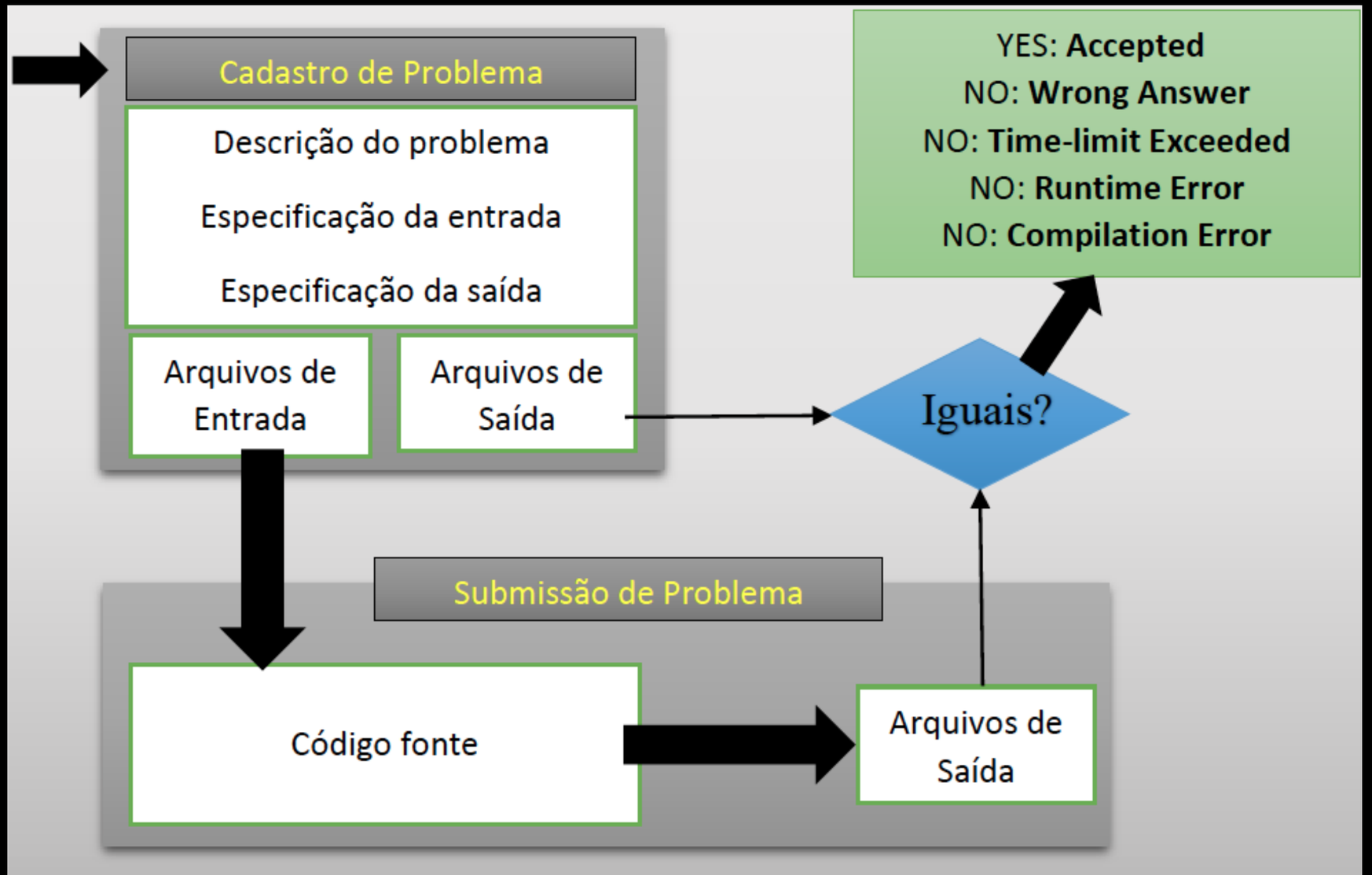
Dicas de Uso do The Huxley

Prof. Alberto Costa Neto

Introdução à Ciência da Computação
(Programação em Python)

Relembrando...



Dicas Gerais

- Atenção ao que o problema pede!
  - Para que um problema seja considerado resolvido, seu programa deve fazer exatamente o que foi pedido no problema (nem mais nem menos)
- Você não está programando para um usuário!
  - Portanto, em geral não é requerido que imprima na tela (print) mensagens amigáveis como ('Digite o seu nome:'). Apenas leia o nome e pronto! Simples e efetivo…
  - O sistema é que irá executar a digitação e leitura da saída e conferir, conforme comandos contidos em seu programa.

Mais Dicas Gerais

- O sistema é exigente e não é flexível como um ser humano
  - Portanto, se o problema pede que um número seja impresso com 2 casas decimais (9.15, por exemplo).
  - Não adianta colocar 3 casas decimais, mesmo sabendo que é mais preciso, porque estará errado.
- Também cuidado com letras maiúsculas e minúsculas
  - Se o problema pedir para imprimir uma mensagem APROVADO e você imprimir Aprovado, a resposta não confere com o esperado. Logo, fique atendo para isso também!

