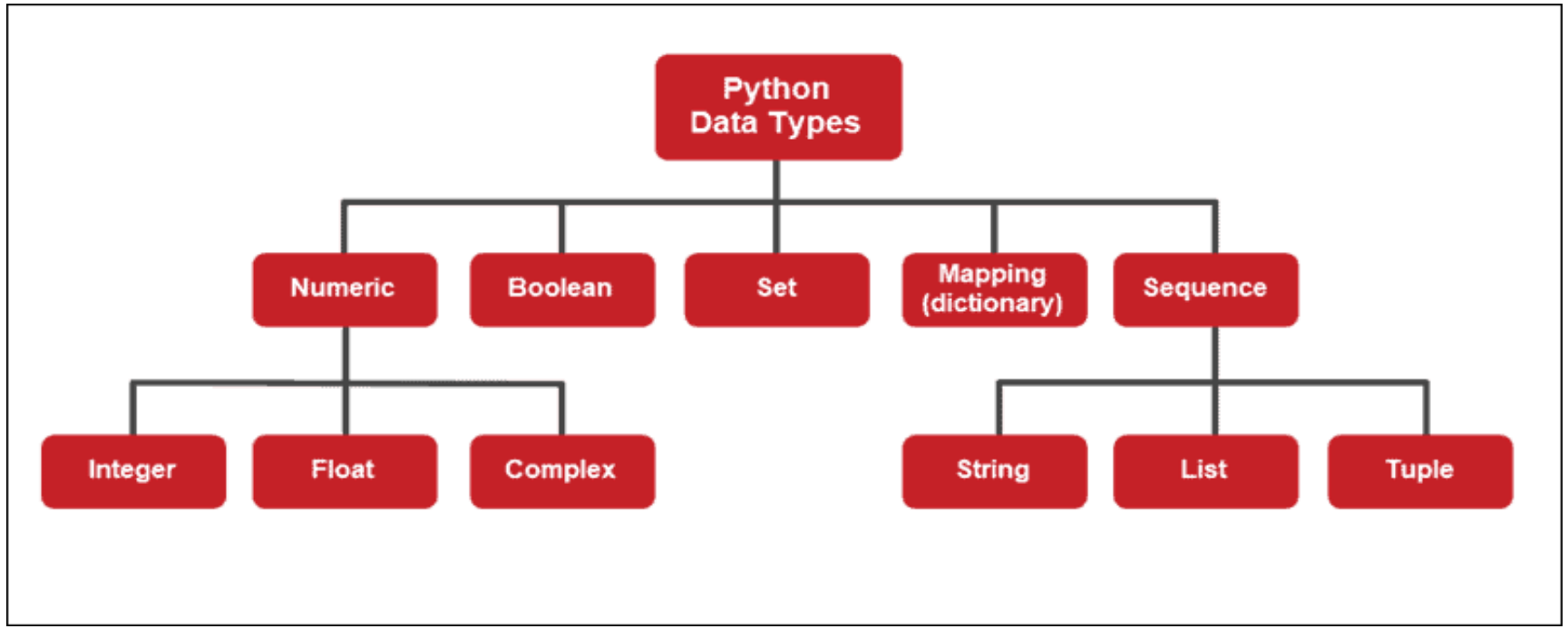
DICTIONARIES in Python

Introduction

- So far we have talked about Numeric and sequence data types, now we will talk about mapping data type



Dictionaries

- The data type *dictionary* fall under mapping. It is a mapping between a *set of keys* and a *set of values*.

- The key-value pair is called an *item*. A key is separated from its value by a colon(:) and consecutive items are separated by commas. For e.g.)

```
dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
```

Note:

Items in dictionaries are unordered, so we may not get back the data in the same order in which we had entered the data initially in the dictionary.

Creating a Dictionary

- To create a dictionary, the items entered are separated by commas and enclosed in curly braces. Each item is a key value pair, separated through colon (:)

  For e.g.)

  ```
  dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
  ```

- The keys in the dictionary must be unique and should be of any immutable data type, i.e., number, string or tuple. However, the values can be repeated and can be of any data type.

