



LP1

Prof. Luciano Bernardes de Paula



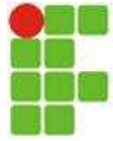
Estruturas de repetição (laços de repetição)

São comandos usados quando é necessário que uma instrução (ou conjunto de instruções) seja repetida diversas vezes.

Em C existem:

for → indicado quando **sabemos** quantas vezes deve se repetir

while e *do-while* → indicados quando **não sabemos** quantas vezes o trecho se repetirá

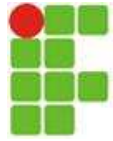


Laço *for*

Geralmente usado para repetir instruções por um número fixo de vezes (mesmo que esse número fixo seja variável).

Exemplos de problemas a serem resolvidos com o laço *for*:

- Repetir uma instrução 10 vezes;
- Apresentar uma contagem até 1000;
- Apresentar uma tabuada na tela;
- Apresentar uma frase N vezes, sendo N informado pelo usuário.



O laço ***for*** necessita de uma variável auxiliar, para contabilizar as execuções.

Forma genérica

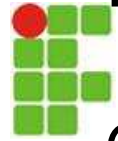
...

```
for(inicialização da var.; condição; incremento ou decremento){
```

```
...
```

```
}
```

```
...
```



Exemplo de código

O laço for necessita de uma variável auxiliar, para contabilizar as execuções.

...

```
int i;
```

...

```
for(i = 0; i < 10; i++){
```

```
    printf("Testando! Valor de i = %d", i);
```

```
}
```

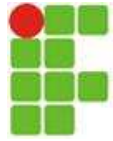
...



Se houver **somente uma instrução**, pode ser escrito:

```
for(i = 0; i < 10; i++) printf("Testando!!");
```

Para mais de uma instrução é preciso escrever o bloco completo (dentro de { }).



Não é preciso inicializar a variável auxiliar se for utilizar o valor atual de mesma.

```
for(i; i < 10; i++) printf("Testando!!");
```

O laço acima usará o valor atual de i.



Outro exemplo: repita na tela a palavra “teste” n vezes, sendo o valor de n informado pelo usuário.

```
for(i = 0; i < n; i++) printf("Teste!!\n");
```




É possível usar caracteres

```
for(ch = 'a'; ch <= 'z'; ch++){  
    ...  
}
```



É possível fazer o laço **for** de forma regressiva.

```
for(i = 10; i > 0; i--){  
    printf("%d\n", i);  
}
```

Ou alterando a forma de incremento

```
for(i = 0; i < 10; i = i + 2){  
    printf("%d\n", i);  
}
```



Laços aninhados

```
for(i = 0; i < 10; i++){  
    for(j = 0; j < 10; j++){  
        ...  
    }  
    ...  
}
```




Laço *while*

Indicado para repetir instruções por um número **imprevisível** de vezes.

Exemplos de problemas a serem resolvidos com o laço `while`:

- Repetir uma instrução enquanto o usuário entra com a resposta errada;
- Repete enquanto o usuário entra com uma letra minúscula;
- etc

O laço **while** faz um teste inicial; caso o mesmo resulte em  **verdadeiro**, as instruções são executadas e, após isso, a expressão é avaliada novamente.

Repete-se esse processo até que o teste resulte **falso**.

Forma genérica

...

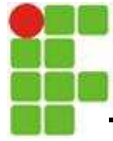
```
while(exp){
```

...

```
}
```

...

Exemplo de código



```
int num;
```

```
...
```

```
scanf("%d", &num);
```

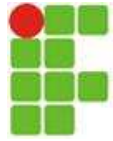
```
while(num < 10){
```

```
    printf("O valor deve ser menor que 10.");
```

```
    printf("Entre com um novo valor: ");
```

```
    scanf("%d", &num);
```

```
}
```



Se houver **somente uma instrução**, pode ser escrito:

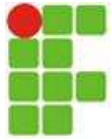
```
while(num < 10) scanf("%d", num);
```

Mais de uma instrução é preciso o bloco completo
(dentro de { }).



Laço **do-while**

O funcionamento do laço **do-while** é igual ao **while**, com a diferença que o teste é feito no final.



Exemplo de código

do {

```
    scanf("%d", &num);
```

```
    if(num >= 10){
```

```
        printf("O valor deve ser menor que 10.");
```

```
        printf("Entre com um novo valor: ");
```

```
    }
```

```
} while(num < 10);
```



Uma diferença entre o while e o do-while que também pode ser citada é:

while: se o teste falhar na primeira avaliação, o código do corpo nunca é executado.

