PYTHON STRINGS, TUPLES, LISTS, DICTIONARIES AND ARRAY

FAKULTI SAINS SOSIAL & KEMANUSIAAN

SEKOLAH PENDIDIKAN

2020

Learning Outcomes

At the end of this lesson, students should learn on:

- Strings
- Tuples
- Lists
- Dictionaries

PYTHON STRINGS

Strings

A string consists of text characters:

- just one character
- a few lines
- a whole text file

Inside Python:

- A string literal is surrounded by single/double/triple quotation marks, which distinguish it from other kinds of data, such as integers or names.
- But when you print a string, the quotation marks don't appear.

Strings

Identify the **differences** and the **escape sequences**.

```
welcome = "Welcome to Sentosa’s Resort and Spa. "
enjoy = "Enjoy your stay, \"Hanin\"!"

print (welcome)
print (enjoy)
print (welcome + enjoy)
```

```
print ('Welcome to Sentosa\'s Resort and Spa. ')
print ('Enjoy your stay, "Hanin"!')
print ('Welcome to Sentosa\'s Resort and Spa. \n Enjoy your stay, "Hanin"!')
'
```

```
welcome = """Welcome to Sentosa’s Resort and Spa.
Have a nice day.
Enjoy your stay, "Hanin"!"""

print (welcome)
```

Working with Strings

1. Concatenation

```
ayat1 = "hello "
ayat2 = "world"
ayat3 = ayat1 + ayat2
print (ayat3)
```

Continuous exercise from Subtopic 1: Concatenation

2. Repetition

```
ayat4 = ayat1*3 + ayat2
print (ayat4)
```

Working with Strings

3. Indexing

```
ayat5 = ayat1 [1]
ayat6 = ayat1 [-2]
print (ayat5)
print (ayat6)
```

Continuous exercise from Subtopic 1: Concatenation

4. Slicing and Reversing

```
ayat7 = ayat2 [0:3] + "ding"
print (ayat7)
print (ayat7 [ : : -1])
```

Working with Strings

Continuous exercise from Subtopic 1: Concatenation

5. Measuring

```
ayat8 = "Saya suka belajar programming
menggunakan Python"
length_ayat8 = len (ayat8)
print ("Panjang ayat8 = ", length_ayat8)
```

6. Splitting

```
split_ayat8 = ayat8.split ()
print ("Pecahan ayat8 = ", split_ayat8)
```

Working with Strings

7. Comparison

```
a = "utm"
b = "utm"
c = "UTM"
d = "uTM"

print ("a = b : ", a==b)
print ("a = c : ", a==c)
print ("c = d : ", c!=d)
```

Continuous exercise from Subtopic 1: Concatenation

PYTHON TUPLES

Tuples

- A tuple is a sequence data type that can contain elements of different data types.
- Tuples are different from lists because tuples are immutable/cannot be changed.
- If you want to change the contents of a tuple, you must create a new tuple that has the new content you want.

Working with Tuples

1. A tuple literal is defined in one of three ways:
   - A single element followed by a comma
   - Multiple elements separated by commas
   - An empty set of parentheses

```
tuple1 = "hello world"
tuple2 = ("hello world")
tuple3 = ("hello", "world")
tuple4 = "hello", "world"
tuple5 = ()

print (tuple1)
print (tuple2)
print (tuple3)
print (tuple4)
print (tuple5 + tuple3)
```

Working with Tuples

2. Operator on a single-element tuple and on an integer.

```
tuple_1 = (3,)
print (tuple_1 * 3)

tuple_2 = (3)
print (tuple_2 * 3)
```

3. Indexing, Slicing and Measuring

```
negeri = ("johor", "melaka", "kedah", "selangor")
print (negeri)
print (negeri [2])
print (negeri [0:2])
print (len (negeri))
```

Lists VS Tuples

Guido (Python's creator) and the Python community promote the following conventions for choosing between lists and tuples:

Heterogeneous and homogeneous data

- Tuples for heterogeneous data, lists for homogeneous data.
- Use a tuple if your data includes several different data types, such as names and addresses.
- Use lists for elements that are all of the same type.

Tuples for sequence keys

- If you need to use a sequence as a dictionary key, you must use a tuple because dictionary keys are immutable.

Tuples for some functions

- Some functions require arguments to be passed in tuples.

Lists for mutable objects

- Avoid using mutable objects in immutable containers because errors and unexpected results can occur if you attempt to change the mutable objects.

PYTHON LISTS

Lists

- A list is a mutable data type, which means you can change the contents of a list without creating a new list.
- The elements of a list can be of different data types.
- A single list can contain numbers, strings, other lists, tuples, functions and classes.

