CENTRO UNIVERSITÁRIO UNIMETROCAMP | WYDEN

Oficina de Introdução de Programação usando Linguagem Python

Prof. Ms. José CarlosPerini

Apresentação do Professor

- Professor:
  - José Carlos **Perini**
  - E-mail: jose.perini@unimetrocamp.edu.br
  - Blog: http://profperini.com
  - Formação em Análise de Sistemas – Univ. S. Francisco
  - Pós em Administração – Universidade S. Francisco
  - Mestrado em Ciência da Computação – UNIMEP
  - Oracle Certified Professional, Java SE 6 Programmer
  - Professor desde 1996
  - Atualmente:
    - Metrocamp - desde 2003
      - Algoritmos Computacionais
      - Estruturas de Dados
      - Programação Orientada a Objetos
      - Introdução à Informática

Contatos do professor

- José Carlos **Perini**
- E-mail**:** jose.perini@unimetrocamp.edu.br
- Blog: http://profperini.com
- E-mail pessoal: profjoseperini@gmail.com
- Canal do youtube: www.youtube.com/profperini
- Site: http://about.me/profperini

Linguagem de Programação Python

- *Linguagem Python*. Uma linguagem multiparadigma, interpretada. É simples de ser codificada.

- Download do Python:

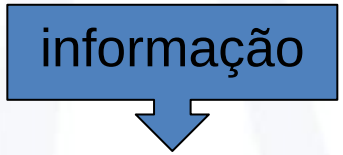
https://www.python.org/downloads/

- Vamos utilizar, para desenvolver os programas,

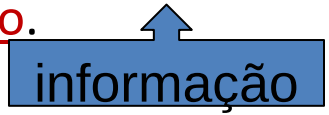
o IDLE (ambiente de desenvolvimento da própria linguagem).

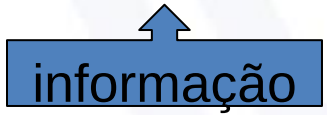
ITENS FUNDAMENTAIS PARA A REPRESENTAÇÃO DE PROGRAMAS

Itens Fundamentais



- Desenvolver um algoritmo para calcular o salário bruto de um funcionário horista, sabendo a quantidade de horas trabalhadas e o valor da sua hora trabalho.





- Pergunta: quais são as informações trabalhadas nesse problema?
- Pergunta: como estas informações são armazenadas para serem manipuladas pelos programas?

Algoritmo

- *Pseudocódigo.* O pseudocódigo é a forma de descrever as ações para a resolução de um problema proposto por meio de *regras preestabelecidas*.
- *Linguagem de Programação*. Um algoritmo pode ser representado por qualquer linguagem de programação.
- *Linguagem Python*. Uma linguagem multiparadigma, interpretada. É simples de ser codificada.

Exercício

- Considerando que queremos resolver o seguinte problema: calcular a média aritmética das 2 notas de um aluno, e imprimir:
  - Olá [nome do aluno], você foi [aprovado|reprovado], com média X

- Quais são as informações que estamos trabalhando neste problema?

Exercício - solução

- Considerando que queremos resolver o seguinte problema: calcular a média aritmética das 2 notas de um aluno, e imprimir:
  - Olá [nome do aluno], você foi [aprovado|reprovado], com média X

- Quais são as informações que estamos trabalhando neste problema?

- Nome do Aluno
- Nota 1
- Nota 2
- Média
- Aprovação

Tipos de Dados

- *tipo inteiro* caracteriza qualquer dado numérico que pertença ao conjunto dos números inteiros
- *tipo real* caracteriza qualquer dado numérico que pertença ao conjunto dos números reais
- *tipo caracter* caracteriza qualquer dado que pertença a um conjunto de caracteres *alfanuméricos*
- *tipo lógico* caracteriza qualquer dado que possa assumir somente uma de duas situações: verdadeiro ou falso

Exercício

- Quais os tipos dos dados abaixo?

- Nome do Aluno
- Nota 1
- Nota 2
- Média
- Aprovação

Exercício - Solução

- Quais os tipos dos dados abaixo?

