Overview

A Simple Python Program

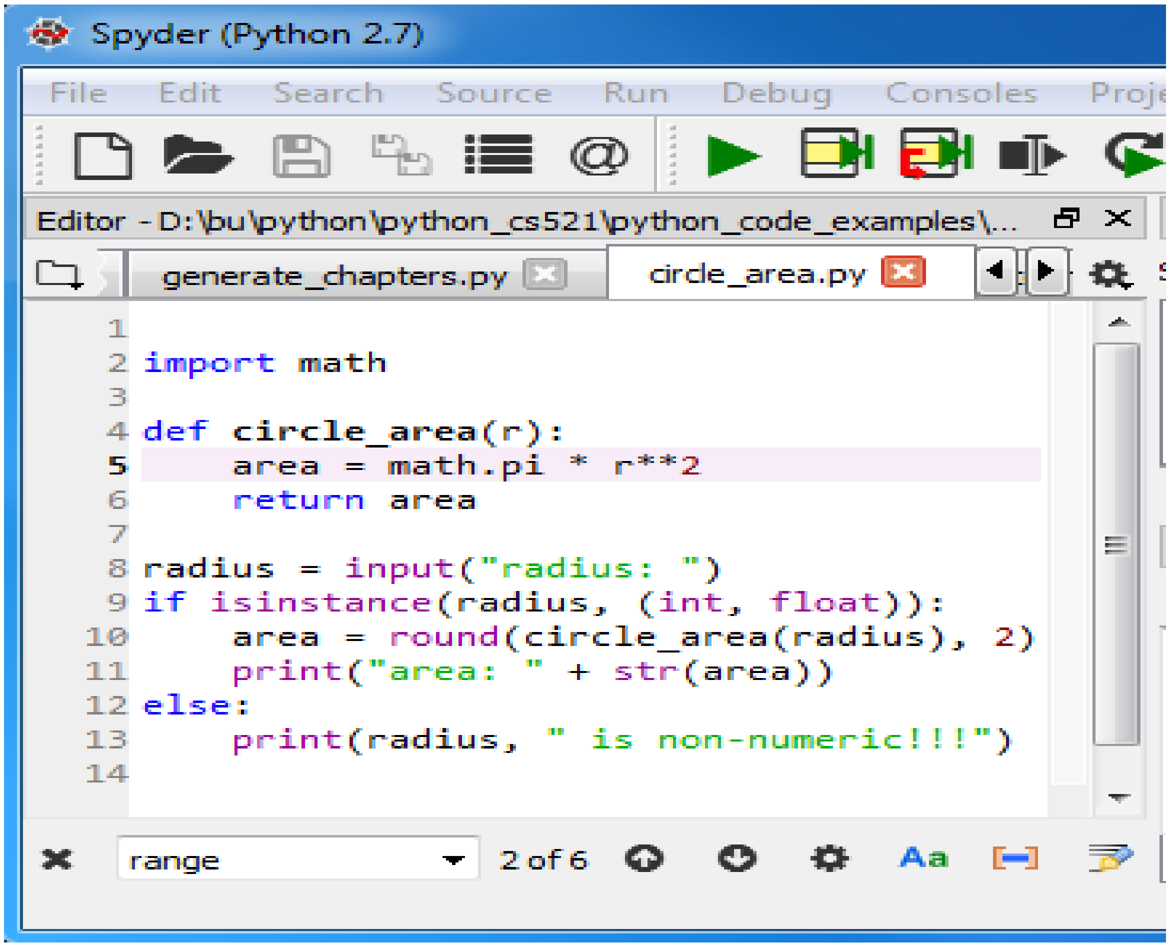
```
import math

def circle_area(r):
    area = math.pi * r**2
    return area

radius = input("radius: ")
if isinstance(radius, (int, float)):
    area = round(circle_area(radius), 2)
    print("area: " + str(area))
else:
    print(radius, " is non-numeric!!!")
```

Comments and Indentation

```
import math
# one line comment
def circle_area(r):
    return area = math.pi * r**2
“”””
Recommended indentation is 4 spaces
“”””

radius = input("radius: ")
if isinstance(radius, (int, float)):
    area = round(circle_area(radius), 2)
    print("area: " + str(area))
else:
    print(radius, " is non-numeric!!!")
```

