Object Oriented Programming

CSE - 207

Lecture 1

Course Objectives

- Learn the fundamentals of Object Oriented programming.
- Understand OOP principles and features and how to apply them in Python.
- Learn the basics of Python Classes, Objects & Interfaces.
- Get extensive hands-on experience with Python programming.

Mark Distribution

S.L.	Exam	Mark	Distribution (Percentage)	
1	Midterm	20		
2	Lab	30	Quiz	20%
			Lab test	30%
			Assignment	50%
3	Teacher Assessment	10	Class Attendance, Class Performance	
4	Final	40		
	Total	**100**		

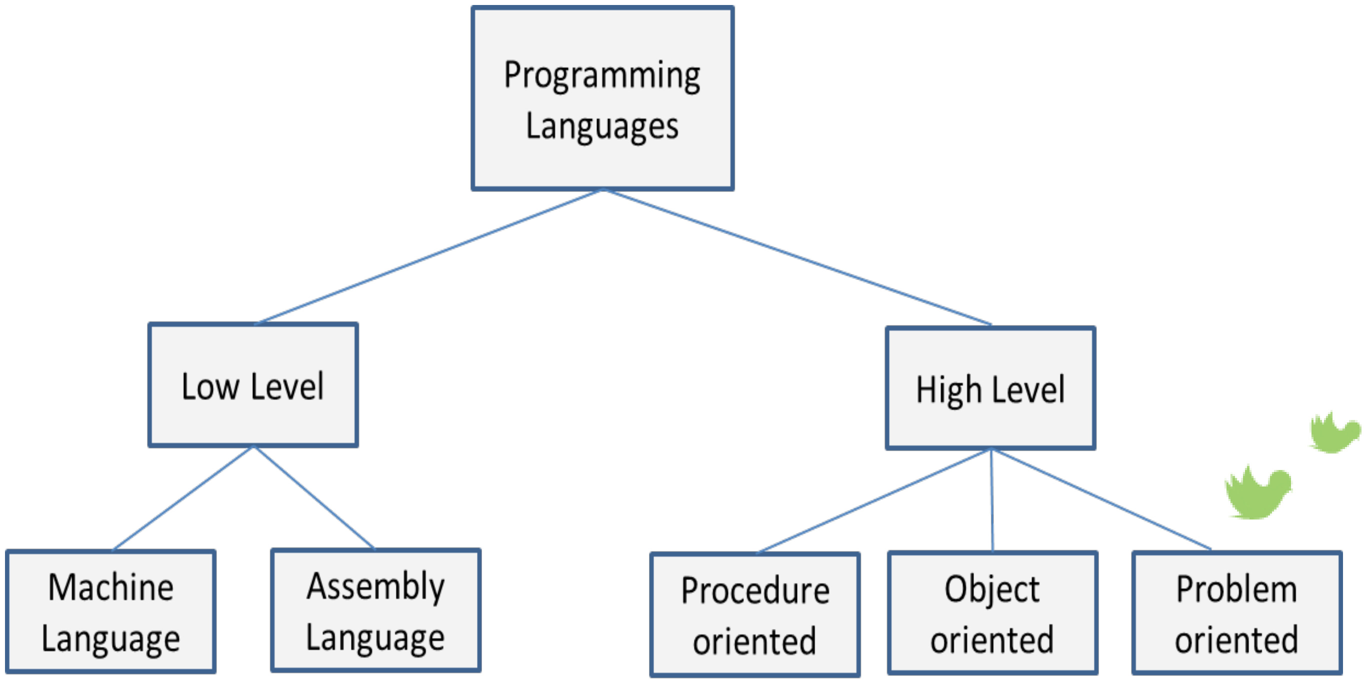
Course Outline

Lecture	Topic
1	Python Overview, Numbers, Strings
2	Loops, List, Tuples, Dictionary
3	Classes, Objects
4	Functions and Built-in functions
	Midterm
5	List comprehension, map, filter, functions lambda
6	Modules and Sub-process Modules
7	Files I/O, Exception
8	ML Packages (Numpy, Matplotlib, Pandas)
9	Exercise/ Problem solve
10	**Quiz Test, Lab Test**
11	Online Problem solve / contest

What Programming Language ?

A programming language can be defined as – “The language used for expressing a set of instructions that can be executed by the computer”.

Programming languages can be divided into two major categories:
low level and **high level** languages.



Generations of Programming Languages

Since the advent of computers, programming languages have evolved from unstructured, difficult-to-read languages to object oriented, easy-to-read programming languages.

