**Texto base**

**7**

**Estrutura de Repetição**

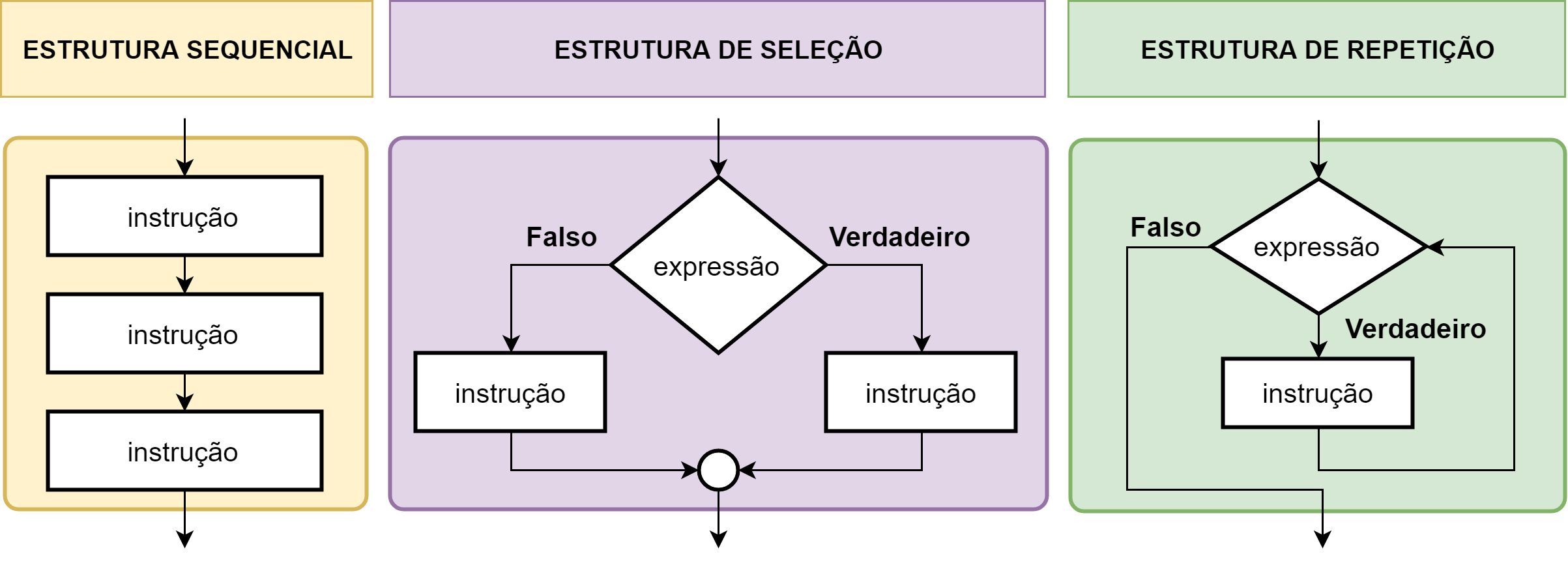
Gilberto Alves Pereira

***Resumo***

*Neste texto é discutido o conceito de Estrutura de Repetição. Basicamente, a Estrutura de Repetição (também chamada loop ou laço) repete comandos. Podemos classificá-las em dois tipos: Indefinida e Definida. Na Estrutura de Repetição Indefinida, não sabemos o número de vezes que teremos que repeti-los. Na Estrutura de Repetição Definida, tem o objetivo de repetir comandos um determinado número fixo de vezes, ou seja, sabemos o número de vezes que teremos que repeti-los. Abordamos também o uso de laços usados na validação de dados de entrada, laços que utilizam variáveis chamadas Flags e, por fim, as variáveis contadoras usadas em contagens de eventos. Em seguida, abordamos laços que utilizam variáveis chamadas acumuladoras usadas para somar valores e por fim loops infinitos, aqueles que nunca terminam. Ao final é mostrado um exemplo usando fluxograma com sua conversão para Python.*

# 1.1. Estruturas

Já vimos duas formas de controlar o fluxo de um programa: estrutura de controle sequencial e estrutura de controle de seleção ou condicional.