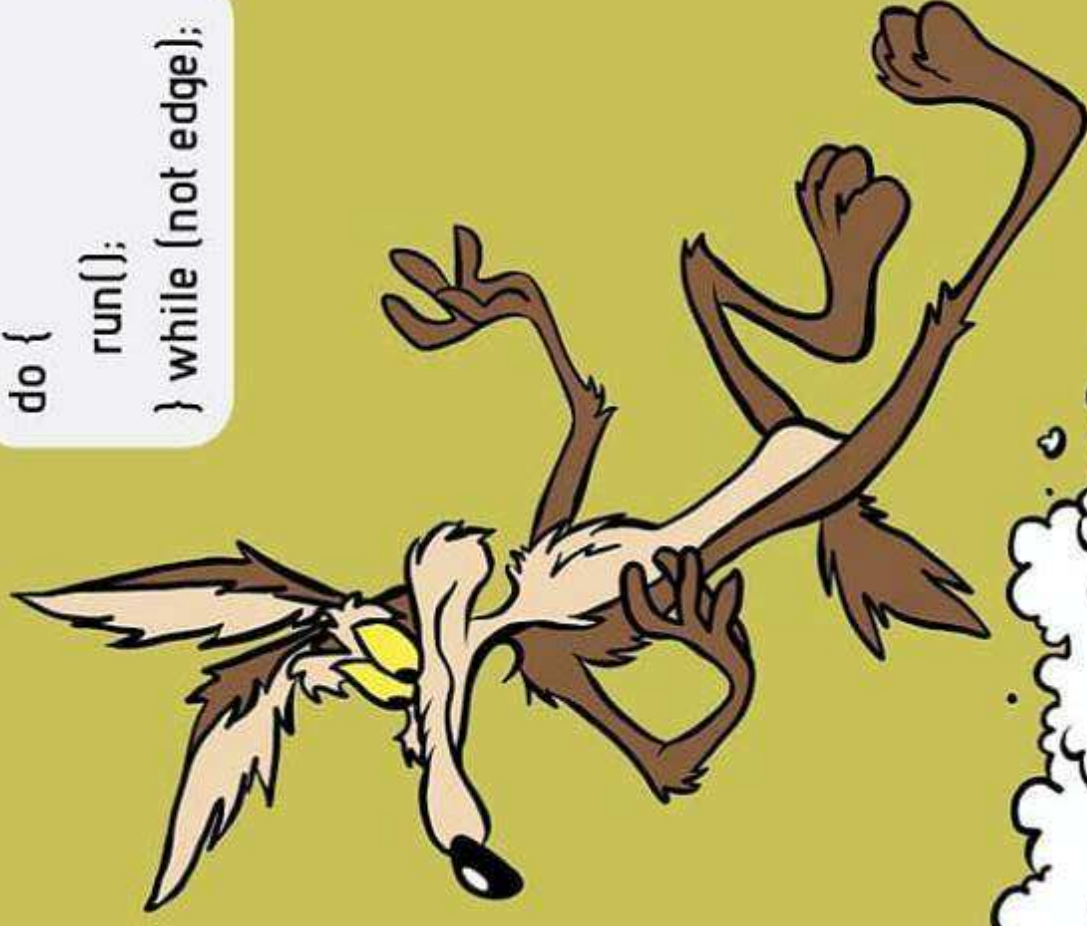
do-while: o corpo sempre é executado pelo menos uma vez, pois o teste é feito no final.

```
while (not edge) {
```

```
  run();
```

```
}
```

```
do {  
  run();  
} while (not edge);
```





Comandos **break** e **continue**

São comandos que podem pertencer ao corpo de um laço **for**, **while** ou **do-while**.

O comando **break** faz com que um laço seja terminado imediatamente.

O comando **continue** força a execução da próxima iteração do laço.



Exemplo

```
for(i = 0; i < 10; i++){  
    if((i == 3) || (i == 5)) continue;  
  
    j = j + i;  
  
    printf("j = %d\n", j);  
  
}
```



```
while(x < 10){
```

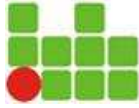
```
    j = j + x;
```

```
    if(j == 7) break;
```

```
    x++;
```

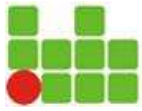
```
}
```

Exemplo de laço do-while com switch-case



```
printf("Entre com um valor entre 0 e 2: ");

do{
    scanf("%d", &valor);
    switch(valor){
        case 0:
            printf("Opção 0 escolhida...");
            break;
        case 1:
            printf("Opção 1 escolhida...");
            break;
        case 2:
            printf("Opção 2 escolhida...");
            break;
        default:
            printf("Por favor, entre com uma opção entre 0 e 2");
    }
} while((valor < 0) || (valor > 2));
```



Exercícios.