- indentation should be consistent

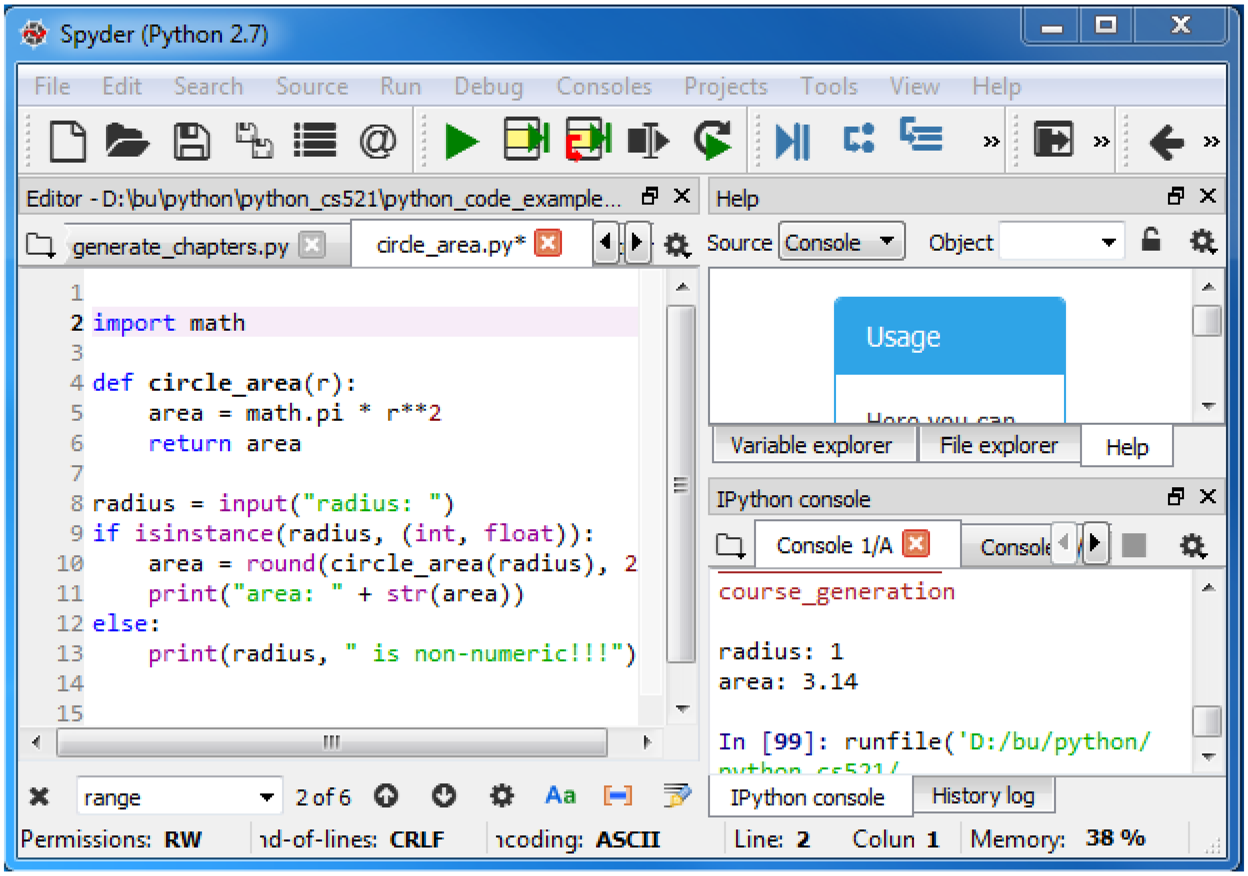
Python Program Structure


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Proje
Editor - D:\bu\python\python_cs521\python_code_examples\...
generate_chapters.py
circle_area.py
range
2 of 6

```

import math

def circle_area(r):
    area = math.pi * r**2
    return area

radius = input("radius: ")
if isinstance(radius, (int, float)):
    area = round(circle_area(radius), 2)
    print("area: " + str(area))
else:
    print(radius, " is non-numeric!!!")

