Functions 2 and File Handling

Distributed Network and Systems Laboratory,
Chonnam National University
Presenter : Shivani Kolekar

Recap

- ✓ **Introduction to Functions**
- ✓ **Types of Functions**
- ✓ **Defining and calling Functions**
- ✓ **Local Variables**
- ✓ **Passing Arguments**
- ✓ **Global Variables**

Functions Example

Problem to solve:

Lets consider that John is hungry and wants to make himself a smoothie and a sandwich. Write a program to give John recipe instructions to make a smoothie and a sandwich.

Functions needed for making breakfast:

1) Function for instructions to make a smoothie .
2) Function for instructions to make a sandwich.
3) Function for user input, whether he wants recipe for smoothie or sandwich.

```
def function_name():
    statement
    statement
```

Defining and Calling a function

- ✓ **Functions are given names.**
  - ❑ **Function naming rules:**
    1. Cannot use key words as a function name
    2. Cannot contain spaces
    3. First character must be a letter or underscore
    4. All other characters must be a letter, number or underscore
    5. Uppercase and lowercase characters are distinct

- ✓ **In Python, each line in a block must be indented.**

```
def function_name():
    statement
    statement
```

Passing String Arguments

Argument :

- ✓ Piece of data that is sent into a function.
- ✓ Function can use argument in calculations.
- ✓ When calling the function, the argument is placed in parentheses following the function name.


```
def main():
  a = 6
  b = 4
  numbers(a,b)

def numbers(n1,n2):
  print(n1)
  print(n2)

main()
```

Output

```
6
4
```

Lets Code!

Today's Goals

- ✓ **Practice of what we learned last week**
- ✓ **Creating files with Python**
- ✓ **File related methods (read,write,append)**
- ✓ **Q & A**

Python File Handling

- ✓ File handling is an important part of any web application.
- ✓ Python has several functions for creating, reading, updating, and deleting files.

Python File Handling: open() function

✓ The key function for working with files in Python is the open() function.

✓ The open() function takes two parameters; filename, and mode.

✓ General Format:

file_variable = open(filename, mode)

- file_variable →name of the variable that will reference the file.
- filename → string specifying the name of the file.
- mode → string specifying the mode (reading, writing, etc.)

Python File Handling: open() function

- Some of the Python File modes:

Mode	Description
'r'	Open a file for reading only. The file cannot be changed or written to.
'w'	Open a file for writing. If the file already exists, erase its contents. If it does not exist, create it.
'a'	Open a file to be written to. All data written to the file will be appended to its end. If the file does not exist, create it.

✓ <u>General Format:</u>

file_variable = open(filename, mode)

✓ <u>Example:</u>

new_file = open(my_first_file.txt, 'r')

Python File Handling: close() function

- Once a program is finished working with a file, it should close the file.
- Closing a file disconnects the program from the file.
- In some systems, failure to close an output file can cause a loss of data.

- This happens because the data that is written to a file is first written to a buffer, which is a small "holding section" in memory.

- **General Format:**

  new_file.close()

Python File Handling: write() method

- Writing in the file using write() method.

astronauts.py

```python
# This program writes three lines of data to a file.
def main():
    # Open a file named astronauts.txt.
    myfile = open('astronauts.txt', 'w')

    # Write the names of three astronauts to the file.
    myfile.write('Kalpana Chawla\n')
    myfile.write('Sunita Williams\n')
    myfile.write('Rakesh Sharma\n')

   # Close the file.
    myfile.close()
   # Call the main function.
main()
```

Output File

astronauts.txt

```
1 Kalpana Chawla
2 Sunita Williams
3 Rakesh Sharma
4
```

Python File Handling: readline() function

- A line is simply a string of characters that are terminated with a \n.
- The method returns the line as a string, including the \n.

Read_line.py

```python
# The output of readline() method
# returns the line as a string, including the \n.

def main():
  infile = open('astronauts.txt','r')
  line1 = infile.readline()
  line2 = infile.readline()
  line3 = infile.readline()
  infile.close()
  print(line1)
  print(line2)
  print(line3)

main()
```

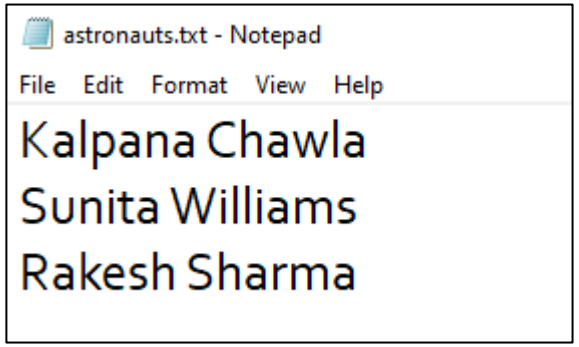
Output

```
Kalpana Chawla

Sunita Williams

Rakesh Sharma
```

File looks like:



Concatenating a newline to a string

- When a program writes data that has been entered by the user to a file, it is usually necessary to concatenate a \n escape sequence to the data before writing it.