Entrada de Dados

- Sugerimos o uso da função raw_input
  - Ela recebe uma String como parâmetro. Este parâmetro será impresso na tela e o cursor ficará aguardando que algo seja digitado via teclado.

```
valor = raw_input('Digite seu Nome:')
```

  - A linha acima irá imprimir na tela

```
Digite seu Nome:_
```

  - E o cursor ficará aguardando que algo seja digitado após o caracteres dois pontos

Entrada de Dados

- Em geral, vamos apenas chamar raw_input(), sem passar parâmetros (a não ser que o problema exija)

```
valor = raw_input()
```

  - A linha acima não irá imprimir nada na tela

```
_
```

  - O cursor ficará piscando esperando que algo seja digitado e se aperte a teclar <ENTER>. Apenas após isso é que o valor digitado será armazenado na variável valor (do tipo String)

Entrada de Strings

- A função raw_input pega tudo que foi digitado na linha e retorna uma String, mesmo que tenha espaços ou números no meio

```
valor = raw_input()
```

- Exemplos de entrada e valores atribuídos à variável valor:

Este foi o valor digitado

valor = 'Este foi o valor digitado'

Digitei 25.0 e 10

valor = 'Digitei 25.0 e 10'

12.55

Valor = '12.55'

Entrada de Números Inteiros (int)

- A função raw_input() retorna uma String. Como posso ler um inteiro? Simples! Transforma-se o conteúdo da String em um número inteiro (em Python, são representados pelo tipo int)

```
valor = raw_input()
valor_inteiro = int(valor)
```

- valor continua sendo um valor String. Mas a variável valor_inteiro contém o inteiro que corresponde aos caracteres numéricos contidos na String. De forma resumida, poderíamos escrever apenas:

```
valor_inteiro = int(raw_input())
```

Entrada de Números em Ponto Flutuante (float)

- De forma semelhante aos inteiros, os números em ponto flutuante são lidos como String usando a função raw_input() e em seguida são convertidos para o tipo float (ponto flutuante/real).

```
valor = raw_input()
valor_float = float(valor)
```

Ou

```
valor_float = float(raw_input())
```

Vários dados em uma mesma linha da entrada

- Uma questão pode pedir, por exemplo, para ler a matrícula e 2 notas para calcular sua média final.
- Como fazer a separação dos dados se a função raw_input lê uma linha e retorna uma única String?

```
>>> entrada = raw_input()
2015021234 10.0 9.0
>>> print entrada
2015021234 10.0 9.0
>>> valores = entrada.split()
>>> print valores
['2015021234', '10.0', '9.0']
>>> matricula = int(valores[0])
>>> print matricula
2015021234
>>> nota1 = float(valores[1])
>>> print nota1
10.0
>>> nota2 = float(valores[2])
>>> print nota2
9.0
```

Saída de Dados

- Você pode fazer o programa todo correto, mas se não imprimir a saída conforme pedido no problema, o sistema considera a questão errada.
- Portanto, é importantíssimo estar atento a como a saída deve ser apresentada.
- Para executar a impressão de dados na tela, sugerimos usar o comando print
  - Ele recebe um ou mais valores separados por vírgula e imprime na tela os valores separados por espaços em branco.

Imprimindo ponto flutuante com casas decimais

- Muitas questões pedem para imprimir um número real (ponto flutuante) com um certo número de casas decimais.
- Neste caso sugerimos usar o operador % que no caso de Strings, serve para formatação (no estilo C)

```
>>> import math
>>> print math.pi
3.14159265359
>>> print '%6.4f' % math.pi
3.1415
>>> print '%6.3f' % math.pi
 3.142
>>> print '%7.3f' % math.pi
  3.142
>>> print '%.3f' % math.pi
3.142
```

Imprimindo vários dados em uma mesma linha da saída

- Sugerimos o uso do comando print
  - Ele recebe um ou mais valores separados por vírgula e imprime na tela os valores separados por espaços em branco.
  - Uma alternativa é concatenar Strings

```
>>> print 'Ok'
Ok
>>> print 'Valor =', 'Ok'
Valor = Ok
>>> print 'Valor = ' + 'Ok'
Valor = Ok
>>> print 'Valor =', 10
Valor = 10
>>> print 'Valor = ' + 10
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str'
and 'int' objects
>>> print 'Valor = ' + str(10)
Valor = 10
```

Respostas dos Juízes *on-line*

Operador	Operação
YES/Aceito/Correto	Seu programa está correto. Passou em todos os casos de teste.
NO: Incorrect Output / Resposta Errada	Seu programa executou, mas deu alguma resposta errada.
NO: Time-limit Exceeded / Limite de tempo excedido	O programa demorou muito executando e foi abortado.
NO: Runtime Error / Erro em tempo de execução	O programa gerou um erro durante a execução, sendo impossível concluir a execução.
NO: Compilation Error / Erro em tempo de compilação	O programa possui algum erro de sintaxe. Isto inviabiliza a execução do mesmo.
NO: Output Format Error / Presentation Error / Erro de Apresentação	O programa não segue a especificação de formatação da saída, mas o resultado aparenta estar correto.