Python and Colab Introduction

Akitaka Matsuo
Department of Government

Motivation

- We run some basic codes using Colab
- **Menu**
  - Create a Jupyter Notebook on Colab
  - Python basics
  - Object types
  - Basic manipulation

Start Using Colab

- Let’s try using Colab
  - Open new notebook
  - Do some calculation
  - Save into Google Drive

University of Essex

Python objects

- Variables
- Lists
- Dictionaries
- (Tuples, sets)

Variables

- Float
  - `a = 2.0`
- Integer
  - `b = int(3)`
- String
  - `some_string = 'this is a string'`

Colab

- Generate some scaler objects
  - Numeric
    - Float
    - Integer
  - String

Lists

- Collection of variables (e.g. vector)
- No fixed type for elements (can be mixed)
- Constructor: []
  - `an_list = [1, 3, 4]`
  - `[None, "hello", 3, 5]`
- Index from 0
  - `an_list[0:2]`
  - `an_list[3:]`
- New element:
  - `an_list.append(5)`
  - `an_list[3] = 10`

Colab

- Generate some list
- Accessing list elements
- Slicing

Dictionaries

- Collection of key value pairs
- Constructor: `{}`
  - `dic = {"a": 1, "b": 3, "hello": 4}`
- Accessing elements with key
  - dic['a']
- Value assignment:
  - `dic['b'] = 5`
  - `dic['x'] = 7`

Tuples, sets

- Tuples
  - Unchangeable lists
- Sets
  - List of non-duplicated elements, also unchangeable

Colab

- Dictionary
  - Create dictionaries
  - Accessing elements
- Sets and tuple

Control Flow

- `for`-loop
- `while`-loop
- `if-elif-else`-sequence

- Python uses indentation for indicating the control flow, as well as functions
  - Loop keeps going until the indent ends
  - Nested loop uses higher indentation

`for` loop

- Basic syntax:

```
for i in ...:
  print(i)
  j = i + 2
  print(j)
```

- `...` is the place to put the object to loop over
  - List
  - Dictionary
  - Tuple
  - Iterator: `range()`

`while` **loop**

- Basic syntax:

```
i = 0;
while i < 6:
  print(i)
  i += 1 # this is the same as i = i + 1
```

if-elif-else

- Run a command based on the evaluation results of a conditional statement
  - Example conditions:
    - `==`
    - `>`
    - `<`
    - `and, or`

- Basic syntax:

```
if x > 3:
  print("x is larger than 3")
elif x > 1:
  print("x is between 1 and three")
else:
  print("x is smaller than 1")
```

University of Essex

Functions

- An example
  - ```
    def my_function(x):
        x = x + 2
        return(x)
    ```
- Rules
  - First line:
    `def function_name(arguments):`
  - Function continues until indentation ends
  - You can decide the default value for arguments, but such arguments have to come after non-default arguments
  - `return` has to be explicit (otherwise `None`)

University of Essex
    ```

List comprehension

- Very python like way of creating new list from a list
- Basic syntax:
  - `newlist = [expression for item in iterable if condition == True]`
- For example: to subset a list by selecting element > 5

```
oldlist = [1, 5, 10, 8, 9]
newlist = [ item for item in oldlist if item > 5]
```

University of Essex

Importing packages

- In order to use packages you need to import (unlike Stata)
- Syntax
  - `import ***`
  - `import *** as xxx`
  - `from *** import xxx, yyy`

University of Essex