1. First generation Languages (1GL)
2. Second generation Languages (2GL)
3. Third generation Languages (3GL)
4. Fourth generation Languages (4GL)
5. Fifth generation languages (5GL)

The Timeline of Development of PL

Year	Language	Remarks
1950-56	Assembly	Low level languages
1957	FORTRAN	High level language for Sc. Applications
1958	ALGOL	Algorithmic language supporting block structures
1960	COBOL, LISP	Business programming and List processing
1962	Simula	Simulation programming, first OOP language
1964	BASIC	General purpose, easy to program language
1970	PROLOG, Hope	Logic programming language suitable for AI applications Hope was a functional programming language
1972	‘C’	High level language suitable for system programming
1973	PASCAL	Block structured language

The Timeline of Development of PL

Year	Language	Remarks
1983	Smalltalk, Ada	OOP languages
1984	ML	Functional programming
1986	C++, Eiffel	OOP languages
1990	Haskell	Functional programming
1990-95	Perl, Python, JavaScript, PHP	Scripting languages
1995	Java	OOP language suitable for Internet programming
2000	C#	A multi-paradigm programming language
2005-2006	Ruby on Rails	Web application framework
2010	(Standard) PHP	Scripting language
2014	iOS/swift	Programming language for iOS and OS X developers

What is Python ?

“Python is a high-level programming language, and its core design philosophy is all about code readability and a syntax which allows programmers to express concepts in a few lines of code.”

Properties of Python

- **Python is Interpreted:** Python is processed at run-time by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive:** One can start a Python prompt and interact with the interpreter directly by writing code.

- **Python is Object-Oriented:** Python supports Object-Oriented style of programming that encapsulates code within objects.

- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to games.

Advantages of Python

- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable:** Low-level modules can be added to the Python interpreter.
- **Databases:** Python provides interfaces to all major commercial databases.
- **Readable**: Intuitive and strict syntax.

Advantages of Python

- **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- **Extendable:** Low-level modules can be added to the Python interpreter.
- **Scalable:** Python provides a better structure and support for large programs than shell scripting.
- **Integration**: It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

What is Python used for ?

- **Web Development**
  Django, Flask
- **Data Analysis**
  NumPy, Pandas
- **Data visualization**
  Matplotlib, Seaborn
- **Internet Of Things**
  Raspberry Pi + python
- **Machine Learning**
  Scikit-Learn, scipy
- **Web Scraping**
  scrapy
- **Computer Vision**
  OpenCV library
- **Deep Learning**
  NLTK, TensorFlow, PyTorch
- **Game Development**
  PyGame

Popular Python IDE

Anaconda is the world’s most popular Python data science platform (Everything you need 'out of the box').

Includes:

