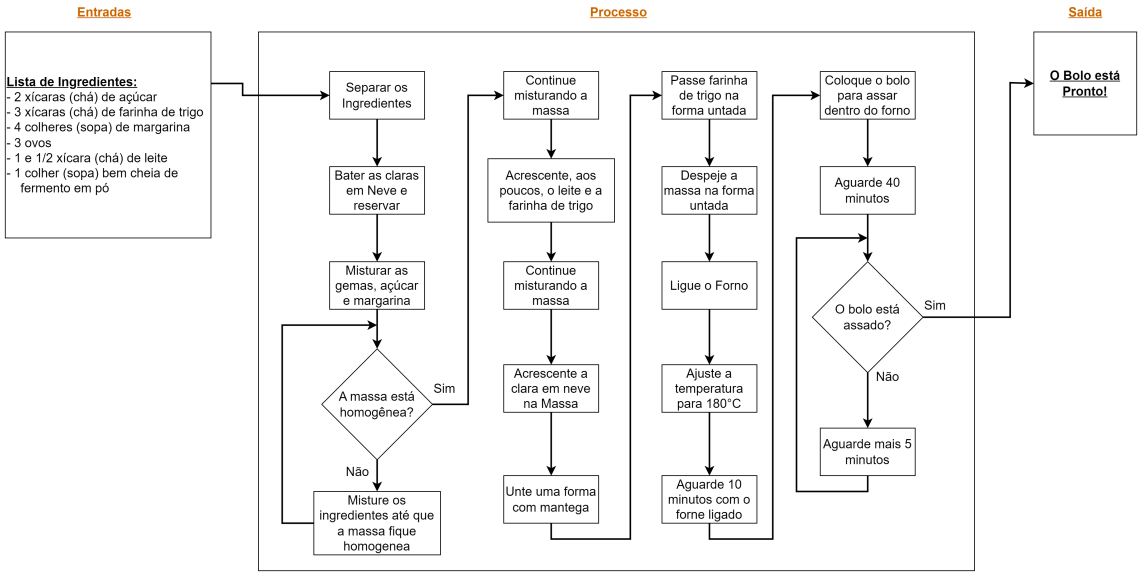
Aula 5 - Transformando um Diagrama de Blocos em Pseudocódigo

Ciclo	Ciclo 01: Lógica de Programação
# Aula	5
Created	@October 19, 2022 2:37 PM
Reviewed	☐
Material PDF	
Finished	☑
Status	

▼ O Algoritmo de Receita de Bolo

Na aula passada criamos um algoritmo para fazermos um bolo utilizando Diagrama de Blocos.



Porém, vimos na aula passada também que o computado não consegue entender esse diagrama e que temos que transcrever esse diagrama em comandos utilizando uma linguagem de programação.

Mas antes de começarmos a utilizar linguagens de programação, como o Python, vamos primeiro aprender como transcrever um Diagrama de Blocos em comandos mais simples, para que faça sentido o método de transformação de uma estrutura em outra! Com a prática e tempo, essa transformação será tão natural que você não precisará mais nem utilizar Diagramas de Blocos!

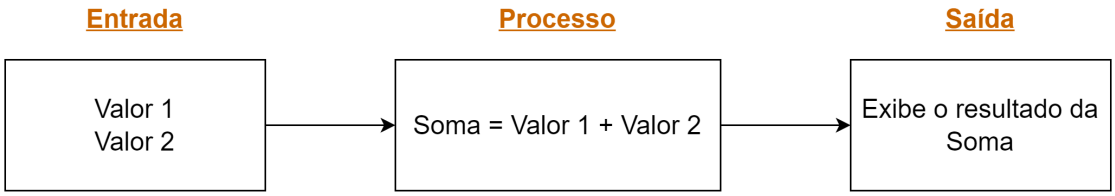
▼ Pseudocódigo

Pseudocódigo é uma maneira de escrevermos comandos, em uma sequencia lógica mas sem utilizar linguagens de programação, mas utilizar comandos que sejam mais fáceis de assimilar. A ideia é “imitar” os comandos que daríamos em linguagens de programação. Para isso temos os seguintes comandos:

- `var`: utilizamos para declarar variáveis que serão utilizadas para armazenar as nossas entradas.
- `leia`: utilizamos para receber dados que um usuário irá inserir no sistema
- `ate_que <condição> efetue`: utilizamos para criar as estruturas que possuem comandos de repetição até que uma determinada condição seja cumprida
- `escreva "Texto"`: utilizamos para exibir para o usuário o valor de uma variável