- Nome do Aluno *Tipo caracter*
- Nota 1 *Tipo real*
- Nota 2 *Tipo real*
- Média *Tipo real*
- Aprovação *Tipo caracter*

Tipos de dados em pseudocódigo

- para o tipo inteiro usaremos *numérico_inteiro*
- para o tipo real usaremos *numérico_real*
- para o tipo caracter usaremos *alfanumérico*
- para o tipo lógico usaremos *lógico*

Tipos de dados em pseudocódigo

- Deste modo, para as informações do nosso exemplo, teremos:

• Nome do Aluno	*Alfanumérico ou caracter*
• Nota 1	*real*
• Nota 2	*real*
• Média	*real*
• Aprovação	*Alfanumérico ou caracter*

Exercício

- Quais os tipos dos dados para as informações abaixo?

- Idade de uma pessoa
- Altura de uma pessoa
- Nome de uma pessoa
- Estado civil de uma pessoa

- Código de um produto
- Descrição de um produto
- Preço de um produto
- Quantidade do produto no estoque

Exercício - Solução

- Quais os tipos dos dados para as informações abaixo?

- Idade de uma pessoa — *Tipo inteiro*
- Altura de uma pessoa — *Tipo real*
- Nome de uma pessoa — *Tipo caracter*
- Estado civil de uma pessoa — *Tipo caracter*

- Código de um produto — *Tipo caracter*
- Descrição de um produto — *Tipo caracter*
- Preço de um produto — *Tipo real*
- Quantidade do produto no estoque — *Tipo inteiro*

Variáveis

- Uma variável é a representação simbólica dos dados envolvidos na solução de problemas computacionais.
- Cada variável corresponde a uma posição de memória do computador, cujo conteúdo pode variar ao longo do tempo de execução do programa.
- Embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante.

Variáveis em Python

- Na linguagem Python, ao criarmos variáveis, não precisamos declarar o seu tipo.
- Ao inicializarmos, a variável assume o tipo de acordo com o valor.
- Se a seguir, atribuímos à variável um valor de tipo diferente, seu tipo é alterado automaticamente.

Exemplo de variável em Python

```
numero = 50
nome = "José Carlos Perini"
valor = 15.90
letra = 'X'
print(numero)
print(nome)
print(valor)
print(letra)
```

Exercício de exemplo

- Fazer um programa que inicialize as seguintes variáveis e depois apresente:
- Seu nome
- Sua idade
- Sua altura
- Seu peso
- Seu endereço

Exercício Exemplo

```
# Exercicio exemplo
nome = 'José carlos Perini'
idade = 58
altura = 1.65
peso = 54
endereco = 'Rua Artur Lugli, 68'
print('Nome: ' + nome)
print('Idade: ' + str(idade))
print('Altura: ' + str(altura))
print('Peso: ' + str(peso))
print('Endereço: ' + endereco)
```

Entrada e saída de dados

- Para saída de dados, em Python:

```
print("Nome: " + nome)
print("Idade: " + str(idade) + " anos")
print("Altura: " + str(altura))
print("Peso: " + str(peso))
print("Endereço: " + endereco)
```

Entrada e saída de dados

- Formatando a saída em String:

```
print("Nome: %s "   % nome)
print("Idade: %d anos" % idade)
print("Altura: %.2f" % altura )
print("Peso: %.2f" % peso)
print("Endereço: %s" % endereco)
```

Tipos de dados

Tipo	Formato
int	%d ou %i
float	%f
string	%s

Entrada e saída de dados

- Para entrada de dados, em Python:

```
# Exemplo 2
nome = input("Digite o seu nome: ")
idade = input("Digite a sua idade: ")
idade = int(idade)
altura = input("Digite a sua altura: ")
altura = float(altura)
peso = input("Digite o seu peso: ")
peso = float(peso)
endereco =input("Digite o seu endereço: ")
print("Nome: %s "  % nome)
print("Idade: %d anos" % idade)
print("Altura: %.2f" % altura )
print("Peso: %.2f" % peso)
print("Endereço: %s" % endereco)
```

Entrada e saída de dados

- Para entrada de dados, em Python:
- Valores numéricos devem ser convertidos.

```
idade = input("Digite a sua idade: ")
idade = int(idade)
altura = input("Digite a sua altura: ")
altura = float(altura)
peso = input("Digite o seu peso: ")
peso = float(peso)
```

Exercício

- Refazer o exercício 1 para os valores serem digitados.

