Practice exercises: Basics

The FOSSEE Group

Department of Aerospace Engineering
IIT Bombay

Mumbai, India

Note: Python 2.x and 3.x

If you are using Python 2.x

- Use **raw_input** instead of **input**
- Use the following for **print**

```
from __future__ import print_function
```

Exercise: print string

1. Ask the user to enter a name
   - use **`input()`** (or **`raw_input()`** on 2.x)
2. Lets say the user entered:

   `abc`

   then print

   `hello abc`

Possible solution

```
name = input() # Or raw_input()
print("hello", name)
```

Exercise: input prompt

1. Ask the user to enter a name but give them a prompt:
   `"Please enter your name:  "`
   (note the trailing space)
2. Lets say the user entered:
   `abc`
   then print
   `hello abc`

Possible solution

```
name = input("Please enter your name: ")
print("hello", name)
```

Exercise: integers

1. Ask the user for a single integer (no prompt string)
2. Print the square of this number

Possible solution

```
x_str = input()
x = int(x_str)
print(x*x)
```

Exercise: digits of integer

1. Ask the user for a single integer (use an empty prompt)
2. Square this integer
3. Print the number of digits the squared integer has

Possible solution

```
x_str = input()
x = int(x_str)
y_str = str(x*x)
print(len(y_str))
```

Exercise: complex numbers

1. Ask the user for a single complex number
2. If the number entered was **`1+2j`**, print:

   ```
   1 2
   ```

3. Print the absolute value of this complex number
4. Print the conjugate of this complex number

Possible solution

```python
z_str = input()
z = complex(z_str)
print(z.real, z.imag)
print(abs(z))
print(z.conjugate())
```

Exercise: Booleans

1. Ask the user to enter an integer (use an empty prompt)
2. Print **`True`** if the number is odd, print **`False`** otherwise

Possible solution

```
x_str = input()
x = int(x_str)
print(x%2 == 1)
```

Exercise: Booleans

1. Ask the user to enter an integer (use an empty prompt)
2. Print `True` if the number is even, print `False` otherwise

Possible solution

```
x_str = input()
x = int(x_str)
print(x%2 == 0)
```

Exercise: string operations 1

- Ask the user to enter a name (use an empty prompt)
- Print the name entered but in upper case
- For example, if the user enters ’abc’, print ’ABC’

Possible solution

```python
name = input()
print(name.upper())
```

Exercise: string operations 2

1. Ask the user to enter the name of a file (use an empty prompt)
2. Add an extension '.txt' to the name and print it
3. For example, if the user enters 'abc', print 'abc.txt'

Possible solution

```
name = input()
print(name + ’.txt’)
```

Exercise: string slicing

1. Ask the user to enter a string
2. Print the string without the first character
3. Print the string without the last character
4. Print the string in reverse
5. Finally print every alternate character of the string starting from the first

For example, if the user enters ’abcd’, print the following:

```
bcd
abc
dcba
ac
```

Possible solution

```
name = input()
print(name[1:])
print(name[:-1])
print(name[::-1])
print(name[::2])
```

Exercise: string manipulations

- Ask the user to enter a string
- Count the number of vowels in the string
- The code should be case-insensitive

Possible solution

```
x = input()
x = x.lower()
n = (x.count(’a’) + x.count(’e’) +
     x.count(’i’) + x.count(’o’) +
     x.count(’u’))
print(n)
```

Exercise: string containership

- Ask the user to enter a string
- Check if the substrings ’abc’ OR ’def’ exist in the string
- If it does, print True, else False

Possible solution

```
x = input().lower()
print(’abc’ in x or ’def’ in x)
```

Exercise: digits of an integer

Given a 2 digit integer **`x`**, find the digits of the number.

- For example, let us say **`x = 38`**
- Find a way to get **`a = 3`** and **`b = 8`** using **`x`**
- Print the digits, one in each line

Possible Solution

```
x = int(input())
a = x//10
b = x%10
print(a)
print(b)
```

Another Solution

```
x = input()
print(x[0])
print(x[1])
```