Introduction to Python

Day 1

Introduction to Python

- Python is a high-level, interpreted programming language.
- It supports procedural, functional, object oriented and modular programming paradigms.
- It uses dynamic type system and automatic memory management.
- Python is expressive.
- Python has rich standard library, vibrant community and lots of easily installed modules that enrich the language.

Hello, World!

Hello, World!

```
print("Hello, World!")
```

Hello, World!

```
print("Hello, World!")
```

Built in functions: abs(), divmod(), input(), open(), staticmethod(), all(), enumerate(), int(), ord(), str(), any(), eval(), isinstance(), pow(), sum(), basestring(), execfile(), issubclass(), print(), super(), bin(), file(), iter(), property(), tuple(), bool(), filter(), len(), range(), type(), bytearray(), float(), list(), raw_input(), unichr(), callable(), format(), locals(), reduce(), unicode(), chr(), frozenset(), long(),reload(), vars(), classmethod(), getattr(), map(), repr(), xrange(), cmp(), globals(), max(), reversed(), zip(), compile(), hasattr(), memoryview(), round(), __import__(), complex(), hash(), min(), set(), delattr(), help(), next(), setattr(), dict(), hex(), object(), slice(), dir(), id(), oct(), sorted()

Hello, World!

```
print('Hello, World!')
```

Hello, World!

```
print('Hello, "World!"')
```

Hello, World!

```
print('Hello, "World!"')
```

Run it!

Hello, World!

```
print("""
          When we are born,
          we cry, that we are come
          to this great stage of fools.
                         - William Shakespeare, King Lear""")
```

quote_generator.py

```
quote = ("When we are born, we cry, that we "
         "are come to this great stage of fools.\n"
         "    - William Shakespeare, King Lear")

print(quote)
```

quote_generator.py

```
quotes = ["When we are born, we cry…",
          "My mistress' eyes are nothing like the sun…",
          "There never was a story of more woe…",
          "And the rain it raineth every day…"]

print(quotes[0])
```

quote_generator.py

```
quotes = ["When we are born, we cry…",
          "My mistress' eyes are nothing like the sun…",
          "There never was a story of more woe…",
          "And the rain it raineth every day…"]

print(quotes[0:2])
```

quote_generator.py

```
quotes = ["When we are born, we cry…",
          "My mistress' eyes are nothing like the sun…",
          "There never was a story of more woe…",
          "And the rain it raineth every day…"]

last_quote = len(quotes) - 1
user_message = “Pick a number from 0 to {0}“.format(last_quote)
s = input(user_message)
n = int(s)
print(quotes[n])
```

quote_generator.py

```
quotes = {"plays": ["When we are born, we cry…",
                    "There never was a story of more woe…",
                    "And the rain it raineth every day…"],
          "sonnets": [“My mistress' eyes are nothing like the sun”,
                      "Two loves I have of comfort and despair"]}

plays = quotes[“plays”]
last_quote = len(quotes) - 1
user_message = “Pick a number from 0 to {0}“.format(last_quote)
s = input(user_message)
n = int(s)
print(plays[n])
```

quote_generator.py

```
def pick_quote(genre):
    quotes = {"plays": ["When we are born, we cry…",
                        "There never was a story of more woe…",
                        "And the rain it raineth every day…"],
               "sonnets": [“My mistress' eyes are nothing like the sun”,
                            "Two loves I have of comfort and despair”]}
    quotes_list = quotes[genre]
    s = input("Pick a number from 0 to {0}“.format(len(quotes_list)))
    n = int(s)
    return quotes[n]

for genres in [“plays”, “sonnets”]:
    print(pick_quote(genre))
```

quote_generator.py

```
import random

def pick_quote(genre):
    quotes = {"plays": ["When we are born, we cry…",
                        "There never was a story of more woe…",
                        "And the rain it raineth every day…"],
              "sonnets": ["Present mirth hath present laughter",
                          "Two loves I have of comfort and despair"]}
    n = random.randint(0, len(quotes[genre]) - 1)
    return quotes[genre][n]

print_quote(“plays”)
print_quote(“sonnets”)
```

quote_generator.py

```
import random

def pick_quote(genre):
    quotes = {"plays": ["When we are born, we cry…",
                        "There never was a story of more woe…",
                        "And the rain it raineth every day…"],
              "sonnets": ["Present mirth hath present laughter",
                          "Two loves I have of comfort and despair”]}
    n = random.randint(0, len(quotes[genre]) - 1)
    return quotes[genre][n]
```

main.py

```
import quote_generator

print(quote_generator.pick_quote(“plays”))
```

quote_generator.py

```
import random

def pick_quote(genre):
    quotes = {"plays": ["When we are born, we cry…",
                        "There never was a story of more woe…",
                        "And the rain it raineth every day…"],
              "sonnets": ["Present mirth hath present laughter",
                          "Two loves I have of comfort and despair”]}
    n = random.randint(0, len(quotes[genre]) - 1)
    return quotes[genre][n]

if __name__ == "__main__":
    print(pick_quote())
```

quote_generator.py

```
import random

def pick_quote(genre):
    quotes = {"plays": ["When we are born, we cry…",
                        "There never was a story of more woe…",
                        "And the rain it raineth every day…"],
              "sonnets": ["Present mirth hath present laughter",
                          "Two loves I have of comfort and despair”]}
    n = random.randint(0, len(quotes[genre]) - 1)
    return quotes[genre][n]

if __name__ == "__main__":
    print(pick_quote())
```

double leading and trailing underscore indicate "magic" objects or attributes that live in user-controlled namespaces.

Reading Command Line Arguments

```
import sys

if len(sys.argv) < 2:
    print('Usage:', sys.argv[0], 'path')
else:
    print('Path:', sys.argv[1])
```

```
import argparse

parser = argparse.ArgumentParser()
parser.add_argument('path')

args = parser.parse_args()
print('Path:', args.path)
```

Random Password Generator

- Write a python file named password_generator.py which contains a function named generate_password(n). The function should return a password of length n (n>2) with the following constrains:
    - The password contains only english lowercase and uppercase letters, digits and the signs !@#$%^&*.
    - The password must contains at least one uppercase letter and at least one digit.
- Add a logic that prints a random password of length 8 when the file is executed and not imported.
- Write another python file main.py which imports password_generator. main.py gets as argument the length of a password (assume it is a number between 2 to 100) and prints a password of that length.
- Hint: Standard library random module contains random.sample(population, k) function. Look for its description in the internet or with Python's interpreter help function.

Searching In Strings

```
import re

s = ‘Little Birds are dining warily and well’

print(s.find(‘Little’))
… 0

print(s.find(‘Big’))
… -1

if re.search(‘are[\sa-z]*and’, s):
    print(True)
… True
```

echo.py

```
import sys

def echo():
    for line in sys.stdin:
        print(line.strip())

if __name__ == "__main__":
    echo()
```

echo.py

```
import sys

def echo():
    for i, line in enumerate(sys.stdin):
        print(i, line.strip())

if __name__ == “__main__”:
    echo()
```

Tic-Tac-Toe