- This ensures that each piece of data is written to a separate line in the file.

Concatenating_newline.py

```python
def main():
    # Get three names.
    print('Enter the names of three friends.')
    name1 = input('Friend #1: ')
    name2 = input('Friend #2: ')
    name3 = input('Friend #3: ')
    # Open a file named friends.txt.
    myfile = open('friends.txt', 'w')
    # Write the names to the file.
    myfile.write(name1 + '\n')
    myfile.write(name2 + '\n')
    myfile.write(name3 + '\n')
    # Close the file.
    myfile.close()
    print('The names were written to friends.txt.')
    # Call the main function.
main()
```

```
Enter the names of three friends.
Friend #1: Ram
Friend #2: John
Friend #3: Jungkook
The names were written to friends.txt.
```

Entering user input

friends.txt

```
1 Ram
2 John
3 Jungkook
4
```

Output_file

Lets Code!

Stripping the newline from the string: rstrip() method

Stripping_newline.py

```
# The output of readline() method
# returns the line as a string, including the \n.

def main():
  infile = open('astronauts.txt','r')
  # Read three lines from the file.
  line1 = infile.readline()
  line2 = infile.readline()
  line3 = infile.readline()
  print(line1)
  print(line2)
  print(line3)

  # Strip the \n from each string.
  line1 = line1.rstrip('\n')
  line2 = line2.rstrip('\n')
  line3 = line3.rstrip('\n')
  infile.close()
  print(line1)
  print(line2)
  print(line3)

main()
```

- The \n → separates the items that are stored in the file.
- However, in many cases, you want to remove the \n from a string after it is read from a file.

Output

```
Kalpana Chawla

Sunita Williams

Rakesh Sharma

Kalpana Chawla
Sunita Williams
Rakesh Sharma
```

Appending new data to an existing file

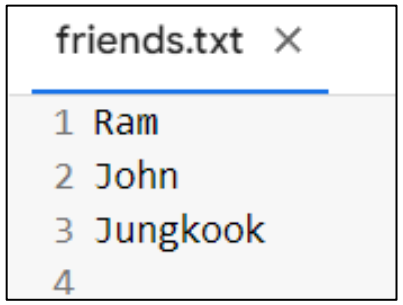
- Appending data to a file means writing new data to the end of the data that already exists in the file.

- In Python, you can use the 'a' mode to open an output file in append mode, which means the following:
  1. If the file already exists, it will not be erased. If the file does not exist, it will be created.
  2. When data is written to the file, it will be written at the end of the file's current contents.

Appending new data to an existing file

- ✓ For example, assume the file friends.txt contains the following names, each in a separate line:

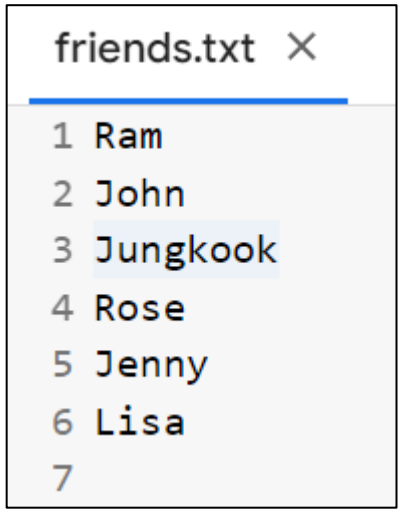
Ram
John
Jungkook



Existing file before appending new data

- ✓ The following code opens the file and appends additional data to its existing contents.

```
myfile = open('friends.txt', 'a')
myfile.write('Rose\n')
myfile.write('Jenny\n')
myfile.write('Lisa\n')
myfile.close()
```



Existing file after appending new data

Python File Handling: remove() function

We need to import os for this operation:

```
import os
os.remove("demofile.txt")
```

```
This program will remove the existing file - demofile.txt
```