**Figura 7.1. Fluxograma de estruturas usadas em lógica de programação. Fonte Autor.**

Vamos iniciar as **estruturas de repetição**, também conhecidas como **iterativas** ou de **laço** (loop) que, junto com as demais estruturas vistas até agora, propiciarão a solução de uma gama muito maior de problemas.

As estruturas de repetição permitem que uma ação seja executada **várias vezes** sem que tenhamos que executar novamente o programa.

Uma estrutura de **controle iterativo**, estrutura de repetição, laço de repetiçãoou simplesmente **laço** é uma estrutura de controle de fluxo formada por um conjunto de instruções que são executadas um **determinado número de** **vezes** ou enquanto uma determinada **condição** for verdadeira.

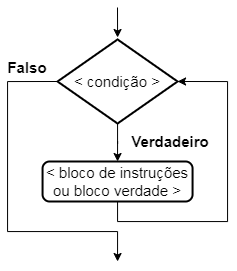
# 1.2. Estruturas de Repetição

Uma estrutura de repetição, também chamada de loop ou laço, tem o objetivo de repetir comandos. Seu funcionamento depende de uma condição ou expressão lógica. Os comandos são repetidos enquanto a condição for verdadeira. Quando a condição for falsa a repetição termina e o programa segue. Figura 7.2.

Esta estrutura permite executar diversas vezes um bloco de instruções, sempre verificando **antes** se a **<condição>** é verdadeira.

A condição é uma expressão booleana, semelhante às expressões que usamos nas estruturas de seleção.

A **<condição>** é primeiro avaliada. Enquanto a **<condição>** for verdadeira, o **<bloco verdade>** é executado. Quando a **<condição>** for falsa, a iteração termina e a execução continua com a instrução após o laço.

**Figura 7.2. Fluxograma de uma estrutura de repetição. Fonte Autor.**

Com relação a forma como o laço é montado temos duas possibilidades:

1. Estrutura de Repetição Indefinida: Quando não sabemos quantas vezes serão repetidos os comandos.
2. Estrutura de Repetição Definida: Quando sabemos quantas vezes serão repetidos os comandos.

# 1.3. Estrutura de Repetição Indefinida

A Estrutura de Repetição Indefinida ocorre quando não sabemos quantas vezes serão repetidos os comandos. No dia-a-dia temos inúmeros exemplos desse tipo de necessidade. Quando vamos a um caixa eletrônico, por exemplo, a princípio não sabemos exatamente quantas operações vamos fazer. Chegamos no caixa eletrônico, fazemos a nossa operação e ao final nos é perguntado se queremos fazer uma nova operação. Se respondermos que sim, o processo recomeça e efetuamos a nova operação e assim sucessivamente até que resolvemos parar e respondemos não à pergunta sobre a nova operação e o processo finaliza.

Vejamos o exemplo da Figura 7.3 abaixo. Nesse exemplo utilizamos uma variável *opc* que nos ajuda a controlar o laço. Inicialmente foi armazenado um valor nessa variável “s” de maneira que o teste resulta em verdadeiro. Assim, o bloco verdade é executado (a exibição da palavra “Hello”) e ao final dele o usuário digita um novo valor para essa variável. Caso ele digite “s”, ao retornar ao teste novamente ele resultará em Verdadeiro o que fará executar o bloco verdade novamente. Caso seja digitado qualquer valor diferente de “s” o teste resultará em falso e o laço se encerra.

Perceba que nesse exemplo não sabemos quantas vezes o laço será executado. Isso vai depender totalmente do que o usuário digitar na entrada dentro do loop. Devido a esse fato essa montagem é chamada de Laço Indefinido.