Creating an empty Dictionary

```
1  #creating an empty dictionary
2  #using curly braces
3  dict1 = {}
4  print(dict1)
5
```

```
{}


...Program finished with exit code 0
Press ENTER to exit console.
```

```
1  #creating an empty dictionary
2  #using built-in function
3  dict1 = dict()
4  print(dict1)
5
```

```
{}


...Program finished with exit code 0
Press ENTER to exit console.
```

Accessing items in a dictionary

- The items of a dictionary are accessed via the keys rather than via their relative positions or indices. Each key serves as the index and maps to a value.

  for e.g.)

```
In [4]: dict1 = {'Mohan':95,'Ram':89,'Suhel':92,'Sangeeta':85}

In [5]: dict1['Sangeeta']
Out[5]: 85
```

- If we try to access the key which is not present in the dictionary, then we will get the key error:

  for e.g.)

```
In [6]: dict1['Shyam']
Traceback (most recent call last):

  File "<ipython-input-6-0580745d34c2>", line 1, in <module>
    dict1['Shyam']

KeyError: 'Shyam'
```

Dictionaries are mutable

- Dictionaries are mutable which implies that the contents of the dictionary can be changed after it has been created.
- Items in a dictionary can be added, or modified with the help of an assignment operator

Adding an item in dictionary

- In order to add a new item in dictionary, we need to use the following syntax:
  dictionary_name[key] = value

  where key and value follow the general concept of dictionary

for e.g.)

```
In [14]: dict1 = {"Ram":89, 78:"Shyam", 23:76}

In [15]: dict1["Rohit"] = "Kumar"

In [16]: dict1
Out[16]: {'Ram': 89, 78: 'Shyam', 23: 76, 'Rohit': 'Kumar'}

In [17]: dict1["Himanshu"] = [1,2]

In [18]: dict1
Out[18]: {'Ram': 89, 78: 'Shyam', 23: 76, 'Rohit': 'Kumar', 'Himanshu': [1, 2]}
```

Adding an item in dictionary

- If we try to add an item with key as mutable data type, then we will get an error. For e.g.)

```
In [13]: dict1[[1]] = "Ram"
Traceback (most recent call last):

  File "<ipython-input-13-bd2b13554f70>", line 1, in <module>
    dict1[[1]] = "Ram"

TypeError: unhashable type: 'list'
```

Modifying an existing item in dictionary

- Any item in the existing dictionary can be modified by just overwriting the key-value pair of that item

  for e.g.)

```
In [19]: dict1 = {"Ram":89, 78:"Shyam", 23:76}

In [20]: dict1["Ram"] = "Sita"

In [21]: dict1
Out[21]: {'Ram': 'Sita', 78: 'Shyam', 23: 76}
```

Operations on Dictionary: Membership

- The membership operator in checks if the key is present in the dictionary and returns True, else it returns False.

  For e.g.)

```
In [1]: dict1 = {1:"Ram", 2:"Hi", 3: True, 4: False}

In [2]: 1 in dict1
Out[2]: True

In [3]: 6 in dict1
Out[3]: False
```

Operations on Dictionary: Membership

- The not in operator returns True if the key is not present in the dictionary, else it returns False.

  For e.g.)

```
In [9]: dict2 = {"Ram": 12, "Hi": True, 2:"Lokesh"}

In [10]: 3 not in dict2
Out[10]: True

In [11]: "Ram" not in dict2
Out[11]: False
```

Traversing a Dictionary

- We can access each item of the dictionary or traverse a dictionary using for loop.

```
1  '''
2  Sample Program: Traversing a dictionary using for loop
3  @author: Himanshu
4  '''
5  dict1 = {1: "Himanshu", 2: "Ram", 3: "Suresh", 4: 3.4}
6  #accessing each item with the help of key
7  for i in dict1:
8      print(i,":",dict1[i])
9
```

input

```
1 : Himanshu
2 : Ram
3 : Suresh
4 : 3.4


...Program finished with exit code 0
Press ENTER to exit console.
```

Dictionary methods and built-in functions

- built-in functions:

  len(), dict(), keys(), values(), items(), get(), update(), del(), clear(), fromkeys(), copy(), pop(), popitem(), setdefault(), max(), min(), count(), sorted(), copy();

