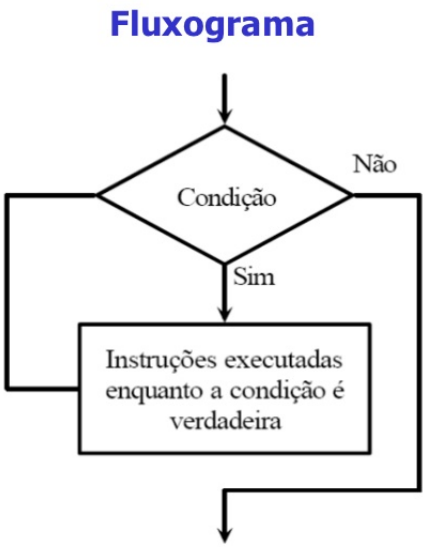
Recapitulando as aulas anteriores

Nas aulas anteriores vimos como fazer a leitura de dados para **int**, **double**, **char** e strings e como fazer as operações com cada um. Também vimos como fazer decisões com as condicionais vistas.

Loop

Temos que nossos códigos por enquanto só têm um número determinado de passos. Uma vez que programamos para determinado valor. No exemplo da calculadora, gostariamos de fazer mais de uma conta e não abrir o programa para cada vez que quiséssemos calcular. Assim, utilizamos o comando **while** que significa enquanto. Um fluxograma com um **loop** é mostrado pela Figura 1.

Figure 1.



O while. A declaração do **while** é mostrado pelo codigo 22-while.c:

```
1   while( /* Condicao de parada */ )
2   {
3     /* Comandos */
4   }
5   return 0;
```

Assim, abaixo(23-cafe.c) é um algoritmo que pergunta se quer café até o usuário digitar sim:

```
1  char resposta;
2  printf("Voce quer cafe? ");
3  scanf("%c", &resposta);
4  getchar();
5  while(resposta != 's')
6  {
7    printf("Voce quer cafe? ");
8    scanf("%c", &resposta);
9    getchar();
10 }
11 printf("Que bom! Seu cafe esta aqui!\n");
```

do while. Às vezes é interessante que uma sequencia de comandos seja feita e então testar a condicional. Assim, o nosso algoritmo 23 faz a mesma coisa que 24-cafe.c

```
char resposta;
do{
  printf("Voce quer cafe? ");
  scanf("%c", &resposta);
  getchar();
}while(resposta != 's');
printf("Que bom! Seu cafe esta aqui!\n");
```

Assim, neste caso foi economizado 4 linhas. Em alguns casos, isso é importante para deixar o código mais legível e em outros é preferível utilizar o **while**.

E se quisermos imprimir todos os números de 1 até 100, ou mesmo somar(exercício)? Bem, neste caso teremos que repetir 100 vezes uma impressão(porque copiar e colar 100 printf's é complicado, e se quiséssemos 5000?). Para fazer isso, precisamos de **contador**. O código 25-imprimir_linhas.c exemplifica o uso de um contador, que utilizaremos como variável.

```
  int contador;
  contador = 1;
  while(contador != 100)
  {
    printf("%d\n", contador);
    contador += 1;
  }
```

for e argumentos. Existe uma outra alternativa como **loop** que é o **for**. Seu uso é exemplificado pelo código 26-for.c, que faz a mesma coisa que o codigo 25.

```
  int contador;
  for(contador = 1; contador != 100; contador += 1)
  {
    printf("%d\n", contador);
  }
```

Dependendo do caso, é melhor utilizar o comando **for**, já em outros é melhor utilizar um **while**. Os argumentos do for é mostrado:

```
  for(/* Atribuicao Valor */; /* Condicao de parada */; /* Alteracao
     Condicao de Parada*/)
  {
    /* Comandos */
  }
```

Que é mesma coisa que:

```
  /* Atribuicao Valor*/
  while(/* Condicao de parada */)
  {
    /* Comandos */
    /* Alteracao Condicao de Parada*/
  }
```