SCC 124 - Introdução à Programação para Engenharias

Módulos

python

Professor: André C. P. L. F. de Carvalho, ICMC-USP
Pos-doutorando: Isvani Frias-Blanco
Monitor: Henrique Bonini de Britto Menezes

Tutoria

- Aluno com nota maior ou igual a 7 pode ser tutor de aluno com nota inferior a 5
  - Aluno com nota inferior a 5 procura tutor, que pode ou não aceitar
    - Mais de um aluno pode procurar o mesmo tutor, que pode escolher apenas um aluno dentre eles
  - Tutor ganhará em uma das provas metade do que conseguir aumentar na nota do aluno tutorado

Possíveis tutores

Amanda Bolsoni Ferreira
Arthur Tristan Dourado Beloso
Diego Prochnow Fracasso
Elvio Jose Seneda Filho
Guilherme Garcia de Vasconcelos
Guilherme Peres Finardi
Hugo Murarolli Lazaro
Isabela Scarpin Contatto
Joao Marcos Ribeiro dos Santos
Marilia Nori de Menezes Ravagnani
Paulo Augusto Fernandes de Souza
Silvio Levcovitz
Vinicius Soares Machado
Vitor Hugo Rocha Pereira

Aula de hoje

- Introdução
- Módulos
- Importação de módulos
- Pacotes
- Importação de pacotes
- Exemplos

Introdução

- Funções
  - Permitem reutilizar códigos nos programas
  - Geralmente formadas por blocos de comandos
- Módulos
  - Permitem reutilizar um conjunto de funções de um programa em outros programas
    - Reutilizar programas
  - Podem ser escritos em Python ou na linguagem em que o interpretador Python foi implementado
    - Linguagem C

Módulo

- Permite incluir várias definições em um arquivo e usá-las
  - No modo script
  - No modo interativo
- Pode ser importado por qq outro módulo
  - Inclusive pelo módulo *main*
  - Todo programa em Python é um módulo
    - Qualquer arquivo Python que contém código pode ser importado como um módulo

Módulo

- É importado da mesma forma que as bibliotecas de Python são importadas
  - Ex.: *import nome-modulo*
  - Importação de um módulo permite usar suas definições (classes, funções, variáveis)
  - Comandos de importação de módulos geralmente ficam no início de um módulo
    - O ideal eh que o arquivo do módulo esteja no mesmo diretório do programa que o importa

Módulo

- Arquivo contendo definições e comandos na linguagem Python
  - Nome do arquivo termina com sufixo *.py*
  - Dentro de um módulo, sua versão pode ser disponibilizada como uma variável global
    - *__nome__ = numero-da-versao*

Módulo

- Todo programa em Python é um módulo

```
def sauda ():  # Salvo como meumodteste.py
    print (" Ola, sou eu, seu modulo")
__versao__ = '1.0'
```

Outro arquivo:

```
import meumodteste  # Salvo como meumodteste

meumodteste.sauda()
print('Versao', meumodteste.__versao__)
```

Saida:

Ola, sou eu, seu modulo
Versao 1.0

Exemplo

```
# modulo series de numeros ate n
def fibonacci(n):    # serie de Fibonacci
    a, b = 0, 1
    while b < n:
        print(b, end=' ')
        a, b = b, a+b
    print()

def triangulo(m):  # serie de numeros triangulares
    a,b = 1,1
    while b < m:
        print(b, end=' ')
        b = int(a*(a+1)/2)
        a = a + 1
    print()

__versao__ = '1.0'
```

Salvar arquivo como series.py

Séries de números

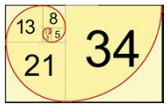
- Modelam vários fenômenos importantes
- Seguem regra ou fórmula
- Exemplos
  - Sequência de Fibonacci
  - Sequência de números triangulares

Sequência de Fibonacci

- Final do século 12
  - Importantes contribuições do matemático italiano Leonardo Pisano Bigollo (Leonardo de Pisa)
    - Apelido: Leonardo Fibonacci (filho de Bonacci)
    - Considerado um dos melhores matemáticos do ocidente
- Fibonacci popularizou sistema indo-arábico
  - Antes usava-se os dedos ou algarismos romanos
  - Introduziu na Europa a sequência de Fibonacci
    - Exemplo no Liber abbaci (livro de cálculos), 1202

Sequência de Fibonacci

- Inspiração:
  - Quão rápido coelhos podem se reproduzir em uma situação ideal
  - Quantos pares serão gerados em um ano
    - 1 par no início, 1 par após 1 mês, 2 pares após 2 meses, 3 pares após 3 meses, ...
      - 1, 1, 2, 3, 5, 8, 13,...
    - Também se aplica a:
      - Número de pétalas de flores
      - Número de ramos em plantas
      - ...



Sequencia de números triangulares

- Encontrado pelo mmatemático alamão Johann Carl Friedrich Gauss xéculo XIX
- Conta número de objetos que forma um triângulo equilátero
- Número do n- ésimo triângulo
  - Número de pontos que compõe um triângulo com n pontos em um dos lados

Sequencia de números triangulares

- Padrão de pontos que forma um triângulo
  - 1, 3, 6, ..