```

- programs contain modules
- modules contain statements
- statements contain expressions
- expressions create and process objects

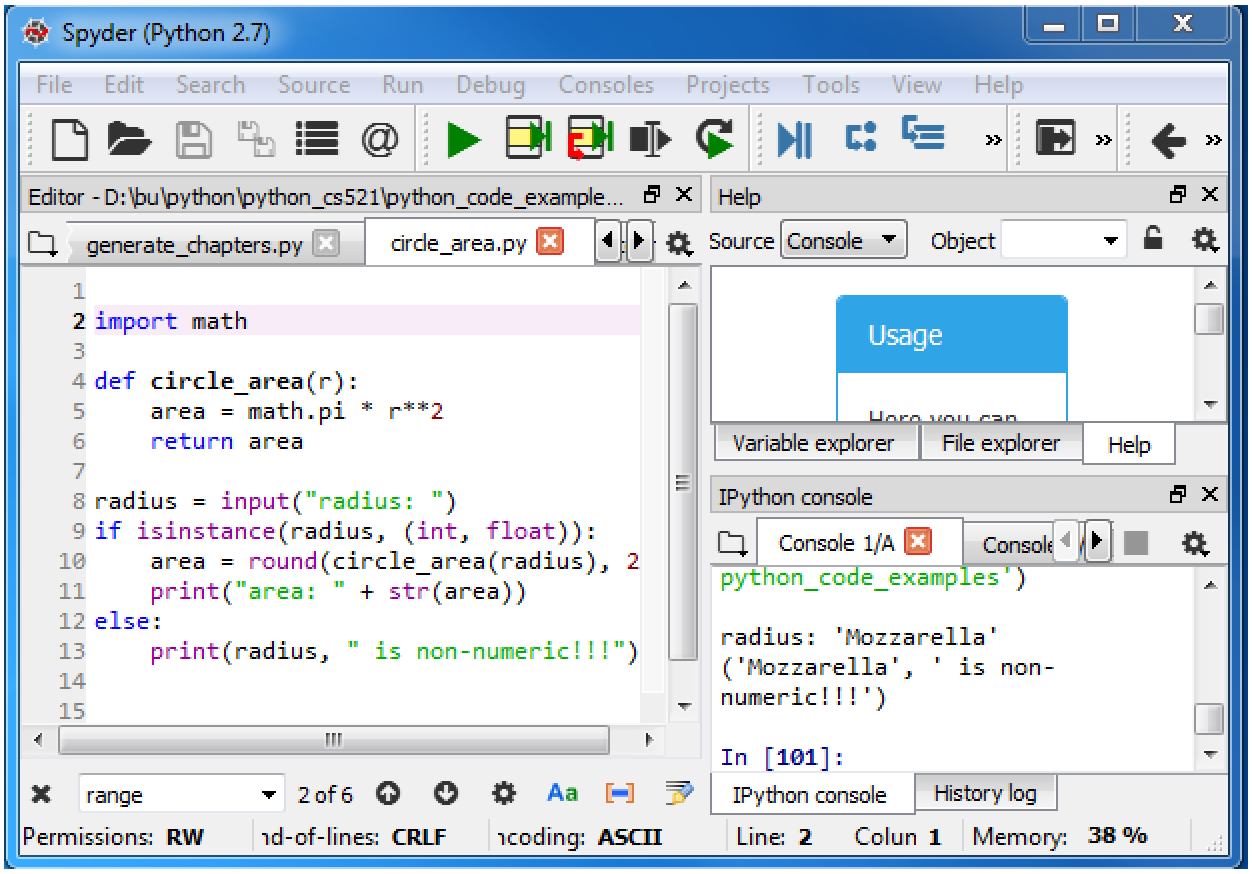
Sample Valid Run (iPython)


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - D:\bu\python\python_cs521\python_code_example...
generate_chapters.py
circle_area.py*
Help
Source Console Object
Usage
Variable explorer File explorer Help
IPython console
Console 1/A
IPython console History log
range 2 of 6
Permissions: RW nd-of-lines: CRLF ncoding: ASCII Line: 2 Colun 1 Memory: 38 %

```
import math

def circle_area(r):
    area = math.pi * r**2
    return area

radius = input("radius: ")
if isinstance(radius, (int, float)):
    area = round(circle_area(radius), 2
    print("area: " + str(area))
else:
    print(radius, " is non-numeric!!!")
```

```
course_generation

radius: 1
area: 3.14

In [99]: runfile('D:/bu/python/
```