Correção do exercício

```
# Exemplo 2
nome = input("Digite o seu nome: ")
idade = input("Digite a sua idade: ")
idade = int(idade)
altura = input("Digite a sua altura: ")
altura = float(altura)
peso = input("Digite o seu peso: ")
peso = float(peso)
endereco =input("Digite o seu endereço: ")
print("Nome: %s "  % nome)
print("Idade: %d anos" % idade)
print("Altura: %.2f" % altura )
print("Peso: %.2f" % peso)
print("Endereço: %s" % endereco)
```

Expressões Aritméticas

- Uma expressão aritmética é o conjunto de operadores (aritméticos) e operandos (constantes ou variáveis numéricas) dispostos numa determinada ordem.

- O *resultado* de uma expressão aritmética sempre será *numérica*

Operadores Aritméticos

- Operadores aritméticos básicos
  - \+ adição 1 + 2
  - \- subtração 5 - 3
  - * multiplicação 3 * 4
  - / divisão 4 / 2

Operadores Aritméticos

- Operadores aritméticos auxiliares em Python
  - ** potenciação 2**3 = 8
  - math.sqrt radiciação math.sqrt(4) = 2
  - % resto divisão 4 % 3 = 1

- Prioridades
  - parênteses mais internos
  - pot rad
  - * / div mod
  - + -

Exemplo de operação aritmética

```
# Exemplo de operação aritmética
# A soma de dois números inteiros:

num1 = input('Digite um número inteiro: ')
num2 = input('Digite outro número inteiro: ')
num1 = int(num1)
num2 = int(num2)
soma = num1 + num2
# Três maneiras de apresentar o resultado:
print('A soma entre %d e %d vale %d' % (num1, num2, soma))
print('A soma entre', num1, 'e', num2, 'vale', soma)
print ('A soma entre {} e {} vale {}'.format(num1,num2,soma))
```

Exercícios

1. Desenvolva um programa que receba o salário de um funcionário, calcule e mostre seu novo salário com reajuste de 15%.
2. Desenvolva um programa que receba os valores do comprimento (C), da largura (L) e da altura (H) de um paralelepípedo, calcule e mostre o volume desse paralelepípedo. Fórmula do volume de um paralelepípedo:
   V = C . L . H
3. Desenvolva um programa que receba o raio (R) de uma circunferência, calcule e mostre a área dessa circunferência. Fórmula da área: A = PI * R^2, sendo que PI vale 3,14.
4. Desenvolva um programa que receba o número de horas trabalhadas por um funcionário e quanto esse funcionário recebe por hora trabalhada, calcule e mostre o valor que deve ser recebido por esse funcionário.

Expressões Lógicas

- Uma *expressão aritmética* é o conjunto de operadores (aritméticos) e operandos (constantes ou variáveis numéricas) dispostos numa determinada ordem.

- O *resultado* de uma expressão aritmética sempre será *numérica*

Expressões Lógicas

- Uma *expressão lógica* é um conjunto de operadores (relacionais ou lógicos) e operandos (relações, constantes ou variáveis inteiras, reais, alfanuméricas ou lógicas) dispostos numa determinada ordem.

- O *resultado* de uma expressão lógica sempre será *lógica*, ou seja, verdadeiro ou falso

Operadores Relacionais

- Em Python

>	maior que	3 > 2 verdadeiro
<	menor que	3 < 2 falso
>=	maior ou igual que	5 >= 7 falso
<=	menor ou igual que	5 <= 7 verdadeiro
==	igual	4 == 4 verdadeiro
!=	diferente	4 != 4 falso

Operadores Lógicos

- Em Python

not	negação
and	conjunção
or	disjunção

- Prioridades da esquerda para a direita, de cima para baixo

não

e ou

Estrutura condicional

- Comando *if*

Estruturas Condicionais

- Em algumas situações, o fluxo de execução do algoritmo necessita ser desviado ou alguma condição necessita ser testada.
- Dessa forma, as **Estruturas Condicionais** permitem a escolha de um grupo de **ações** a ser executado quando determinadas **condições**, representadas por **expressões lógicas** ou **relacionais**, são ou não satisfeitas.

