



Programação Imperativa e Funcional

Estruturas de controle: decisão e repetição

Diego Bezerra

dfb2@cesar.school



Na aula anterior...

- Discussão sobre os conceitos básicos da linguagem C
 - Tipos e variáveis
 - Operadores lógicos, de comparação e matemáticos
 - Entrada e saída padrão

Objetivos da aula de hoje

- Revisar na prática o que já foi visto
- Estruturas Condicionais
 - if-else
 - switch
- Estruturas de Repetição
 - while
 - do while
 - for
- Desvios incondicionais
 - break
 - continue

Exercícios práticos Beecrowd



Estruturas de Controle

- Os programas são executados de forma **sequencial**
- **Estruturas de controle** permitem que o programador mude o fluxo de execução do programa
- C ainda suporta **goto** (mas não é indicado seu uso)

```
int main() {  
    int opcao;  
    REFAZ: printf("\n\n Escolha uma opcao entre 1 e 5: " );  
    scanf("%d", &opcao);  
    if ((opcao > 5) || (opcao < 1) )  
        goto REFAZ;  
    return 0  
}
```

Desvio Condicional: if-else

- Não é possível utilizar o comando **else** sozinho
- O corpo dos comandos **if** e **else** é limitado por '{' e '}'
 - É possível omitir, desde que só contenha um comando
- A condição **sempre** é limitada por '(' e ')'

```
if(expressão booleana) {  
    comando1;  
}
```

```
if (expressão booleana) {  
    comando1;  
} else {  
    comando2;  
    comando3;  
}
```


Desvio Condicional: if-else



```
double nota1=3.5, nota2=7.3, nota3=9.5;
double media = (nota1 + nota2 + nota3) / 3.0;

if ( media < 5 ){
    printf("Reprovado");
} else {
    printf("Aprovado");
}
```

Desvio Condicional: if-else




```
int x=2, y=0, z=0, a=0;
if ( x > 3 )
    y = 2;
    z+= 8;
    a = y+x;
```

É uma má prática de programação usar **if-else** sem chaves!

```
int x = 1;
if (x == 3){ //faz nada }
else if(x>4) { printf(">4"); }
else if(x<2) { printf("<2"); }
else { printf("else"); }
```

É possível encadear comandos **if-else**, mas não é adequado que seja algo muito extenso!

Desvio Condicional: if-else



```
if ( media < 5 )  
    printf("Reprovado");  
else  
    printf("Aprovado");
```

Não recomendado!

```
if ( media < 5 )  
    printf("Aprovado");  
printf("%s", media);  
else  
    printf("Aprovado");
```



Desvio Condicional: switch



```
int x = 3;
switch(x) {
    case 1:
        printf("x vale %d", 1);
        break;
    case 2:
        printf("x vale %d", 2);
        break;
    case 3:
        printf("x vale %d", 3);
        break;
    default:
        printf("Não sei quanto vale x");
}
```

Desvio Condicional: switch




- O valor da expressão em cada **case** deve ser único
- É possível delimitar o corpo de cada **case** com chaves
- A cláusula **default** não precisa vir por último, mas essa é a forma mais convencional de usar
- Uso de **break** é importante
- É executado tudo a partir do **case** válido ou **default** (caso nenhum **case** seja válido) até a condição de parada

Desvio Condicional: switch



- Dentro das cláusulas **case** é possível ter outros blocos **switch**, **if-else**, laços de repetição, etc...
- **switch** não substitui totalmente o uso de **if-else** encadeado
 - **switch** só verifica **igualdade**
 - Com **if-else** é possível utilizar outros operadores de comparação

Desvio Condicional: switch

```
int x = 3;
switch(x) {
    case 1:
        printf("x vale %d", 1);
        break;
    case 2:
        printf("x vale %d", 2);
        break;
case k: 
        printf("x vale %d", 3);
        break;
    default:
        printf("Não sei quanto vale x");
}
```

Laço de Repetição: while

```
while (<expressão booleana>){  
    //comandos  
}
```

Teste é feito no início


```
int contador = 0;  
while (contador < 10) {  
    printf("%d", contador);  
    contador++;  
}
```

```
while ( 1 ){  
    //fazer alguma coisa  
}
```

Loop infinito

Se não usar chaves, só executará um comando.

Laço de Repetição: do-while



```
do{  
    //comandos  
}while (<expressão booleana> );
```

Teste é feito no **final**, o que significa que o loop executará **pelo menos 1 vez**

```
int contador = 0;  
do{  
    printf("%d", contador);  
    contador++;  
}while (contador < 10);
```

Se não usar chaves, só executará um comando.

Laço de Repetição: for



Pode ser declaração e inicialização ou somente inicialização

```
for (<inicialização> ; <condição> ; <incremento> ) {  
    //comandos  
}
```

Se a condição for **false**,
o laço é “quebrado”

É executado depois de
comandos

Laço de Repetição: for



```
for(int contador = 0 ; contador < 10 ; contador++ ){  
    printf("%d",contador);  
}
```

Laço de Repetição: for

- Campos não são obrigatórios

Loop infinito

```
for (;;) {  
    printf("executou for");  
}
```

```
int contador = 0;  
for (; contador < 10; ) {  
    printf("%d", contador);  
    contador++;  
}
```

De maneira equivalente ao **while**

Laço de Repetição: for

- É possível declarar mais de uma variável

```
for(int i=0, j=0; (i<10); i++, j++){  
    printf("i vale %d %d", i, j);  
}
```

Desvio Incondicional: break



- Saída imediata da estrutura de controle na qual a instrução é executada
- Além de ser usado em **switch**, pode ser usado em **while**, **for** e **do-while**

```
for(int i=1; i<10; i++){  
    if(i%2 == 0) break;  
    printf("i vale %d", i);  
}
```

Desvio Incondicional: continue

- Interrompe a execução da iteração corrente e avalia a condição de repetição
- Usada em laços de repetição: **while**, **for** e **do-while**
 - Não se usa em **switch**

```
for(int i=1; i<10; i++){  
    if(i%2 == 0) continue;  
    printf("i vale %d", i);  
}
```

Referências



- Rangel Netto, J. L. M., Cerqueira, R. D. G., & Celes Filho, W. (2004). **Introdução a estrutura de dados: com técnicas de programação em C.**

