1. Comando de Repetição

Antes de iniciarmos os comandos de repetição, é importante relembrarmos de que a função ***loop()*** da estrutura básica do código, permanece em execução durante o período em que o código está sendo executado no Arduino (ligado). Então cuidado para gerar repetições de repetições desnecessárias.

1.1. while()

Os loops da instrução ***while()*** deverão rodar, até que a expressão torne-se falsa.

Sintaxe:

```
while(expressão){
  // instrução(s)
}
```

Exemplo 01: Neste exemplo a expressão de controle é "**contador <= 10**", como está dentro da função loop(), teremos duas repetições. Após o loop interno (while), será executado uma nova repetição pela função ***loop().***

```
int contador = 0;
void setup()
{
      Serial.begin(9600);
}
void loop()
{
            Serial.println("Iniciando a contagem While");
            while(contador <= 10){
                  Serial.println("Contador:"+String(contador));
                  contador++;
                  delay(500);
            }
            contador = 1;
delay(500);
}
```

Exemplo 02: Para visualizarmos de forma mais dinâmica o controle do loop, analise o código e observe o resultado das variáveis ***cont_loop*** e ***cont_while***.

```
int cont_while = 1;
int cont_loop = 1;
void setup()
{
        Serial.begin(9600);
}
void loop()
{

                                Serial.println("Contador loop(): "+String(cont_loop));
                                   while(cont_while <= 10){
                                        Serial.println("Contador while: "+String(cont_while));
                                        cont_while++;
                                         delay(500);
                                }
                                cont_loop++;
                                cont_while = 1;
delay(500);
}
```

Exemplo 03: Caso tenha a necessidade de realizar alguma interrupção durante o loop do **while()**, poderá utilizar a instrução "***break***".

```
int controle = 1;
void setup()
{
        Serial.begin(9600);
}
void loop()
{
                        Serial.println("Contando:");
                        while(true){
                                Serial.println(String(controle));
                                controle++;

                        if(controle >10){break;}
                        delay(500);
                        }
```

```
                controle = 0;
}
```

1.2. Do ... while()

Similar ao comando ***while()***, mas executa o teste no final da estrutura.

Sintaxe:

```
do {
  // instrução(s)
} while(expressão){
```

Exemplo 04: Neste caso, o teste condicional será realizado após a execução das instruções dentro do loop, isso garante a execução no mínimo de uma vez o código. Este exemplo é adaptação do exemplo 01.

```
int contador = 0;
void setup()
{
      Serial.begin(9600);
}
void loop()
{
            Serial.println("Iniciando a contagem While");
             do {
                  Serial.println("Contador:"+String(contador));
                  contador++;
                  delay(500);
             } while(contador <= 10);
            contador = 1;
delay(500);
}
```

1.2. For()

Os loops do ***for()***, irão rodar conforme os parâmetros de inicialização, expressão e incremento.

Sintaxe:

```
for (inicialização; expressão; incremento) {
        //// instrução(s);
}
```

Exemplo 05: A proposta é realizar a impressão dos números 1 até 10.

```
void setup()
{
      Serial.begin(9600);
}
void loop()
{
            Serial.println("Iniciando a contagem For()");
            for(int x=1; x<=10;x++){
                  Serial.println("Contador:"+String(x));
                   delay(500);
            }
}
```

Exemplo 06: Contando de forma decrescente os números 10 até 1.

```
void setup()
{
      Serial.begin(9600);
}
void loop()
{
            Serial.println("Iniciando a contagem For()");
            for(int x=10; x>=1;x--){
                  Serial.println("Contador:"+String(x));
            delay(500);
            }
}
```

Exemplo 07*:* Fazendo um LED piscar 10 vezes usando o ***while()***, mas sem o LED! A cada acionamento do Led existe um ***delay()*** para ajustar o tempo de visualização com o usuário.

```
int pisca = 1;
void setup()
{
	Serial.begin(9600);
	// ativa sinal para o led
}
void loop()
{
		while(pisca <=10){
				Serial.println("Pisca:"+String(pisca));
				pisca++;
				// liga led
				delay(500);
				// desliga led
				delay(500);
				}
Serial.println("Fim do Pisca, mas ainda estamos em loop.");
}
```

Exemplo 08: O mesmo exemplo usando ***for()***. Observe que a variável ***pisca*** não fui inicializada, pois temos uma variável global já definida.

```
int pisca = 1;
void setup()
{
	Serial.begin(9600);
	// ativa sinal para o led
}
void loop()
{
		for(pisca; pisca <=10; pisca++){
				Serial.println("Pisca:"+String(pisca));
				// liga led
				delay(500);
				// desliga led
				delay(500);
		}
```

```
Serial.println("Fim do Pisca, mas ainda estamos em loop.");
}
```

1.2. Continue

Podemos utilizar a instrução ***continue*** para interromper o fluxo do loop e realizar a próxima interação, podendo ser aplicado nos comandos de repetição (***for, while ou do..while***), realizando a verificação da expressão (condicional).

Exemplo 09: Durante a impressão do loop, uma faixa será ignorada (41 ...59).

```
void setup()
{
      Serial.begin(9600);
}
void loop()
{
        for(int contador=0; contador <=100; contador++){
              if(contador >40 && contador <60){
                    Serial.println("Pulando uma faixa");
                    continue;
              }
              Serial.println(contador);
              delay(5);
        }
}
```

1.3. Instrução Goto

A instrução ***goto*** permite a criação de labels, assim podemos realizar desvios no decorrer do código.

Exemplo 10: Contando de 1..100, usando o goto.

```
int contador = 1;

void setup()
{
                    Serial.begin(9600);
```

```
}
void loop()
{
                    contar:
                                Serial.println(contador);
                                delay(50);
                                contador++;
                                if (contador==100) goto fim;
                                goto contar;
                    fim:
                                Serial.println("Fim");
                                contador = 0;
                                goto contar;
}
```