Estruturas Condicionais

- Por exemplo, se o valor da média final for maior ou igual a 5, o aluno está aprovado:

  se (media >= 5)

  então mostrar "APROVADO"

- Por exemplo, se o salário bruto for maior que 1000 e menor que 2500, então o percentual de desconto do imposto de renda será de 10%:

  se ((SB >= 1000) e (SB<=2500))

  então IR = 10

- Nestas duas situações existe um teste (condição) para que alguma operação seja executada.

Primeiro exemplo

- Por exemplo, se o valor da média final for maior ou igual a 5, o aluno está aprovado:

se (media >= 5)

```
# Primeiro exemplo Estrutura condicional

nota1 = float(input('Entre com a primeira nota: '))
nota2 = float(input('Entre com a segunda nota: '))
media = (nota1 + nota2)/2
if media >= 5 :
    print('Aprovado com média %.2f' % media)
else :
    print('Reprovado com média %.2f' % media)
```

Segundo exemplo

- Por exemplo, se o salário bruto for maior que 1000 e menor que 2500, então o percentual de desconto do imposto de renda será de 10%:

se ((SB >= 1000) e (SB<=2500))

```
# Segundo exemplo Estrutura condicional
# se ( (SB >= 1000) e (SB<=2500) )
# então    IR = 10

sb = float(input('Entre com o salário base: '))

if sb >= 1000 and sb <=2500 :
    ir = sb * 0.10
    print('Imposto de renda a pagar: %.2f' % ir)
```

Terceiro exemplo

se ((SB >= 1000) e (SB<=2500))

então IR = 10

senão

se (SB > 2500)

então IR = 15

senão

IR = 0

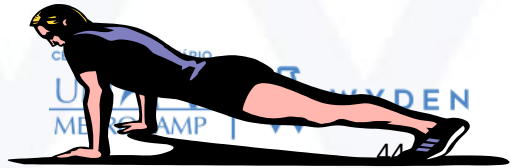
```
# Terceiro exemplo Estrutura condicional
# se ( (SB >= 1000) e (SB<=2500) )
# então    IR = 10
# senão
#    se (SB > 2500)
#     então IR = 15
#    senão
#      IR = 0


sb = float(input('Entre com o salário base: '))

if sb >= 1000 and sb <=2500 :
    ir = sb * 0.10
elif sb > 2500 :
    ir = sb * 0.15
else :
    ir = 0
print('Imposto de renda a pagar: %.2f' % ir)
```

Exercícios de Fixação

1. Fazer um programa para ler dois números inteiros e mostrá-los em ordem crescente.
2. Fazer um programa para mostrar uma mensagem na tela dizendo se um número inteiro lido é par ou ímpar.
3. Construa um programa que receba como entrada a altura e o sexo de uma pessoa (letra ‘F’ para Feminino e letra ‘M’ para Masculino). Em seguida, calcule e escreva o peso ideal dessa pessoa, utilizando as seguintes fórmulas:
   - para homens: (72.7 * altura) – 58;
   - para mulheres: (62.1 * altura) – 44.7;



Estrutura de Repetição Enquanto

- Uma estrutura de repetição **enquanto** pode ser utilizada quando o algoritmo precisa **testar determinada condição antes de executar um conjunto de comandos** repetidas vezes
- Se a condição avaliada for verdadeira, o conjunto de comandos dentro da estrutura de repetição **enquanto** é executado e após esta execução, a condição é novamente avaliada
- Se o resultado da avaliação for falso, este conjunto de comandos não será executado e o fluxo do algoritmo segue normalmente.
- Nesta estrutura de repetição, pode ocorrer do conjunto de comando não ser executado nenhuma vez.

Estrutura de Repetição Enquanto - Pseudocódigo

- Sintaxe da Estrutura de Repetição enquanto

<inicialização da variável de controle>;
enquanto *(<condição>)* ***faça***
<comando_1>;
<comando_2>;
...
<comando_n>;
<atualização da variável de controle>;
fimenquanto*;*

Nota: a ***<atualização da variável de controle>*** pode ser feita em qualquer parte dentro do enquanto, não necessariamente após o último comando.

