www.estudar.com.vc

MC102 Algoritmos e Programação de Computadores

Resumo Teórico P1

1. Variáveis

Guardam valores.

Podem ser somadas, multiplicadas, etc.

Tipos de Variáveis

$int, float, comples, bool, str$

Conversão: $tipo(valor)$

$$x = 5$$
$$float(x) \rightarrow 5.0$$
$$str(x) \rightarrow "5"$$

2. Operadores

Realizam operações em variáveis e valores:

Tipos

Soma$(+)$	$2 + 4 = 6$
Subtração$(-)$	$7 - 5 = 2$
Multiplicação$(*)$	$5 * 2 = 10$
Divisão real$(/)$	$12 / 10 = 1{,}2$
Divisão inteira$(//)$	$12 // 10 = 1$
Módulo(resto da divisão inteira)$(\%)$	$12 \% 10 = 2$
Exponenciação$(**)$	$2 ** 3 = 8$

Abreviação

$\mathbf{x} = x + 5$ equivale a $x += 5$

Ordem de execução e parênteses funcionam como na matemática.

3. Impressão

$$print()$$

Impressão de Strings

$$print("Oi") \rightarrow Oi$$

Impressão de Variáveis

$$x = 5$$
$$print(x) \rightarrow 5$$

Impressão de Strings e Variáveis

Separa com vírgula.

$$x = 10$$
$$print("O\ número\ é", x) \rightarrow O\ número\ é\ 10$$

Formatos de Impressão

Fica no meio da string, e no final do print, você coloca o que vai substituir no lugar.
$\%d$ para inteiros, $\%f$ para float, $\%c$ para caracteres.

$$x = 7$$
$$print("O\ número\ é\ \%d"\ \%(x)) \rightarrow O\ número\ é\ 7$$

Argumentos Opcionais do $print()$

end: define o caractere a ser adicionado no final da string.
sep: define o caractere que une as coisas que estão separadas por vírgula.

$$print\ ("Oi", "Tchau") \rightarrow Oi\ Tchau$$
$$print("Oi", "Tchau", sep = '') \rightarrow OiTchau$$

4. Entrada de Dados

$$input("Mensagem")$$

Dentro dos parênteses, vai a mensagem que vai ser impressa para o usuário. O programa então para, e espera o usuário digitar algo e apertar *Enter*.

Recebe como String.
Pode ser preciso converter.
Pode ser convertido direto.

$$x = int(input("Digite\ um\ número: "))$$

5. Variáveis Booleanas

Podem assumir apenas 2 valores: $False$ ou $True$.

Comparadores

Analisam 2 valores ou variáveis e determinam a expressão como $True$ ou $False$.

Tipos

Maior: $>$
Menor: $<$
Maior ou igual: $>=$
Menor ou igual: $<=$
Igual: $==$
Diferente: $!=$

Operadores Lógicos

Recebem 2 valores booleanos e retornam o valor da expressão.

and: A expressão vai ser verdadeira se ambas as entradas forem verdadeiras.

or: A expressão vai ser verdadeira se qualquer uma das entradas for verdadeira.

not: Inverte o valor da variável booleana.

É possível unir vários operadores, variáveis, comparações e valores booleanos em uma expressão. A ordem de execução é (do primeiro para o último):

$$comparações \rightarrow not \rightarrow and \rightarrow or$$

6. Execução Condicional

Permite ao programa tomar diferentes caminhos baseado nos dados disponíveis.
if: Executa um código indentado apenas se uma certa expressão booleana for verdadeira.

if(expressão booleana): Código que vai ser executado caso a expressão for verdadeira.

else: Executa um código indentado apenas se a expressão booleana de um "if" anterior for considerada falsa.

elif: Executa um código indentado apenas se a expressão booleana de um "if" anterior for considerada falsa, e a expressão do “elif” for verdadeira.

elif(outra expressão booleana): Código que vai ser executado caso a expressão do "if" for falsa.

else: Código que vai ser executado caso ambos falhem.

Encadeamento de Condicionais

De forma geral, é possível dizer que todo trecho de execução consiste de:

Um "if";
Um número maior ou igual a 0 de "$elifs$";
No máximo um "$else$"(opcional).

7. Loops

São usados quando o programa deve repetir linhas de códigos várias vezes.

While

Enquanto uma expressão for verdadeira, continua repetindo as linhas.

Aceita mais de uma condição de parada.

$$i = 0$$
$$while\ i < 10:$$
$$print(i)$$
$$i += 1\ \#sempre\ lembrar\ de\ incrementar\ o\ contador$$

For

Usado para realizar um número fixo de iterações.

2 formas de utilizar:

a. Percorrer uma lista diretamente.

$$for\ elemento\ in\ lista:$$
$$print(elemento)$$

b. Usar $range()$ para fazer um certo número de operações.

$$for\ i\ in\ range(10):$$
$$print(i)$$

Range

Pode receber 1, 2 ou 3 valores.

$range(x)$ → Vai 0 até $x - 1$

$range(x, y)$ → Vai de x até $y - 1$

$range(x, y, z)$ → Vai de x até $y - 1$, com "passo" z (de z em z)

Encadeamento de Laços

É possível fazer 2 laços, um dentro do outro.

Devem ter contadores diferentes.

São úteis para trabalhar com todas permutações de variáveis.

$Break$ → Sai do loop mais interno.

8. Listas

Podem ter tipos diferentes de variáveis.

São acessadas por índices.

Ficam entre colchetes [].

$$minha\_lista = [1, 2, 3, 4, 5]$$

Indexação Base-0

Índices vão de 0 até comprimento da lista $- 1$.

$$minha\_lista[0] \rightarrow 1$$
$$minha\_lista[4] \rightarrow 5$$

Alterar Elementos

$$minha\_lista[2] = "Três"$$

Índices Negativos

Começam a contar do final.

$$minha\_lista[-1] \rightarrow 5$$
$$minha\_lista[-2] \rightarrow 4$$

Métodos de Listas

***append*()**: Adiciona um elemento "x" no final da lista L.

$$L.append(x)$$

***pop*()**: Recebe um índice, remove o elemento com esse índice da lista e retorna o valor removido.

$$removido = L.pop(i)$$

***remove*()**: Remove um valor, e remove o primeiro elemento que tiver aquele valor da lista.

$$L.remove("Valor\ que\ vai\ ser\ removido")$$

***len*()**: Retorna o comprimento da lista

$$comprimento = len(L)$$

Operações com Listas

Concatenação

$$[1, 2, 3] + [4, 5, 6] \rightarrow [1, 2, 3, 4, 5, 6]$$

Multiplicação

$$3 * [1, 2] \rightarrow [1, 2, 1, 2, 1, 2]$$

Fatia

$$fatia = L[a: b]$$

Copia um trecho da lista que começa em "a" e para em "b" – 1.

Matrizes

Listas de listas.

São percorridas com 2 loops e acessadas com 2 índices.

$M[i][j]$ é o elemento da linha "i", coluna "j" da matriz "M".