Spyder (IDE/editor - like pycharm) and

Jupyter

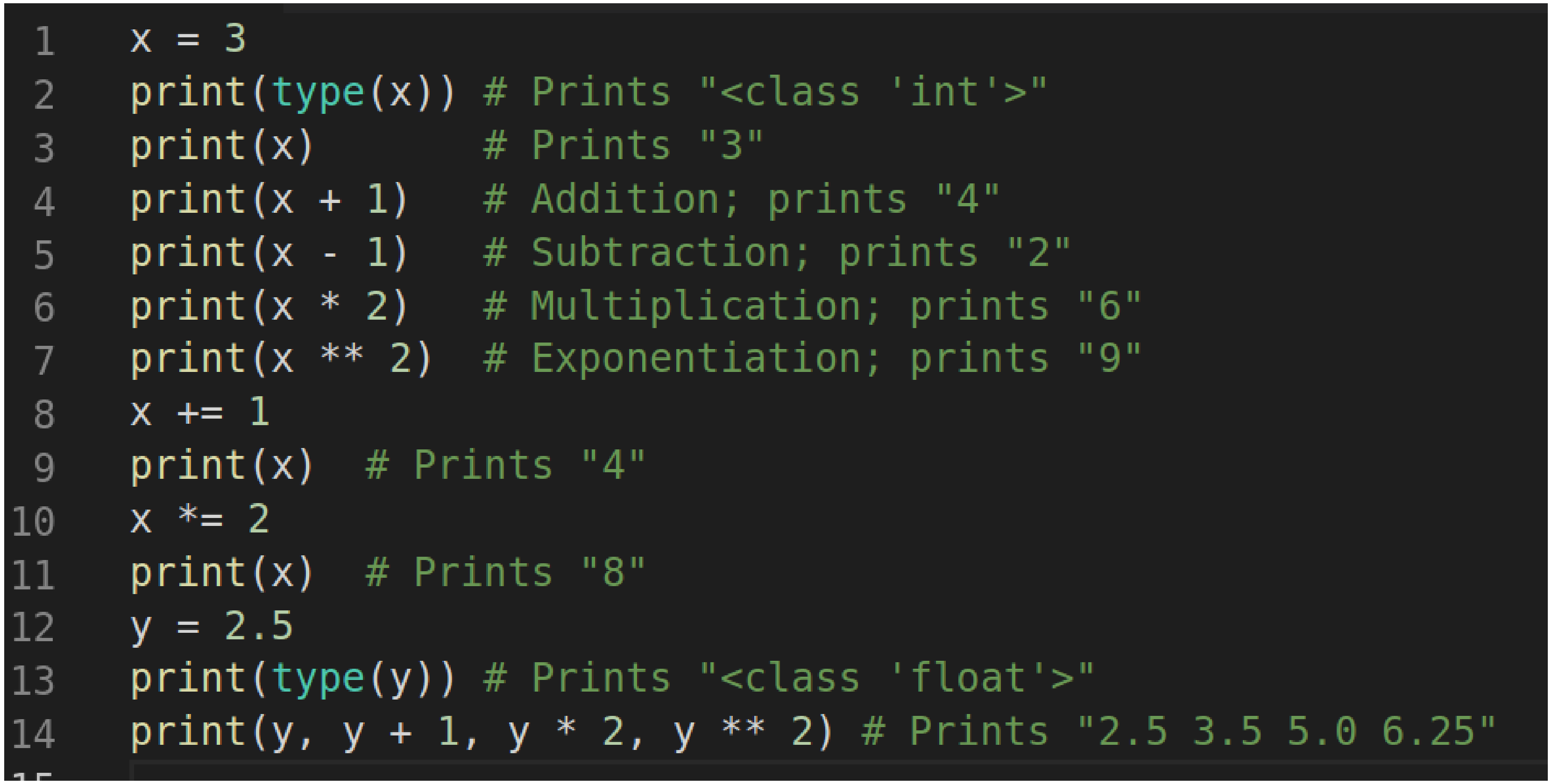
Python Code

Basic Data Types

- Python has a number of basic types including integers, floats, booleans, and strings
- These data types behave in ways similar to other programming languages.

Python Code

Numbers: Integers and floats



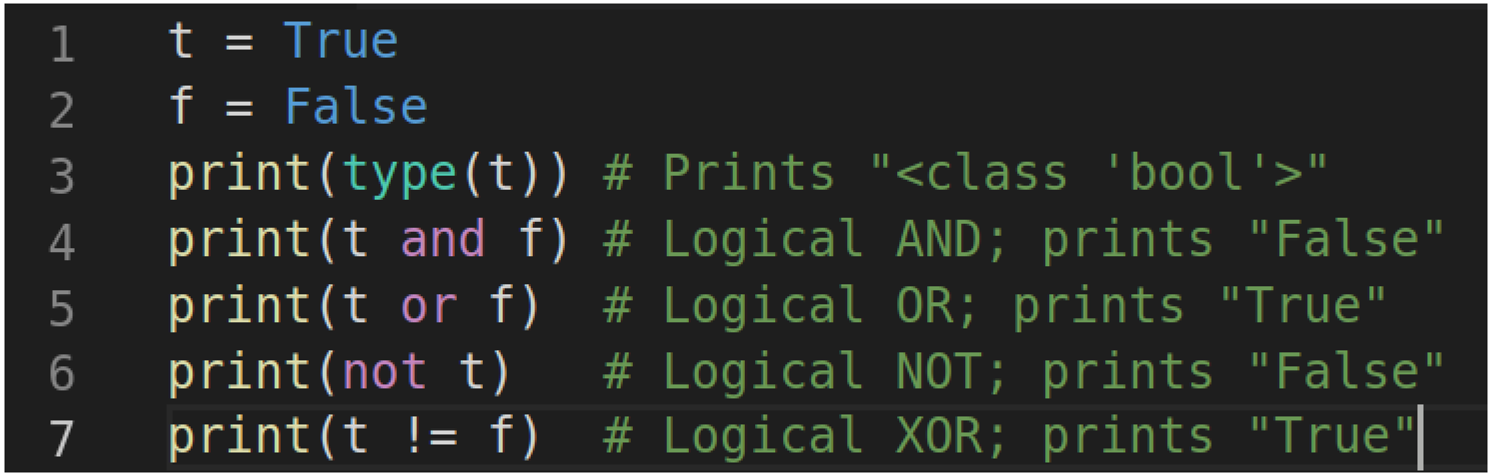
```python
x = 3
print(type(x)) # Prints "<class 'int'>"
print(x)       # Prints "3"
print(x + 1)   # Addition; prints "4"
print(x - 1)   # Subtraction; prints "2"
print(x * 2)   # Multiplication; prints "6"
print(x ** 2)  # Exponentiation; prints "9"
x += 1
print(x)  # Prints "4"
x *= 2
print(x)  # Prints "8"
y = 2.5
print(type(y)) # Prints "<class 'float'>"
print(y, y + 1, y * 2, y ** 2) # Prints "2.5 3.5 5.0 6.25"
```