Estrutura de Repetição Enquanto - Pseudocódigo

- **Exemplo**

```
x ← 0;
enquanto (x<3) faça
  escreva ("O valor de x é: " , x);
  x ← x + 1;
fimenquanto;
```

Nota 1: no exemplo acima, o ***x*** é ***<variável de controle>***. É ele que faz parte da condição do loop.

Nota 2: veja que o ***x*** também é usado no processamento dentro do loop. Portanto a variável ***x*** não é de uso restrito ao controle do loop.

Estrutura de Repetição Enquanto - Python

- Sintaxe da Estrutura de Repetição enquanto

<inicialização da variável de controle>;

while *<condição> :*

<comando_1>;

<comando_2>;

...

<comando_n>;

<atualização da variável de controle>;

Estrutura de Repetição Enquanto - Python

- Exemplo

```
#Exemplo while

x = 0
while x < 3 :
      print('O valor de x é: %d' % x)
      x = x + 1
print('Saiu do while')
```

Exercícios

1 – Desenvolva um programa que recebe números inteiros digitados pelo usuário e calcula a soma entre esses números e a média. Só parar de digitar os números quando o usuário digitar zero.

2 – Desenvolva um programa que recebe 10 números reais digitados pelo usuário e soma somente os números pares.

```python
# Desenvolva um programa que recebe números inteiros digitados
# pelo usuário e calcula a soma entre esses números e a média.
# Só parar de digitar os números quando o usuário digitar zero.

numero = 5
cont = 0
soma = 0

while numero != 0 :
    numero = int(input('Digite um número: '))
    if numero != 0 :
        soma = soma + numero
        cont = cont + 1
# saiu do laço
media = soma / cont
print ('A soma é igual a %d e a média é igual a %.2f' % (soma, media))
```

```python
# Desenvolva um programa que recebe 10 números reais digitados pelo usuário
# e soma somente os números pares.

cont = 0
soma = 0

while cont < 10 :
    numero = int(input('Digite um número: '))
    if numero % 2 == 0 :
        soma = soma + numero
    cont = cont + 1
# saiu do laço
media = soma / cont
print ('A soma dos pares é igual a %d' % (soma))
```

Estrutura de Repetição Para

- Uma estrutura de repetição **para** pode ser utilizada quando o algoritmo precisa ter definido a quantidade de vezes que um conjunto de comandos deve ser executado

- Neste caso, a variável de controle, sua inicialização e finalização bem como sua atualização fazem parte do cabeçalho da estrutura de repetição **para** e o conjunto de comandos dentro da estrutura de repetição **para** é executado a quantidade de vezes determinado no cabeçalho desta estrutura

- Note que nesta estrutura de repetição, pode ocorrer do conjunto de comandos não ser executado nenhuma vez

Estrutura de Repetição for em Python

- Exemplo

```
# Primeiro Exemplo de for

x = 0

for x in range (3) :    # x vai de 0 a 2
        print ('O valor de x é: %d' % x);
print ('Saiu do laço')
```

Estrutura de Repetição for em Python

- Segundo exemplo

```
# Segundo exemplo de for

for x in range (50, 100) : # x vai de 50 a 99
    if x == 88 :
        break              # se x for igual a 88, sai do laço
    print(x)
print('Saiu do for')
```

Estrutura de Repetição for em Python

- Terceiro exemplo

```
# Terceiro exemplo com for - soma dos números impares
total = 0
numero = int(input('Digite um número: '))
if (numero % 2) == 0:
    numero = numero -1
for i in range(numero,0,-2):     # i vai de numero até 0, decrementando de 2
    total = total + i
    print('Valor de i %d:' % i)
print("A soma dos números impares é %d " % total)
```

Exercícios

1 - Desenvolva um programa que calcule e o quadrado dos números inteiros compreendidos entre 10 e 150. **Utilizar for**.

2 - Desenvolva um programa que receba um número inteiro, calcule e mostre o seu fatorial. (Exemplo de Fatorial: se o número 4 for digitado, o programa deverá fazer 1*2*3*4 e mostrar como resultado 24, se o número digitado for 5 o programa deverá fazer 1*2*3*4*5 e mostrar como resultado 120). **Utilizar for**.

3 - Desenvolva um programa que recebe um número inteiro e mostra a tabuada desse número.

4 - Desenvolva um programa que receba um número inteiro, verifique e mostre se esse número é primo ou não.