**Figura 7.3. Fluxograma de uma estrutura de repetição indefinida. Fonte Autor.**

# 1.4. Estrutura de Repetição Definida

A Estrutura de Repetição Definida ocorre quando sabemos quantas vezes serão repetidos os comandos. Portanto usamos essa construção de laço sempre que sabemos de antemão o número de repetições. Na exibição da tabuada do 5 sabemos que temos que repetir 5 x 1 = 5 e assim sucessivamente por 10 vezes. Quando precisamos coletar as notas dos alunos de uma sala de 60 alunos, também sabemos a quantidade de vezes que precisamos repetir.

Exemplo: Considere o programa que exibe os números inteiros de 1 a 3.

|  | **Teste de Mesa - Teste de Execução**   | **x** | **x <= 3** | **Tela** | | --- | --- | --- | | 1 | 1 <= 3 True | 1 | | 2 | 2 <= 3 True | 2 | | 3 | 3 <= 3 True | 3 | | 4 | 4 <= 3 False |  |   A execução para no momento que o teste da condição avalia para **Falso** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

**Figura 7.4. Fluxograma de uma estrutura de repetição definida. Fonte Autor.**

Na figura 7.4, a variável **x** é inicializada com o valor igual a **1** e a condição é avaliada pela primeira vez. Como a condição **x <= 3** é verdadeira, a instrução dentro do laço é executada, exibindo o valor de **x** e atualizando o valor de **x** para **2**. O controle de execução retorna para o topo do laço na avaliação da condição novamente. Como **2** é menor ou igual a **3**, o bloco dentro do laço é executado, exibindo o valor de **x** e atualizando **x** para **3**. Retoma-se ao topo da estrutura com o teste da condição, como é verdade, o bloco de instruções é executado novamente pela terceira vez. Nesse ponto, a variável **x** assumiu o valor 4 e a condição **x <= 3** resulta em falso (False), terminando assim a repetição do bloco.

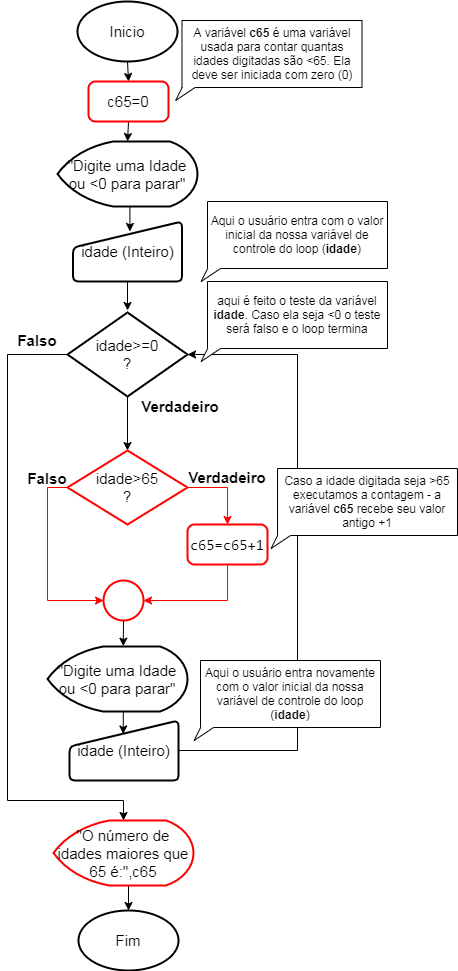
É importante observar que, no caso acima descrito, as instruções do laço são executadas 3 vezes, enquanto a condição é avaliada 4 vezes. Isto acontece pois toda vez que a condição é avaliada, o bloco de instruções ou bloco verdade é executado e retorna-se à verificação da condição, até que a condição seja avaliada para Falso.

# 1.5. Variável Contadora

**Variável contadora** é uma variável utilizada para contagem do número de vezes que ocorre um determinado evento.

Recebe um valor inicial (geralmente 0) e é incrementada em algum ponto do algoritmo de um valor constante (geralmente 1). Incrementar uma variável é o mesmo que somar um valor constante a essa variável.

Com o objetivo de entendermos melhor essa variável contadora, vamos considerar a seguinte questão: Desenhar um fluxograma que recebe diversas idades até que seja digitada uma idade negativa. Contar e exibir a quantidade de idades maiores que 65 foram digitadas.

