Aula 08

MC 102 - Algoritmos e Programação de Computadores

Comandos de repetição II: break, continue e for.

while

Entendendo mais sobre o comando while

```
soma = 0;
printf("Quantos numeros devo somar: ");
scanf("%d", &n);
i = 1;
while (i <= n) {
   printf("%do.numero: ", i);
   scanf("%d", &x);
   soma += x;
   i++;
}
```

Os três principais componentes do while.

while

Entendendo mais sobre o comando while

```
soma = 0;
printf("Quantos numeros devo somar: ");
scanf("%d", &n);
i = 1;
while (i <= n) {
   printf("%do.numero: ", i);
   scanf("%d", &x);
   soma += x;
   i++;
}
```

1º: Inicialização da variável para o comando de repetição
2º: Condição de parada
3º: Incremento/decremento/alteração do valor da variável de loop

do-while

Um pouco mais sobre do-while

```
soma = 0;
printf("Entre com os valores por linha.\n");
printf("Digite 0 para finalizar.\n");
do {
   scanf("%d", &x);
   if (x != 0)
      soma += x;
} while (x != 0);
```

Os principais componentes do do-while.

do-while

Um pouco mais sobre do-while

```
soma = 0;
printf("Entre com os valores por linha.\n");
printf("Digite 0 para finalizar.\n");
do {
   scanf("%d", &x);
   if (x != 0)
      soma += x;
} while (x != 0);
```

1º: Inicialização da variável para o comando de repetição
2º: Incremento/decremento/alteração do valor da variável de loop
3º: Condição de parada.

Repetição

Como terminar a repetição antes da condição de parada?

Escrever um programa que some números inteiros até que 0 seja digitado ou até que 10 números sejam somados.

while com **break**

Somando até 10 numeros ou até 0 ser digitado

```
soma = 0;
printf("Digite 10 numeros (0 p/ finalizar).\n");
i = 1;
while (i <= 10) {
   printf("%do. numero: ", i);
   scanf("%d", &x);
   if (x == 0)
      break;
   soma += x;
   i++;
}
```

O comando **break** também pode ser usado para interromper um **while** ou um **do-while**.

while com **break**

Somando até 10 numeros ou até 0 ser digitado

```
soma = 0;
printf("Digite 10 numeros (0 p/ finalizar).\n");
i = 1;
while (i <= 10) {
   printf("%do. numero: ", i);
   scanf("%d", &x);
   if (x == 0)
      break;
   soma += x;
   i++;
}
```

O comando **break** também pode ser usado para interromper um **while** ou um **do-while**.

Repetição

Como reiniciar a repetição a partir de um determinado ponto do loop, evitando que os demais comandos sejam executados?

Escrever um programa que some números inteiros positivos até que 0 seja digitado.

do-while com **continue**

Somando positivos até que 0 seja digitado

```
soma = 0;
printf("Digite os numeros (0 p/ finalizar).\n");
do {
  scanf("%d", &x);
  if (x < 0) {
    printf("Valor negativo invalido.\n");
    continue;
  }
  soma += x;
} while (x != 0);
```

O comando **continue** reinicia o **while** ou o **do-while** evitando a execução dos comandos seguintes ao continue.

do-while com **continue**

Somando positivos até que 0 seja digitado

```
soma = 0;
printf("Digite os numeros (0 p/ finalizar).\n");
do {
  scanf("%d", &x);
  if (x < 0) {
    printf("Valor negativo invalido.\n");
    continue;
  }
  soma += x;
} while (x != 0);
```

O comando **continue** reinicia o **while** ou o **do-while** evitando a execução dos comandos seguintes ao continue.

Uso comum de repetição

Muitas vezes, precisamos fazer um loop onde o que realmente importa é a condição. O incremento/decremento pode ser realizado em qualquer parte do loop.

```
...
i = 1;
while (i < n) {
  ... comandos ...
  i++;
}
```

Comando **for**

Realiza a repetição de um comando ou de um bloco de comandos da mesma forma que um while, concentrado apenas na condição de parada.

```
for (inicialização ; condição ; passo )
    comando;
ou
for (inicialização ; condição ; passo ) {
    comandos;
}
```

inicialização: uma atribuição inicial para a variável de loop
condição: uma condição de parada
passo: incremento / decremento da variável de loop

Exemplo de for

Imprimir todos os números de 1 a 10.

Usando **while**

```
i = 1;
while (i<=10) {
  printf("%d ", i);
  i++;
}
```

Usando **for**:

```
for (i=1; i<=10; i++) {
  printf("%d ", i);
}
```


for ou while?

Quando usar for e quando usar while?

- Para casos em que o **fim é determinado**:
  - Usar **for**
    - Para c=1 até 100, faça ...
    - Para i=1 até n, faça ...
- Para casos em que o **fim é indeterminado**:
  - Usar **while**
    - Enquanto x diferente de 0, faça ...
    - Enquanto x > 0, some x ...

Exercício 1

Escreva um programa que imprima os N primeiros números ímpares. Note que o N deve ser informado pelo usuário.

Exercício 1

Algoritmo

Obtenha n;
Se n > 0 **então**
    impar = 1;
    Para i = 1; enquanto i <= n ; incremente i ;
        imprimar impar;
        impar = impar + 2;
Senão
    imprima “O numero informado deve ser > 0”;

Exercício 1

```
#include <stdio.h>
int main(int argc, char **argv){
int impar, i, n;
 scanf(%d, &n);
 if (n > 0) {
      impar = 1;
      for (i = 1; i <= n; i++) {
          printf("%d\n", impar);
          impar += 2;
 }
 else
     printf(“n informado deve ser > 0\n”);
 return 0;
}
```


Exercício 2

Escreva um programa que imprima todos os números primos menores que n

Exercício 2

```
#include <stdio.h>
int main(int argc, char **argv){
int num, aux, i, limite;

printf("Informe o numero limite:\n");
scanf(%d, &limite);
for (num = 2; num < limite; ++num) {
     aux = 2;
     while (num % aux != 0)
         ++aux;
     if (aux == num) {
          ++cont;
          printf("%d\n", num);
     }
}
printf(“Existem %d num. primos menores que %d\n” cont,
   limite);
return 0;
}
```