- `se <condição> senao`: utilizamos para escrever o bloco condicional. Ou seja, utilizamos esse comando para verificar algo no algoritmo
- `inicio`: utilizamos para delimitar o inicio da nossa aplicação ou algoritmo
- `fim`: utilizamos para delimitar o fim da nossa aplicação ou algoritmo
- `←`: utilizamos para colocar o resultado de um cálculo em uma variável

Para exemplificar melhor, vamos transcrever o algoritmo que criamos para somar dois valores, utilizando diagrama de blocos. Primeiro observe os Diagrama de Blocos:



Agora, vamos transcrever cada um dos blocos com as suas respectivas ações. O primeiro bloco, o bloco de entrada, faz referência às entradas, ou variáveis que teremos. No caso, iremos utilizar o comando `var` para declarar essas variáveis:

```
var
  numero1, numero2, soma
```

Uma vez que declaramos o bloco de entrada, que é responsável pelas entradas de informação no algoritmo, vamos começar a escrever o algoritmo em si. Primeiro, delimitados o inicio do nosso algoritmo com o comando `inicio`. Depois, iremos transcrever o processo de somar as duas variáveis e armazenar o valor somando dentro da variável soma utilizando o comando `←`:

```
var
  numero1, numero2, soma
inicio
  soma <- numero1 + numero2
```

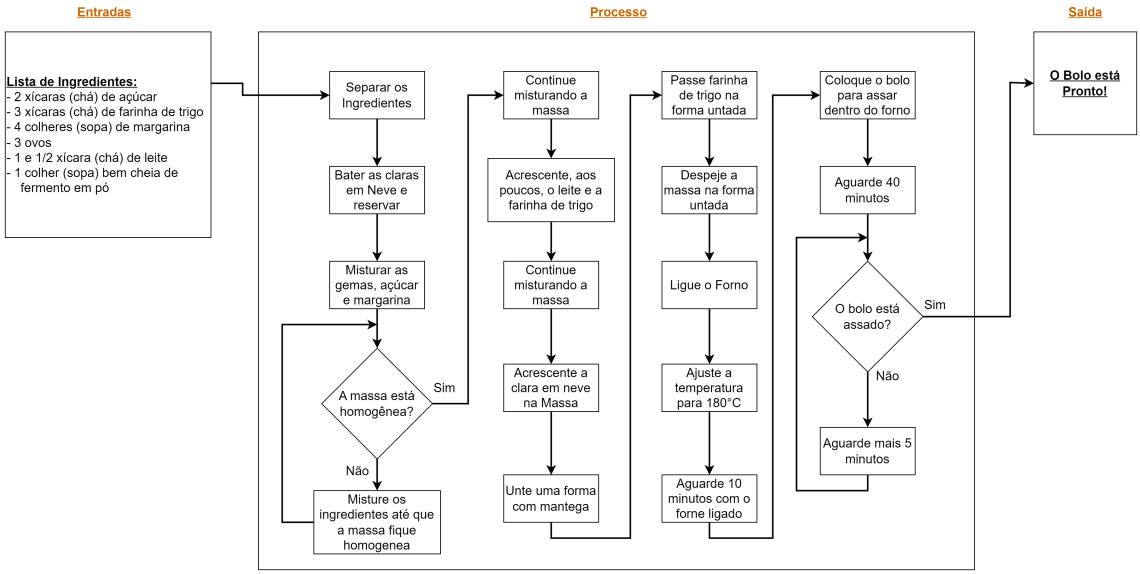
Por fim, vamos exibir o resultado armazenado na variável soma com o comando `escreva` e finalizar o programa, que é o bloco de saída, com o comando `fim`:

```
var
  numero1, numero2, soma
inicio
  soma <- numero1 + numero2
  escreva "O resultado é {soma}"
fim
```

Com isso, temos o algoritmo de soma de dois números descrito utilizando pseudocódigo!

▼ Algoritmo do Bolo no Pseudocódigo

Antes de transcrevermos, vamos visualizar o Diagrama de Blocos:



Feito isso, vamos primeiro pensar nas variáveis de entrada com o comando `var`:

```
var
  acucar
  farinha_trigo
  margarina
  ovos
```

```
  leite
  fermento
```

Com as variáveis definidas, vamos iniciar o nosso algoritmo com o comando `inicio`:

```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
  fermento
inicio

fim
```

Agora vamos começar a transcrever os comandos até o primeiro bloco de de repetição, que utilizamos para misturar a massa até que ela seja homogênea.

```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
  fermento
inicio

  separa os ingredientes
  bater claras em neve e reservar
  misturar as gemas, açucar e margarina

fim
```