**Figura 7.5. Fluxograma exemplo de uso de variável contadora. Fonte Autor.**

Nesse exemplo a variável que controla nosso laço é a idade. Veja que nesse caso, ao invés de colocarmos um valor inicial para a variável de maneira a entrar no laço, executamos o comando de entrada. Assim, caso o usuário digite um valor de idade negativo o teste lógico resulta em Falso e nem chegamos a entrar no laço. Caso a idade digitada seja maior ou igual a zero, o fluxo segue no caminho do verdadeiro entrando no laço. Dentro do laço é verificada a idade novamente. Agora para ver se consideramos essa idade na contagem das idades maiores que 65. Caso o teste seja verdadeiro (idade>65) efetuamos a contagem (c65=c65+1). Esse comando incrementa de 1 a variável c65 efetuando a contagem. O laço finaliza coletando mais uma idade do usuário e voltando ao teste (idade>=0).

Quando o usuário digitar uma idade negativa o teste do laço resulta em Falso, saímos do laço e executamos o comando seguinte (após o laço) de exibição da quantidade de idades maiores que 65 (variável c65).

Podemos efetuar várias contagens dentro de um laço, basta que para isso sejam inseridas estruturas de seleção com os testes adequados e que seja criada uma variável contadora para cada evento que desejarmos contar.

# 1.6. Validação de Dados de Entrada

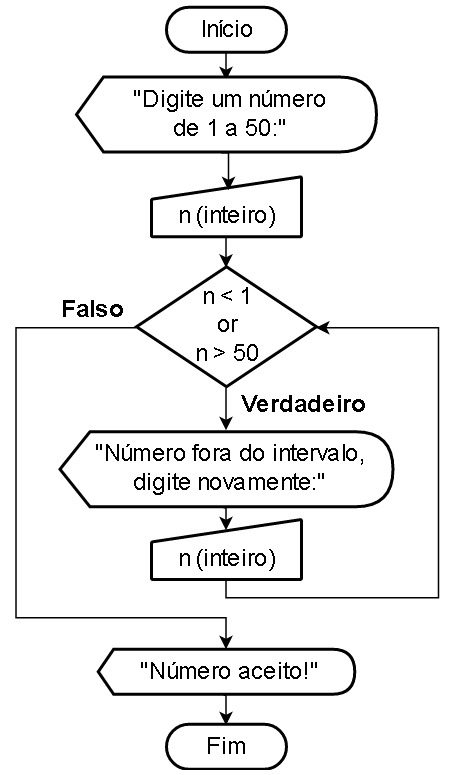
Consiste em verificar se o valor informado pelo usuário está correto ou não. Anteriormente usamos a estrutura condicional para checar informações de entrada e, em caso de erro, encerramos o programa.

Usando uma estrutura de repetição podemos permitir que o usuário digite o dado enquanto o esteja informando de forma incorreta.

**Exemplo 1**: Se o usuário precisar digitar um número no intervalo 1 ≥ n ≤ 50, podemos usar a estrutura de repetição que recebe o número e, enquanto este número estiver fora do intervalo permitido, pede novamente que o número seja digitado. Confira na Tabela 7.6.

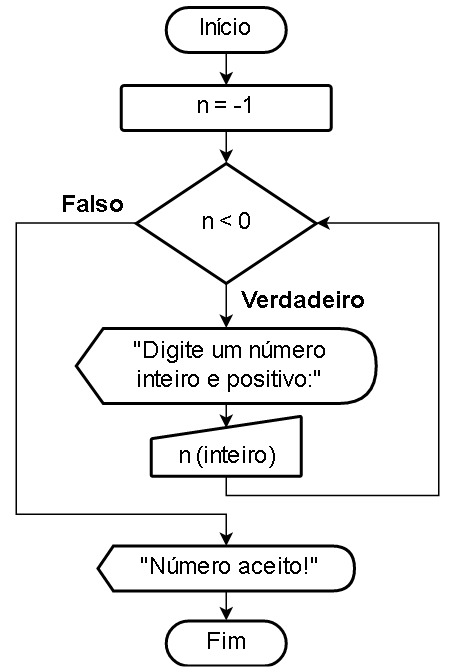
**Tabela 7.6. Teste de Mesa - Fluxograma de validação dos dados de entrada.**

| **Tela** | **n** | **n < 1 or n > 50** |
| --- | --- | --- |
| Digite um número de 1 a 50: | 0 | True or False => True |
| Digite um número de 1 a 50: | 51 | False or True => True |
| Digite um número de 1 a 50: | 10 | False or False => False |
| Número aceito! |  |  |