Method	Description	Example

len()	Returns the length or number of key: value pairs of the dictionary passed as the argument	>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> len(dict1)
		4

dict()	Creates a dictionary from a sequence of key-value pairs	pair1 = [('Mohan',95),('Ram',89), ('Suhel',92),('Sangeeta',85)]
		>>> pair1
		[('Mohan', 95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)]
		>>> dict1 = dict(pair1)
		>>> dict1
		{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85}

keys()	Returns a list of keys in the dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.keys()
		dict_keys(['Mohan', 'Ram', 'Suhel', 'Sangeeta'])

values()	Returns a list of values in the dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.values()
		dict_values([95, 89, 92, 85])

items()	Returns a list of tuples(key – value) pair	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.items()
		dict_items([('Mohan', 95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)])

get()	Returns the value corresponding to the key passed as the argument
	
	If the key is not present in the dictionary it will return None
		>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.get('Sangeeta')
		85
		>>> dict1.get('Sohan')
		>>>

update()	appends the key-value pair of the dictionary passed as the argument to the key-value pair of the given dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict2 = {'Sohan':79,'Geeta':89}
		>>> dict1.update(dict2)
		>>> dict1
		{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85, 'Sohan': 79, 'Geeta': 89}
		>>> dict2
		{'Sohan': 79, 'Geeta': 89}

del()	Deletes the item with the given key
	To delete the dictionary from the memory we write:
	del Dict_name
		>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> del dict1['Ram']
		>>> dict1
		{'Mohan':95,'Suhel':92, 'Sangeeta': 85}
		
		>>> del dict1 ['Mohan']
		>>> dict1
		{'Suhel': 92, 'Sangeeta': 85}
		>>> del dict1
		>>> dict1
		NameError: name 'dict1' is not defined

clear()	Deletes or clear all the items of the dictionary	>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.clear()
		>>> dict1
		{ }

DICTIONARIES in Python

- Introduction to Dictionary
- Creating a dictionary
- Accessing items in a dictionary
- Dictionaries are mutable
    - adding an item in dictionary
    - modifying an existing item in dictionary
- Dictionary Operations: Membership
- Traversing a dictionary
- Dictionary methods and built-in functions

Practice problems on Dictionary

- Sample program: Write a program to enter names of employees and their salaries as input and store them in a dictionary

Code:

```
'''
Sample program: Write a program to enter names of employees and their
                salaries as input and store them in a dictionary
@author: Himanshu
'''
n = int(input("Enter the no. of employess for which you want to enter the data: "))
#creating an empty dictionary
emp = dict()
for i in range(n):
    name = input("Enter the name of employee: ")
    sal = int(input("Enter its salary: "))
    emp[name]=sal
print(emp)

```

Output:

```
Enter the no. of employess for which you want to enter the data: 5
Enter the name of employee: HImanshu
Enter its salary: 10000
Enter the name of employee: Rohit
Enter its salary: 12000
Enter the name of employee: Vaibhav
Enter its salary: 13000
Enter the name of employee: Naman
Enter its salary: 11000
Enter the name of employee: akash
Enter its salary: 8000
{'HImanshu': 10000, 'Rohit': 12000, 'Vaibhav': 13000, 'Naman': 11000, 'akash': 8000}
```

Practice problems on Dictionary

- Sample program: Write a program to count the number of times a character appears in a given string.

Code:

```
'''
Sample program: Write a program to count the number of times
                a character appears in a given string.
@author: Himanshu
'''
#input a string
string = input("Enter a string: ")
#create an empty dictionary
dict1 = dict()
for ch in string:
    if ch in dict1:
        dict1[ch] += 1
    else:
        dict1[ch] = 1
#printing dictionary
for i in dict1:
    print(i,":", dict1[i])
```

Output:

```
Enter a string: Ram Narayan
R : 1
a : 4
m : 1
  : 1
N : 1
r : 1
y : 1
n : 1
```

Summary

- Introduction to Dictionary
- Creating a dictionary
- Accessing items in a dictionary
- Dictionaries are mutable
  - adding an item in dictionary
  - modifying an existing item in dictionary
- Dictionary Operations: Membership
- Traversing a dictionary

Summary

- Dictionary methods and built-in functions
- Practice problems on Dictionaries

Assignment - 6

- Note down the summary points in your notebook (page 223)
- note the down the built-in functions for Dictionary in Python
- Q2 and Q6 of chapter-10

Programming Assignment - 6

- Write a program to convert a number entered by the user into its corresponding number in words. For example, if the input is 8126 then the output should be 'Eight One Two Six'.
- Write a Python program to find the highest 2 values in a dictionary.
- Create a dictionary with the roll number, name and marks of n students in a class and display the names of students who have scored marks above 75.