

# Practice exercises: data structures

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: simple list creation

- 1 Ask the user to enter an integer, **n**
- 2 Create a list of integers from 0 to **n-1**
- 3 Print this list

# Solution

```
n = int(input())  
x = list(range(n))  
print(x)
```

# Exercise: more list creation

- 1 Ask the user to enter an integer, **n**
- 2 Store the square of all the odd numbers less than **n** in a list
- 3 Print this list

# Solution

```
n = int(input())
result = []
for i in range(1, n, 2):
    result.append(i*i)
print(result)
```

# Exercise: tuple creation

- 1 Ask the user to enter an integer, **n**
- 2 Store the square of all the odd numbers less than **n**
- 3 Print a tuple of this list

# Solution

```
n = int(input())  
result = []  
for i in range(1, n, 2):  
    result.append(i*i)  
print(tuple(result))
```



# Hint: string to list/tuple

Here is an easy way to convert a string to a list of characters.

Try this:

```
In []: x = 'hello'
```

```
In []: print(list(x))
```

```
In []: print(tuple(x))
```

# Exercise: list of Fibonacci

- 1 Ask the user to enter an integer,  $n$  ( $\geq 1$ )
- 2 Store the first  $n$  numbers of the Fibonacci series in a list
- 3 Print this list

# Solution

```
n = int(input())  
a, b = 0, 1  
result = [0]  
for i in range(n-1):  
    result.append(b)  
    a, b = b, a+b  
print(result)
```

# Exercise: square a list of integers

- 1 Ask the user to enter a list of integers separated by spaces
- 2 Convert this into a list of integers but square each element
- 3 Print this list

For example, if the user enters **1 2 3 4**, print:

**[1, 4, 9, 16]**

# Solution

```
text = input()
result = []
for item in text.split():
    x = int(item)
    result.append(x*x)
print(result)
```

# Exercise: list of tuples

- 1 Ask the user to enter a list of integers separated by spaces
- 2 Convert this into a list of integers but square each element
- 3 Store the integer and its square in a tuple, put this into a list
- 4 Print this list

For example, if the user enters **1 2 3 4**, print:

```
[ (1, 1), (2, 4), (3, 9), (4, 16) ]
```

# Solution

```
text = input()
result = []
for item in text.split():
    x = int(item)
    result.append((x, x*x))
print(result)
```

# Hint: iterating over a list of tuples

Consider the following code:

```
In []: data = [(1, 1), (2, 4), (3, 9), (4, 16)]
```

We can iterate over this as follows:

```
In []: for x, y in data:  
.....:     print(x, y)
```



# Exercise: list methods

- 1 Ask the user to enter a string
- 2 Convert this into a list of characters
- 3 Sort this list in ascending order
- 4 Now eliminate any repeated values in this list
- 5 Print this list

For example, if the user enters **hello**, print:

```
['e', 'h', 'l', 'o']
```

# Solution

```
text = input()
chars = list(text)
chars.sort()
result = []
for c in chars:
    if c not in result:
        result.append(c)
print(result)
```

# Another solution

```
text = input()
chars = set(text)
chars = list(chars)
chars.sort()
print(chars)
```

# Another solution

```
chars = set(input())  
print(sorted(chars))
```

# Exercise: simple dictionaries

- 1 Ask the user for a list of integers separated by spaces
- 2 For each integer, store the string version as the key and the square of the integer value as the value in a dictionary.
- 3 Print the resulting dictionary

For example if the user enters "1 3 5", print:

```
{ '1' : 1, '3' : 9, '5' : 25 }
```

Hint: simply print the resulting dictionary

# Solution

```
text = input()
d = {}
for item in text.split():
    x = int(item)
    d[item] = x*x
print(d)
```

# Exercise: mapping using dicts

- 1 Create a mapping from 3 character month name to month number
- 2 Hint: for example `{ 'jan' : 1, 'dec' : 12 }`
- 3 Ask the user for a 3 character month code lower/upper mixed
- 4 Print the month number corresponding to the month the user entered.

For example if the user enters "Jul", print:

7

# Solution

```
months = ('jan feb mar apr may jun jul '  
          'aug sep oct nov dec').split()  
month2mm = {}  
for i in range(len(months)):  
    month2mm[months[i]] = i + 1  
  
text = input()  
mon = text[:3].lower()  
print(month2mm[mon])
```



## Aside: using `enumerate` example

```
for i, char in enumerate('hello'):  
    print(i, char)
```

## Aside: using enumerate

```
months = ('jan feb mar apr may jun jul '  
          'aug sep oct nov dec').split()  
month2mm = {}  
for i, month in enumerate(months):  
    month2mm[month] = i + 1
```

*# Is easier/nicer than this:*

```
for i in range(len(months)):  
    month2mm[months[i]] = i + 1
```

# Exercise: dictionaries

- 1 Ask the user to enter a string
- 2 Convert this to lower case
- 3 Count the number of occurrences of each character in the string
- 4 Hint: use a dict
- 5 Print the result in sorted order of the characters

For example, if the user enters **helloo**, print:

```
e 1
h 1
l 2
o 2
```

# Solution

```
text = input().lower()
result = {}
for char in text:
    if char in result:
        result[char] += 1
    else:
        result[char] = 1

for char in sorted(result):
    print(char, result[char])
```

# Problem: datestring to date tuple

You are given date strings of the form “29 Jul, 2009”, or “4 January 2008”. In other words a number, a string and another number, with a comma sometimes separating the items.

Write a program that takes such a string as input and prints a tuple (yyyy, mm, dd) where all three elements are ints.

# Solution

```
months = ('jan feb mar apr may jun jul ' +  
          'aug sep oct nov dec').split()  
month2mm = {}  
for i, month in enumerate(months):  
    month2mm[month] = i + 1  
  
date = input()  
date = date.replace(',', ' ')  
day, month, year = date.split()  
  
dd, yyyy = int(day), int(year)  
mon = month[:3].lower()  
mm = month2mm[mon]  
print((yyyy, mm, dd))
```

# That's all folks!