**Figura 7.7. Exemplo 1 - Fluxograma exemplo de validação de entrada de dados.**

**Fonte Autor.**

**Exemplo 2** : Desenhe um fluxograma que leia e valide a entrada de um número inteiro e positivo. Usando uma estrutura de repetição podemos permitir que o usuário digite o dado enquanto estiver informando de forma incorreta.



**Figura 7.8. Exemplo 2 - Fluxograma exemplo de validação de entrada de dados.**

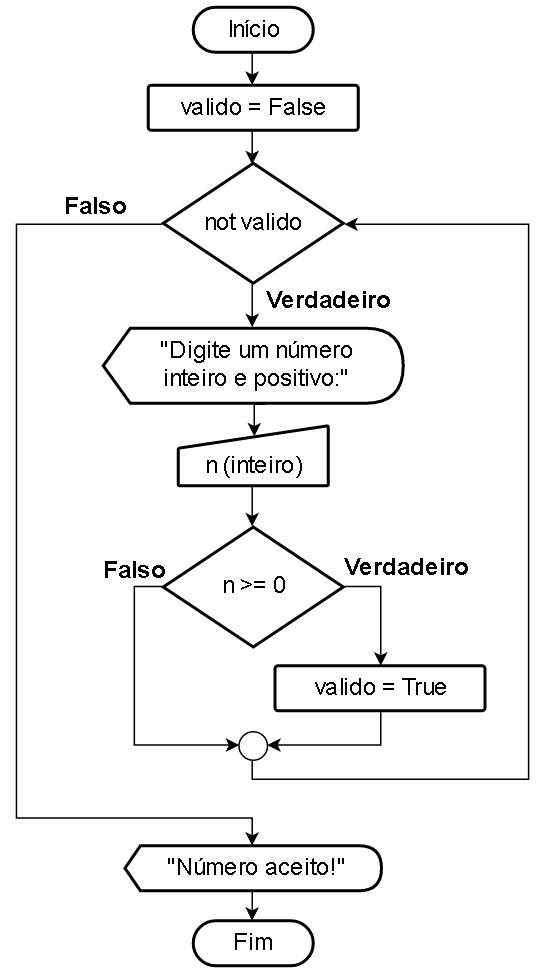
**Fonte Autor.**

Perceba que nos dois exemplos acima a única forma de finalização do loop ocorre se o usuário digitar um valor válido. O teste é feito de maneira a caso o valor digitado seja incorreto o teste resulta em verdadeiro e assim o fluxo permanece no laço.

# 1.7. Interrupção do laço. Flag booleana

O controle do laço pode ser feito por meio do laço infinito de uma única variável booleana, chamada de flag booleana. Dentro do laço verificamos se uma ou mais condições são atendidas através de uma estrutura de seleção e alteramos o valor da flag para que o laço seja encerrado.

Reescrevendo o exemplo anterior, da entrada de um número inteiro e positivo, com o uso de uma flag booleana, chegamos ao seguinte fluxograma:



**Figura 7.9. Fluxograma exemplo de utilização de variável Flag booleana.**

**Fonte Autor.**

É importante notar que o nome da flag e o valor inicial que lhe é atribuído devem ser coerentes com a situação do problema modelado, isto é, iniciar uma flag cujo nome seja VÁLIDO com um valor True não faz sentido, pois teríamos que alterá-la para False justamente no momento em que o número fosse validado. Isso significaria que a flag guardaria o valor True enquanto o número ainda não foi validado (ou seja, é INVÁLIDO) e guardaria o valor False quando o número finalmente fosse VÁLIDO.