- In Python:
  - a list literal (the actual data, not a name referring to the data) is defined by square brackets surrounding zero or more elements.
  - Elements are separated by commas.

Working with Lists

1. Creating, Sorting and Reversing

```
my_list = [ ]
listA = [5, 2, 1, 3, 4]                    # list of integers
listB = [2,"azrena",3.413]                 # list of mixed datatypes
listC = [1,[1,2,3],3,4,5]                   # nested list
listD = ["farah", "zainab", "hamid", "ahmad"]   # list of strings
mylist = listA + listD
print (mylist)
print (listC + [listA])

listA.sort ()
listD.reverse ()
print (listA)
print (listD)
print (listD [::-1])
```

Working with Lists

2. Indexing, Slicing and Measuring

```
list_a = ["universiti", "teknologi", "malaysia", "johor", "bahru"]
list_b = [1, 2, ["apple", "fish", "meat"], "arnab", "kucing"]

print (list_a [2])
print (list_a [1:4])
print (len (list_a))

print (list_b [2][1])

input("\nPress enter to exit")
```

Working with Lists

3. Adding, Removing and Inserting

```
import time
list1 = ["universiti", "teknologi", "malaysia", "johor"]
list1.append ("malaysia")
print (list1)

list1.pop ()
print (list1)
list1.pop (-1)
list1.remove ("universiti")
print (list1)

list1.insert (3, "UTM")
print (list1)
time.sleep (5)
```

PYTHON DICTIONARIES

Dictionaries

- Dictionaries (dicts) are a container type that stores data, like lists and tuples.
- Dictionaries are a mapping data type – indexed by keys.
- A dictionary's elements (the items it contains) are **key:value** pairs.
- In contrast, lists and tuples store data by using numeric indexes and support indexing and slicing.
- Dictionaries are mutable.
- However, a dict's keys must be immutable.

Working with Dictionaries

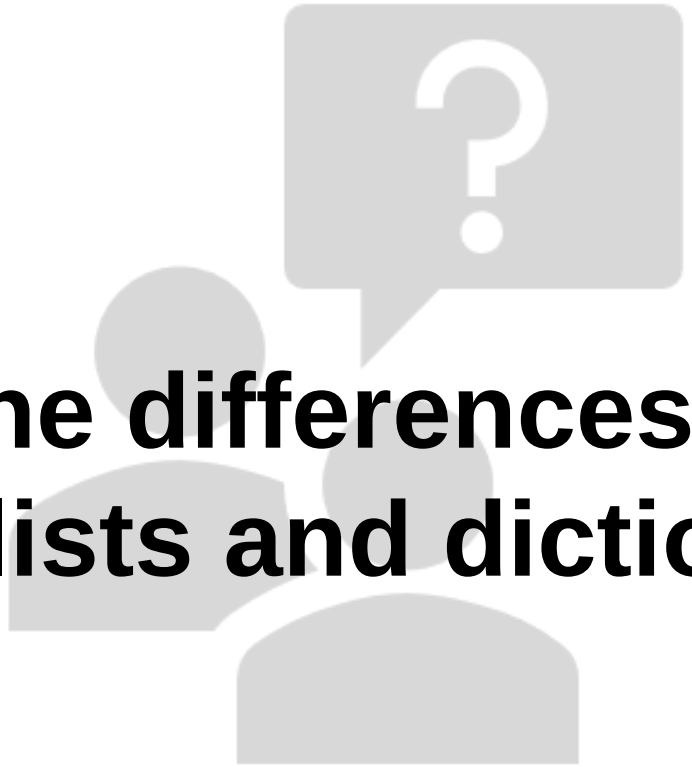
```
kamus_buah = {}
kamus_negeri = {"johor":"johor bahru", "pahang":"kuantan"}
kamus_jantina = {1:"lelaki" , 2:"perempuan"}
print ("Ibu negeri johor adalah ", kamus_negeri ["johor"])

kamus_negeri ["kelantan"] = "kota bharu"
print (kamus_negeri)
del kamus_negeri ["pahang"]
print (kamus_negeri)
print (kamus_jantina [1])
kamus_buah ["durian"] = "RM25 sekilo"
print (kamus_buah)

print (kamus_negeri.keys())
print (kamus_negeri.values())
print (kamus_negeri.items())
```

PYTHON ARRAY

Array

- Arrays are used to store multiple values in one single variable.
- Python does not have built-in support for Arrays, but Python Lists can be used instead.
- Python has a set of built-in methods that you can use on lists/arrays.

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list



Identify the differences between tuples, lists and dictionaries.

That’s all ☺