Sample Invalid Run (iPython)



```

import math

def circle_area(r):
    area = math.pi * r**2
    return area

radius = input("radius: ")
if isinstance(radius, (int, float)):
    area = round(circle_area(radius), 2
    print("area: " + str(area))
else:
    print(radius, " is non-numeric!!!")
```

```
python_code_examples')

radius: 'Mozzarella'
('Mozzarella', ' is non-
numeric!!!')

In [101]:
```

Example: “Golden” Ratio

```
import math


if  __name__==”__main__”:
    x = math.sqrt(5)
    golden_ratio = (x – 1.0)/2
    print(‘x: ‘, round(x, 4))
    print( ‘ratio: ‘, round(golden_ratio,4))
```

```
>>>
runfile('D:/bu/python/python_cs521/python_
code_examples/misc.py')
('x: ', 2.2361)
('ratio: ', 0.618)
>>>
```

Conventions and Syntax

- program consists of statements

```
>>> x = 5.0 + math.sqrt(5)
```

- each statement terminated with newline
- multi-line statements with continuation ‘\’

```
>>> x = 5.0 + \
        math.sqrt(5)
```

- no ‘\’ for multi-line within (), [], {}, “””

```
 ('ratio: ', 0.618)
>>> x = [‘Mozzarella’,
          ‘Brie’]
```

- line comments start with #
- multiple statements separated with ‘;’

```
>>> x = 5 + math.sqrt(5); y = math.pi/2
```

Python Math Module

```
>>> help('math')
Help on built-in module math:
NAME
     math
FILE
    (built-in)
DESCRIPTION
This module is always available.  It provides access to the
mathematical functions defined by the C standard.


FUNCTIONS
     sin(...)
        sin(x)
        Return the sine of x (measured in radians).
    sqrt(...)
        sqrt(x)
        Return the square root of x.
DATA
    e = 2.718281828459045
    pi = 3.141592653589793
```

Importing Modules

- can import all functions or just some

```
>>> import math              # import all
>>> from math import sin     # some
>>> from math import *       # import math
```

- overwrites objects with the same name
- preferred solution is

```
>>> import A_module as A
>>> import B_module as B
```

- can distinguish functions with the same function name from different modules:

```
>>> x = A.function_name()
>>> y = B.function_name()
```

Python Modules

- collection of variables and functions
- definitions are imported
- file name is module name and ‘py’ extension
- some well-known modules:
  NumPy – numerical python (matrices)
  Pandas – panel data (tables)
  Matplotlib – plotting
  SciPy – scientific Python
  Sklearn – machine learning

__*main*__ Namespace

```
import math

def circle_area(r):
    area = 2 * math.pi * r**2
    return area

if  __name__==”__main__”:
    radius = input("radius: ")
    if isinstance(radius, (int, float)):
        area = round(circle_area(radius), 2)
        print("area: " + str(area))
    else:
        print(radius, " is non-numeric!!!")
```

- __main__ is the namespace of the Python interpreter

Namespaces and Objects

- namespace - a list of identifiers assigned to objects
- namespaces have identifiers
- “__main__” namespace for interpreter
- Python allows to include files of objects and functions – modules
- math.py - math functions and constants

Review Problems

Interview Problem

- is Python object oriented?
- what is object oriented programming?

Interview Problem

- how are the functions *help*() and *dir*() different?

Interview Problem

- what is a heap memory?

Interview Problem

- how is memory managed in Python?

Interview Problem

- what is garbage collection?

Interview Problem

- name 5 modules included by default

Interview Problem

- difference between .py and .pyc files

Interview Problem

- explain the difference between local and global namespaces

Interview Problem

- what is a Python module?

Interview Problem

- what is *docstring*?

Interview Problem

- name the four main types of namespaces in Python

Interview Problem

- how to redirect the output of a python script from standout (i.e. monitor) on to a file?

Interview Problem

- which command do you use to exit help window or help command prompt?

Interview Problem

- does the functions *help*() and *dir*() list the names of all the built_in functions and variables? If no, how would you list them?

Interview Problem

- explain how Python does Compile-time and Run-time code checking?

Interview Problem

- why does all the memory is not de-allocated / freed when Python exits?

Interview Problem

- what are different features of Python?