Número do n- ésimo triângulo é o número de pontos que compõe um triângulo com n pontos em um dos lados

$T_1 = 1, T_2 = 3, T_3 = 6, \quad T_4 = 10, \quad T_5 = 15$

Exemplo

- Entrando o nome do módulo, é possível acessar suas funções (modo interativo)

```
>>> import series
>>> series.fibonacci(100)
1 1 2 3 5 8 13 21 34 55 89
>>> series.triangulo(100)
1 1 3 6 10 15 21 28 36 45 55 66 78 91
>>> series.__name__
'series'
>>> series.__versao__
1.0
```


Exemplo

- Entrando o nome do módulo, é possível acessar suas funções (script: PyCharm)

```
import series

series.fibonacci(100)
series.triangulo(100)
print (series.__name__)
print (series.__versao__)
```

Saida:

```
1 1 2 3 5 8 13 21 34 55 89
1 1 3 6 10 15 21 28 36 45 55 66 78 91
series
1.0
```


Exemplo

- Suponha que uma função do módulo seja utilizada com frequência
  - Seu nome pode ser simplificado para um nome menor, mais simples

```
>>> import series
>>> fib = series.fibonacci
>>> fib(500)
```

```
import series
fib = series.fibonacci
fib(100)
```

```
1
1
2
3
5
8
13
21
34
55
89
```

Módulos

- Um módulo também pode conter funções executáveis
  - Servem para inicializar o módulo
  - São executados apenas na primeira vez que o nome do módulo é encontrado em um comando *import*

Módulos

- Cada módulo tem sua própria tabela de símbolos (nomes) particular
  - Que é usada como tabela de símbolos global pelas funções do módulo
    - Nomes dos símbolos dos módulos importados são colocados na tabela de símbolos do módulo importador

Módulos

- Autor de um módulo pode usar variáveis globais
  - Sem se preocupar com confusões que possam ocorrer com varáveis globais definidas pelo usuário
    - Mesma notação usada para funções pode ser utilizada
      - Ex.: *nome-modulo.nome-variavel-global*

Módulos

- Cada módulo é importado apenas uma vez em uma sessão do interpretador
  - Por motivos de eficiência
- Se o módulo for alterado, o interpretador precisa ser reinicializado
  - Se for apenas um módulo, pode ser usado o comando importlib.reload()
    - Ex.: *import importlib*
      importlib.reload(nome-modulo)

Módulos

- Python vem com uma biblioteca de módulos padrão
- Exemplos
  - Módulo *winreg*
    - Fornecido apenas para SO windows
  - Módulo *sys*
    - Parte de todo interpretador Python
    - Entre outros, define o prompt usado (>>>)

Pacotes (packages)

- Permitem incluir mais de um módulo em um ambiente único
  - Autores de diferentes pacotes não precisam se preocupar com nomes dos módulos utilizados em outros pacotes
    - Ex.: *import nome-pacote.nome-modulo*
  - Pacotes podem incluir subpacotes, para reunir funções com fins relacionados

Pacotes

- Pacotes conhecidos
  - Pacote *NumPy*
  - Pacote *Python Imaging Library*
- Supor que você quer projetar um um pacote para manipular arquivos e dados de sons
  - Existem vários formatos de arquivos de som diferentes
  - Serão necessárias funções para converter formatos

Pacote sound

- Inclui funções para diferentes operações sobre os dados
  - Mixagem
  - Adicionar eco
  - Adicionar função para equalização
  - Criar efeito estéreo artificial
  - ...


```
sound/                  # Nível mais alto do pacote
    __init__.py         # Inicializa pacote sound
    formats/            # Subpacores para conversoes de formato de arquivo
        __init__.py
        wavread.py
        wavwrite.py
        aiffread.py
        aiffwrite.py
        auread.py
        auwrite.py
        ...
    effects/            Subpacote para efeitos sonoros
        __init__.py
        echo.py
        surround.py
        reverse.py
        ...
    filters/            Subpacote para filtRos
        __init__.py
        equalizer.py
        vocoder.py
        karaoke.py
        ...
```

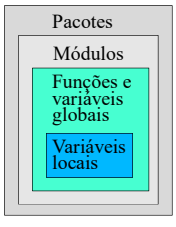
Pacotes

- Pacote pode apresentar uma estrutura hierárquica de módulos
- Como importar módulos de um pacote?
  - Ex.: *import sound.effects.echo*
    - Carrega o módulo sound.effects.echo
- Como usar uma função de um módulo de um pacote?
  - Ex:. sound.effects.echo.echofilter(input, output, delay=0.7, atten=4)

Pacotes

- Alternativas para importação de módulos:
  - Ex.: from sound.effects import echo
    - Pode ser usado sem o prefixo do pacote
      - Ex.: *echo.echofilter(input, output, delay=0.7, atten=4)*
  - Importação da função ou variável diretamente:
    - Ex.: from sound.effects.echo import echofilter
      - Carrega o módulo echo mas torna sua função echofilter() diretamente disponível
        - Ex.: *echofilter(input, output, delay=0.7, atten=4)*
  - Recomenda-se evitar o uso de from

Hierarquia de Programas em Python



Conclusão

- Módulos
  - Reúnem funções relacionadas
- Importação de módulos
- Pacotes
- Importação de pacotes
- Exemplos
- Uso de *from* torna o programa menos legível
- Quanto mais explícito, melhor

Perguntas

Exercício

- Assumindo que as únicas operações aritméticas presentes no interpretador são soma e subtração
  - Escrever um módulo em Python com funções para divisão e multiplicação
  - Escrever uma função, que importa esse módulo para converter valores decimais para binários e vice-versa