Estrutura de Repetição

Prof.: André Bezerra

A **Wikipédia** diz que:

Estrutura de repetição, na ciência da computação, é uma estrutura de desvio do fluxo de controle presente em linguagens de programação que realiza e/ou repete diferentes algoritmos/ações dependendo se uma condição é verdadeira ou falsa, em que a expressão é processada e transformada em um valor booleano. Estão associados a uma estrutura de repetição uma condição (também chamada "expressão de controle" ou "condição de parada") e um bloco de código: verifica-se a condição, e caso seja verdadeira, o bloco é executado. Após o final da execução do bloco, a condição é verificada novamente, e caso ela ainda seja verdadeira, o código é executado novamente.

Bom, o que veremos nesta aula?

- Estrutura de Repetição em C
  - while
  - do-while
  - for
- Comandos de Parada
  - break
  - continue

while

O comando **while** repetirá um bloco de instruções enquanto uma condição for verdadeira.

- Forma Geral

```
while (condicao) {
  /*do this*/
}
```

Exemplo 01

```c
#include <stdio.h>

int main(){
   int x, y;
   scanf("%d", &x);
   scanf("%d", &y);
   while (x < y){
       scanf("%d", &x);
   }
   return 0;
}
```

Exemplo 02

```c
#include <stdio.h>

int main(){
  int x = 4;
  while (x--){
     printf("%d\n", x);
  }
   return 0;
}
```

Exercício 01

- Escreva uma função que imprima todos os caracteres da tabela ASCII

Resposta Exercício 01

```c
#include <stdio.h>

void imprimeTabelaASCII(){
   int i = 0;
   while (i <= 255){
       printf("%d = %c", i, i);
       i++;
   }
}

int main(){
   imprimeTabelaASCII();
}
```

do .. while

- O comando **do-while** é bastante semelhante ao comando **while** visto anteriormente. Sua principal diferença é com relação a avaliação da condição: enquanto o comando **while** avalia a condição para depois executar uma sequência de comandos, o comando **do-while** executa uma sequência de comandos para depois testar a condição.

Forma Geral

```
do {
  /*executa instruções*/
} while (condicao);
```

Exemplo 03

```c
#include <stdio.h>
int main(){
  int num;
  do {
     scanf("%d", &num);
  } while (num == 5);
  return 0;
}
```

Exercício 02

- Leia uma idade e obriga-a ser válida. Vamos considerar uma idade válida entre 0 e 120 anos.

Resposta Exercício 02

```c
#include <stdio.h>

int main(){
  float idade;
  do {
     scanf("%f", &idade);
  } while ((idade < 0) || (idade > 120));
  return 0;
}
```

Resposta (com função)

```c
#include <stdio.h>
int validaIdade(float idade){
  return ((idade >= 0) && (idade <= 120));
}
int main(){
  float idade;
  do {
      scanf("%f", &idade);
  } while (!(validaIdade(idade)));
  return 0;
}
```

for

- Forma Geral

```
for (inicialização, condição de parada, incremento)
```

```
for (i = 0; i < 50; i++){
  /* repeat this 50 times */
}
```

Exemplo 04

```c
#include <stdio.h>

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

Exemplo 05

```c
#include <stdio.h>

int main(){
  int i;
  for (i = 0; i < 10; ){
     printf("%d\n", i);
  }
   return 0;
}
```

O que ocorre ao executar este código?

Exemplo 06

```c
#include <stdio.h>

int main(){
  int i;
  for (i = 0; ; i++){
     printf("%d\n", i);
  }
   return 0;
}
```

E agora?

Exemplo 07

```c
#include <stdio.h>

int main(){
  int i;
  for (; i < 10; i++){
     printf("%d\n", i);
  }
   return 0;
}
```

????

break

- O comando **break** é utilizado para terminar rapidamente uma repetição.
- Por exemplo, se estivermos em uma repetição e um determinado resultado ocorrer, o programa deverá sair da iteração.

Exemplo 08

```c
#include <stdio.h>

int main(){
  int i;
  for (i = 0; i < 10; i++){
      if(i == 5)
        break;
      printf("%d\n", i);
  }
  return 0;
}
```

Exercício 04

- Escrever uma função que retorne o primeiro número divisível por 17 após o 17.

Resposta

```c
#include <stdio.h>

int main(){
  int i;
  for (i = 18; ;i++){
    if (!(i % 17))
      break;
  }
  printf("%d", i);
}
```

continue

- Em uma estrutura de repetição, os comandos que sucedem o comando **continue** no bloco não são executados.
- O laço não para, volta ao inicio.

Exemplo 09

```c
#include <stdio.h>

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

- Exercícios da Lista II