Exceções em Python

Introdução

- Como tratar erros durante a execução de um programa?
  - Situações esperadas *versus* Situações inesperadas

Introdução

- Exemplo de erro em Python:
  - Traceback (most recent call last):

    File "<stdin>", line 1, in \<module\>

    IndexError: list index out of range

Programa de exemplo

- Programa em Python que calcula as 4 operações básicas:
  - http://github.com/esponja92/lingprog20182

O que são Exceções?

- “Exceção” é a indicação de que ocorreu um problema durante a execução do programa, interrompendo a execução do programa
- O tratamento de exceções é fornecido pelas linguagens de programação para permitir aos programas capturar e tratar erros em vez de deixá-los ocorrer

Vantagens e Desvantagens

- Vantagens:
  - Separar o código do programa do código de tratamento de exceções
  - Permite a propagação de erros sem a necessidade de testar o retorno
  - Tratamento de erros como objetos

Vantagens e Desvantagens

- Desvantagens:
  - Não existe suporte nativo nos processadores ao tratamento de exceções
  - Risco de queda de performance do programa

Algumas Exceções utilizadas em Python

Classe	Descrição
Exception	Classe base para todas as exceções
AttributeError	Falha no acesso ou atribuição a atributo de classe
IOError	Falha no acesso a arquivo inexistente ou outros de E/S
IndexError	Índice inexistente de seqüência
KeyError	Chave inexistente de dicionário
NameError	Variável inexistente
SyntaxError	Erro de sintaxe (código errado)
TypeError	Operador embutido aplicado a objeto de tipo errado
ValueError	Operador embutido aplicado a objeto de tipo certo mas valor inapropriado
ZeroDivisionError	Divisão ou módulo por zero

Capturando Exceções em Python

```
try:
    # código a ser executado "sob vigilância"
except Exception:
    # caso ocorrer alguma exceção no código
    # acima, trate-a aqui.
```

Capturando Exceções em Python

- Podemos capturar exceções passando “Exception” como:
  - Classe
  - Classe as var
  - (Classe1, Classe2, …, ClasseN)
  - (Classe1, Classe2, …, ClasseN) as var
- Podemos printar a mensagem de erro usando print(e)

O que fazer após capturar uma exceção?

- De acordo com os requisitos do projeto no qual está trabalhando, o desenvolvedor poderá:
  - Jogar na tela a mensagem de erro
  - Tratar a situação para que o programa volte a um estado *consistente*
  - Simplesmente não fazer nada!

Levantando Exceções

>>> raise exceção(“mensagem da exceção”)

Exemplo:

```
if valor % 2 != 0:
    raise ValueError(“Apenas pares são permitidos”)
```

Criando Exceções

- Basta escrever uma classe que herde da classe Exception
- class MinhaException(Exception):

  #codigo da exceção

- >>> raise MinhaException(“Deu Ruim!”)

Boas práticas de programação

1. Não use exceções para o fluxo principal da aplicação
2. Interceptar exceções somente quando você sabe como tratá-la
3. Conhecer as exceções da biblioteca padrão

Referências

- https://pythonhelp.wordpress.com/2012/09/14/tratamento-de-excecoes
- http://orion.lcg.ufrj.br/python/_10%20-%20Programando%20em%20Python%20-%20Excecoes.pdf
- Java - Como programar (4ª edição)
- https://www.ppgia.pucpr.br/~alcides/Teaching/SistemasDistribuidos/TOF/10_excecoes.pdf
- https://homepages.dcc.ufmg.br/~rimsa/documents/decom009/lessons/Aula08.pdf