# 1.8. Variável Acumuladora

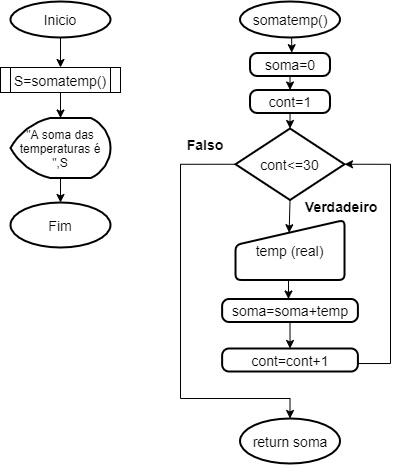
**Variável acumuladora** é uma variável utilizada para controlar a contagem, que recebe um valor inicial (geralmente 0) e é incrementada em algum ponto do algoritmo de um valor variável. Exemplo: Calcular a soma de 5 números digitados pelo usuário.

|  | **Teste de Mesa**   | **soma** | **cont** | **cont <= 5** | **num** | | --- | --- | --- | --- | | 0 | 1 | True | 3 | | 3 | 2 | True | 4 | | 7 | 3 | True | 2 | | 9 | 4 | True | 5 | | 14 | 5 | True | 8 | | 22 | 6 | False  (fim do laço) |  |   **Tela:**  **A soma dos números é: 22** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

**Figura 7.10. Fluxograma mostrando o uso de uma variável acumuladora. Fonte Autor.**

Perceba que tanto para as variáveis acumuladoras quanto para as contadoras precisamos iniciá-las com zero. No exemplo acima a variável **cont** é a variável de controle do nosso laço que executa 5 vezes. Para cada execução do laço o usuário digita um valor (**num**) que é acumulado na nossa variável acumuladora **soma.** Para acumularmos valores, executamos o comando ***variável = variável + valor a ser acumulado***. No nosso exemplo soma=soma + *num*. Veja que *num* é a variável que recebe os diversos valores digitados pelo usuário. Observe no teste de mesa que a variável soma acumula a cada execução do comando **soma = soma + num** o valor da variável **num**. Ao final, a variável soma possui a soma de todos os valores digitados pelo usuário **22**.

**Exemplo 3** - Desenhar um fluxograma que recebe 30 temperaturas de uma cidade do mês de setembro. Exibir a soma dessas temperaturas. Usar uma função para obter a soma.



**Figura 7.11. Fluxograma do exemplo 3 - soma das temperaturas. Fonte Autor.**

Para esse exemplo vamos usar uma função chamada somatemp. O objetivo da nossa função será coletar as temperaturas e calcular a soma. Perceba que a exibição do resultado não está na função. Para usarmos essa função vamos criar um fluxograma que invoca (ou chama) a função, obtém o resultado da função através da variável S e após isso exibe a mensagem com o resultado, Observemos que na função No exemplo acima a variável **cont** é a variável de controle do nosso laço que executa 30 vezes (uma para cada dia do mês) pois o valor dessa variável inicia com 1 no teste, temos cont <= 30 (ou seja quando atingir 31 para o laço), e a cada execução essa variável é incrementada de 1 (cont = cont + 1) - o chamado passo. Portanto esse loop vai ser executado 30 vezes. Colocamos dentro do laço uma entrada de dados na variável **temp** (temperatura). Finalizando criamos a variável soma. Essa é nossa variável acumuladora das temperaturas. Para cada execução do laço o usuário digita um valor (**temp**) que é acumulado na nossa variável acumuladora **soma.** Usando o comando **soma = soma + temp**. Veja que **temp** é a variável que recebe as 30 temperaturas do mês de setembro (uma em cada execução do loop), valores esses digitados pelo usuário. Ao final, a variável soma possui a soma de todas as temperaturas digitadas pelo usuário.Vamos usar o conteúdo dessa variável como retorno da função comando ***return soma***.