Python does not have unary increment (x++) or decrement (x--) operators.

Python Code

Booleans

Python implements all of the usual operators for Boolean logic, but uses English words rather than symbols (&&, ||, etc.):

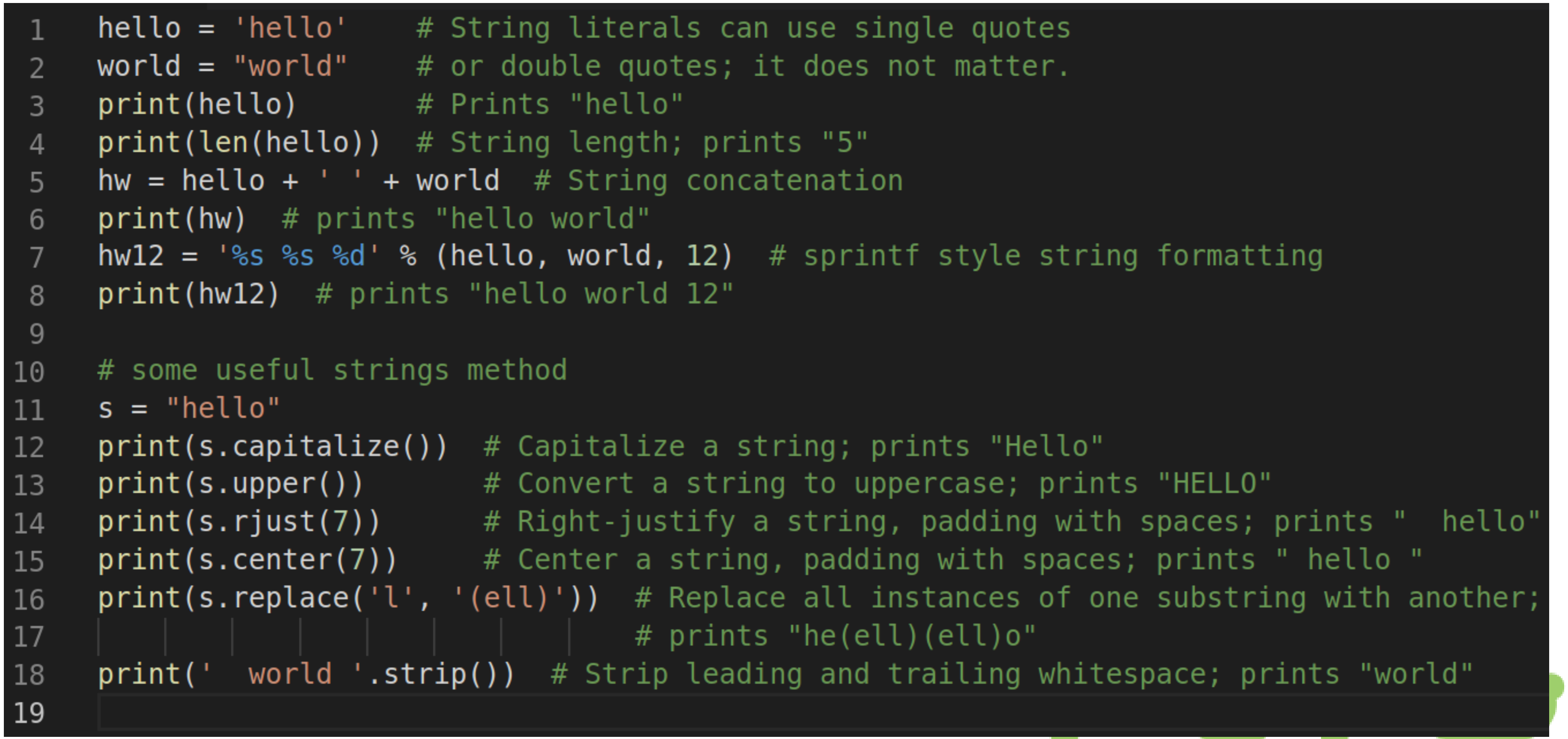


```python
t = True
f = False
print(type(t)) # Prints "<class 'bool'>"
print(t and f) # Logical AND; prints "False"
print(t or f)  # Logical OR; prints "True"
print(not t)   # Logical NOT; prints "False"
print(t != f)  # Logical XOR; prints "True"
```