Agora vamos utilizar o comando `ate_que` para repetir a ação de misturar a massa, uma vez que ela é um bloco de repetição:

```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
  fermento
inicio

  separa os ingredientes
```

```
  bater claras em neve e reservar
  misturar as gemas, açucar e margarina

  ate_que massa fique homogênea efetue:
    misture os ingredientes até que a massa fique homogenea

fim
```

Feito isso, vamos continuar as instruções até o próximo bloco de repetição de comando:

```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
  fermento
inicio

  separa os ingredientes
  bater claras em neve e reservar
  misturar as gemas, açucar e margarina

  ate_que massa fique homogênea efetue:
    misture os ingredientes até que a massa fique homogenea

  continue misturando a massa
  acrescente aos poucos o leite e a farinha de trigo
  continue misturando a massa
  acrescente a clara em neve na massa
  unte uma forma com manteiga
  passe a farinha de trigo na forma untada
  despeje a massa na forma untada
  ligue o forno
  ajuste a temperatura para 180°C
  aguarde 10 minutos com o forno ligado
  coloque o bolo para assar dentro do forno
  aguarde 40 minutos

fim
```

Com os comandos ajustados, vamos criar o outro bloco de repetição de comandos, que verifica se o bolo está ou não assado:

```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
```

```
  fermento
inicio
  separa os ingredientes
  bater claras em neve e reservar
  misturar as gemas, açucar e margarina

  ate_que massa fique homogênea efetue:
    misture os ingredientes até que a massa fique homogenea

  continue misturando a massa
  acrescente aos poucos o leite e a farinha de trigo
  continue misturando a massa
  acrescente a clara em neve na massa
  unte uma forma com manteiga
  passe a farinha de trigo na forma untada
  despeje a massa na forma untada
  ligue o forno
  ajuste a temperatura para 180°C
  aguarde 10 minutos com o forno ligado
  coloque o bolo para assar dentro do forno
  aguarde 40 minutos

  ate_que bolo esteja assado efetue:
    aguarde 5 minutos

fim
```

Uma vez que o bolo estiver assado, a condição colocada dentro do comando `ate_que` será verdadeira e o bloco de repetição será encerrado, e com isso o bolo estará assado. Por isso, podemos colocar o comando `fim` para delimitar o fim do algoritmo:

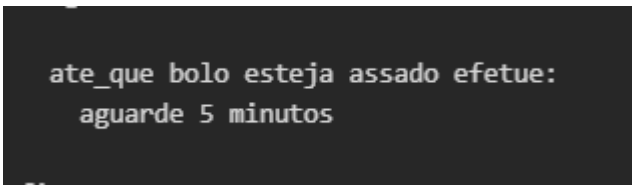
```
var
  acucar
  farinha_trigo
  margarina
  ovos
  leite
  fermento
inicio
  separa os ingredientes
  bater claras em neve e reservar
  misturar as gemas, açucar e margarina

  ate_que massa fique homogênea efetue:
    misture os ingredientes até que a massa fique homogenea

  continue misturando a massa
  acrescente aos poucos o leite e a farinha de trigo
  continue misturando a massa
  acrescente a clara em neve na massa
  unte uma forma com manteiga
  passe a farinha de trigo na forma untada
```

```
  despeje a massa na forma untada
  ligue o forno
  ajuste a temperatura para 180°C
  aguarde 10 minutos com o forno ligado
  coloque o bolo para assar dentro do forno
  aguarde 40 minutos

  ate_que bolo esteja assado efetue:
    aguarde 5 minutos

fim
```

Observe que, quando utilizamos os comandos `inicio` e `fim`, fizemos um recuo nos comandos do algoritmo. Esse recuo é chamado de indentação, e é utilizado em linguagens de programação, como o Python, para delimitar blocos. Ou seja, o bloco principal do algoritmo, que fica entre os comandos `inicio` e `fim`, possui uma indentação para indicar que esses comandos pertencem à esse bloco principal. O bloco de repetição de comandos, possui um indentação em seus comandos internos, tanto o de misturar a massa quanto o de aguardar 5 minutos, para indicar que esses comandos pertencem ao bloco de repetição de comandos, e não ao bloco principal do algoritmo.

Esse conceito é extremamente importante pois ele define como os comandos serão executados pelo computador e quais comandos pertencem a quais blocos.

Por exemplo, o comando `aguarde 5 minutos`, pertence ao bloco `ate_que bolo esteja assado efetue:`, e só será executado caso o bloco `ate_que bolo esteja assado efetue:` for executado.



E na aula que vem, iremos aprender como transcrever esse mesmo Pseudocódigo em código Python!