (s);  
}
```

OBS: executa também a atualização!





continue

```
int main(int argc, char *argv[]) {  
    double angulo, tangente;  
    double pi = 3.1415926535897932384626433832795;  
  
    for (angulo = 0;  
        angulo <= 180;  
        angulo += 10.0) {  
        if (angulo == 90.0) {  
            continue;  
        }  
  
        tangente = tan((angulo/180)*pi);  
        printf("tan(%8.2f)=%8.2f\n", angulo, tangente);  
    }  
  
    return 0;  
}
```

ControleExecucao\Tangete01\Tangente01.vcproj

Controle de Execução

goto

goto

Objetivo do goto:

- Desviar execução para uma marca
- Saltos para pontos arbitrários
- Estrutura de repetição primitiva

Exemplo:
Repetição infinita

```
→ marca1:  
    ...  
    sentença(s);  
    ...  
    goto marca1;
```


goto

Sintaxe: goto

Retrocesso de execução:

Sintaxe:

```
sentença (s) ;
```

```
...
```

```
→ marca1 :
```

```
...
```

```
sentença (s) ;
```

```
...
```

```
goto marca1 ;
```

```
...
```

```
sentença (s) ;
```

Avanço de execução:

Sintaxe:

```
sentença (s) ;
```

```
...
```

```
goto marca2 ;
```

```
...
```

```
sentença (s) ;
```

```
...
```

```
→ marca2 :
```

```
...
```

```
sentença (s) ;
```


goto

```
int main(int argc, char *argv[]) {  
    int numero = 1;
```

→ **inicio_repeticao:**

```
    if (numero > 10) {  
        goto fim_repeticao;  
    }  
    printf("%d ", numero);  
    numero++;  
    goto inicio_repeticao;
```

→ **fim_repeticao:**

```
    return 0;
```

```
}
```

ControleExecucao\Goto01\Goto01.vcproj



goto

Uso do goto:

- Difícil visualizar os destinos do goto
- Oculta lógica de execução
- Programas tornam-se incompreensíveis!
- **Dica:** não use goto