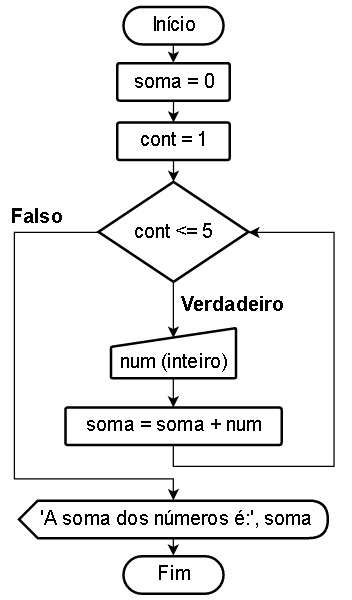
O fluxograma que invoca a nossa função (lado esquerdo da Figura 7.11) possui dois comandos. O primeiro comando invoca a função e obtém o resultado da função (devido ao retorno da função). Como a nossa função retorna da soma das temperaturas, a variável S do nosso fluxograma receberá esse valor. Por fim, é usado o comando de exibição para mostrar a mensagem com o valor da soma das temperaturas.

# 1.9. Laço Infinito

**Laço ou loop infinito** é uma estrutura de controle iterativo que nunca termina (ou eventualmente termina com um erro de sistema).

**Exemplo**: Calcular e exibir a soma de 5 números digitados pelo usuário.

Observe que no fluxograma abaixo (Figura 7.12) o teste lógico sempre resultará em Verdadeiro porque a variável de controle **cont** recebe inicialmente o valor 1 e esse valor não é alterado dentro do laço (não existe o comando do passo **cont = cont + 1**). Dessa forma o teste que será feito **cont <= 5** será sempre **1 <= 5**:Verdadeiro (o valor da variável cont será sempre 1). Sendo assim esse laço nunca termina, resultando em um loop infinito.



**Figura 7.12. Fluxograma de um loop infinito. Fonte Autor.**

# 1.10. Conversão de Fluxograma para Python

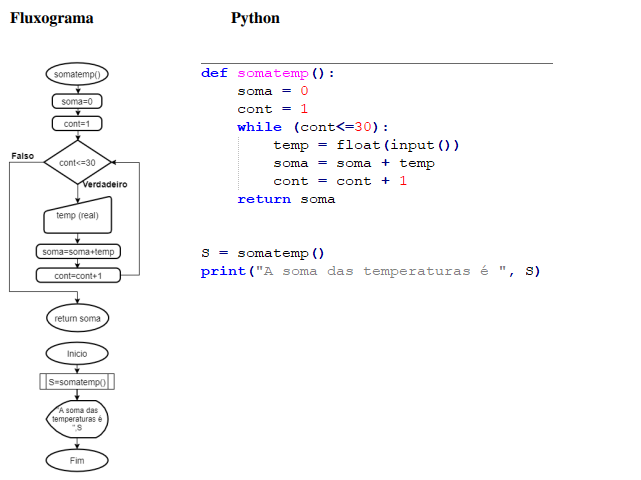
Abaixo podemos ver dois exemplos de fluxograma e sua conversão para python.

Exemplo 1: Desenhar um fluxograma que recebe diversas idades até que seja digitada uma idade negativa. Contar e exibir a quantidade de idades maiores que 65 foram digitadas.

| **Fluxograma** | **Python** |
| --- | --- |
|  | |

**Figura 7.13. Esquema do fluxograma e seu respectivo código em Python referentes ao exemplo do uso da variável contadora. Fonte Autor.**

Exemplo 2: Desenhar um fluxograma que recebe 30 temperaturas de uma cidade do mês de setembro. Exibir a soma dessas temperaturas. Usar uma função para obter a soma.

**Figura 7.14. Esquema do fluxograma e seu respectivo código em Python referentes ao exemplo da soma das temperaturas usando função. Fonte Autor.**

# 1.11. Referências

DIERBACH, C. “Introduction to Computer Science Using Python: A Computational Problem-Solving Focus”1st Edition, New York: Wiley, 2012.