Python Code

Strings

Python implements all of the usual operators for Boolean logic, but uses English words rather than symbols (&&, ||, etc.):

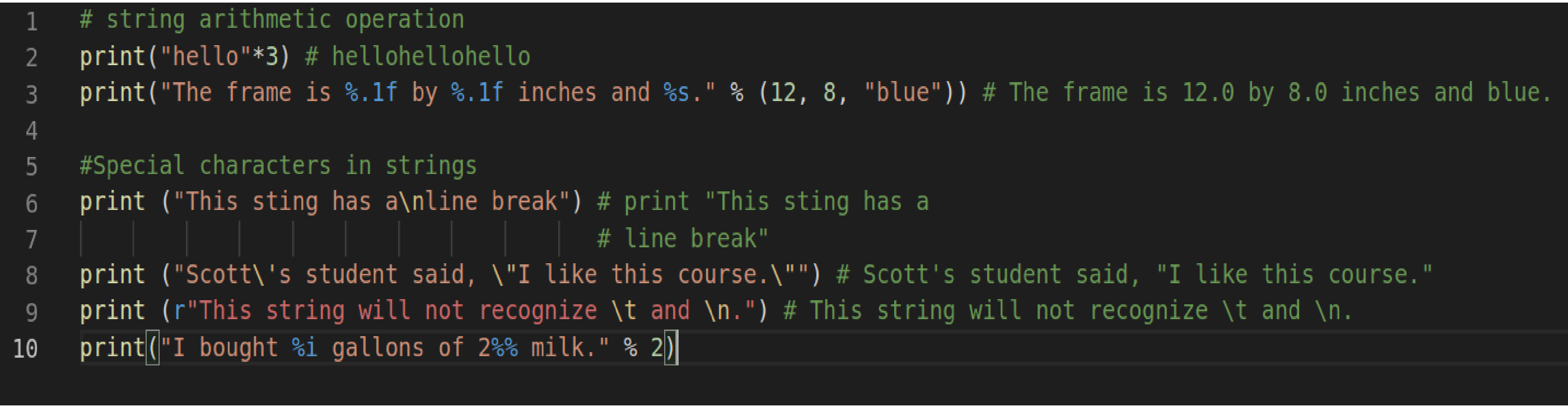


```python
hello = 'hello'    # String literals can use single quotes
world = "world"    # or double quotes; it does not matter.
print(hello)       # Prints "hello"
print(len(hello))  # String length; prints "5"
hw = hello + ' ' + world  # String concatenation
print(hw)  # prints "hello world"
hw12 = '%s %s %d' % (hello, world, 12)  # sprintf style string formatting
print(hw12)  # prints "hello world 12"

# some useful strings method
s = "hello"
print(s.capitalize())  # Capitalize a string; prints "Hello"
print(s.upper())       # Convert a string to uppercase; prints "HELLO"
print(s.rjust(7))      # Right-justify a string, padding with spaces; prints "  hello"
print(s.center(7))     # Center a string, padding with spaces; prints " hello "
print(s.replace('l', '(ell)'))  # Replace all instances of one substring with another;
                                # prints "he(ell)(ell)o"
print('  world '.strip())  # Strip leading and trailing whitespace; prints "world"
```

Python Code

Strings



```python
# string arithmetic operation
print("hello"*3) # hellohellohello
print("The frame is %.1f by %.1f inches and %s." % (12, 8, "blue")) # The frame is 12.0 by 8.0 inches and blue.

#Special characters in strings
print ("This sting has a\nline break") # print "This sting has a
                                       # line break"
print ("Scott\'s student said, \"I like this course.\"") # Scott's student said, "I like this course."
print (r"This string will not recognize \t and \n.") # This string will not recognize \t and \n.
print("I bought %i gallons of 2%% milk." % 2)
```