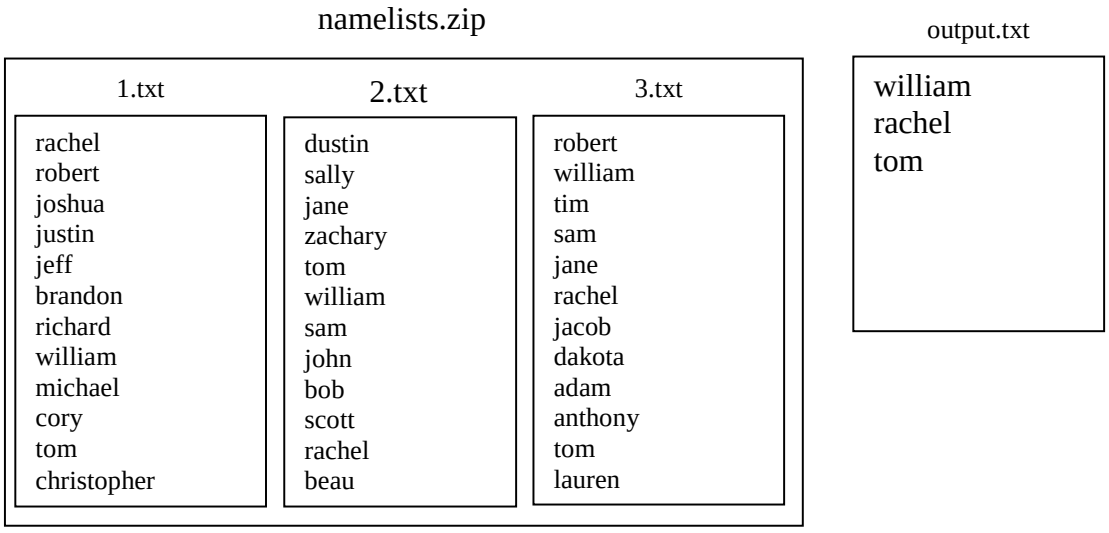
CMPS 401
Survey of Programming Languages

Programming Assignment #6
Python Language
On the Ubuntu Operating System

Write a Python program (**P6.py**) under Ubuntu operating system that extracts a zip archive of text documents and read their contents. Each text document contains a list of names. It displays the results on the **screen** and also writes a new file (**output.txt**), which contains a list of names that appear in **every** document.

1. An example zip (**namelists.zip**) archive is provided along with this assignment, as well as example outputs.



Sample Run:

```
$ python3 P6.py namelists.zip
william
rachel
tom
```

2. You need to write **Python 3** code (not Python 2). Note: Python 2 has different syntax.
3. Your Python script accepts a zip file as a command line argument (ie. $ python3 P6.py namelists.zip). And, you don't know the names of the text documents in the zip file. The zip archive (**namelists.zip**) is just an example. Your script must work with a zip with **any** number of text files with **any** number of names in each. Order does **not** matter in your program's output.

4. You need a CMPS401 account on the Ubuntu Operating System.
5. If you need help with this assignment, go to the following links:
   - Python_Programming_Language (wikipedia.org)
   - The Python Tutorial (python.org)
   - Learn Python The Hard Way: Book (learnpythonthehardway.org)
   - Python Programming (book by Wikibooks.org)
6. Python is one of pure interpretation languages; your program is not compiled before run.
7. Useful UNIX commands are on page 3.
8. Download the zip "namelists.zip" into your folder you will have your program
9. There are 7 Python programming examples to assist you on this assignment.
   1) A simple program to run (TSimple.py)
   2) Test data types and variables (TVar.py)
   3) Test selection statements (TSel.py)
   4) Test loops (TLoop.py)
   5) Test subprograms (TSub.py)
   6) Other concerns: Test File I/O (TFile.py, TFileIn.dat, and TFileOut.dat)
10. Your program assignment #6 consists of the following two files under **"home directory**" of your CMPS401 account:
    1) Python program (P6.py)
    2) Text file (output.txt)

    Note: Your files on the Ubuntu Operating System will be checked and should not be modified after due date.

Useful UNIX Commands

Command	Description
man	help menu
pico	simple text editor
gcc	compiles your source code “gnu C compiler”
a.out	executes your program
ls –al	displays a long list of files “includes hidden files i.e. dot files”
pwd	prints working directory “pathname”
cd	changes directory
mkdir	creates a directory
rmdir	removes a directory
cp file1 file2	copies contents of file1 into file2
mv file1 file2	moves a file from one place to another, or change its name
rm	removes a file
more	displays a file’s contents
grep	searches for a specified pattern in a file or list of files
ps	obtains the status of the active processes in the system
kill –9 pid	terminates all processes
passwd	modify a user's password
logout	terminates your session
who	display who is on the system
finger	displays the user information
date > myfile	“output redirection” saves the output of date command in myfile
cal >> myfile	**“**appends” calendar to myfile
cal	display a calendar and the date
wc file1	counts the number of lines, words, and characters in file1

Examples

A Simple Program to Run (TSimple.py)

```
# Display "Hello" on your screen
# Program-ID:   TSimple.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TSimple.py
#    $ python3  TSimple.py

print ("Hello")
print ('Hello')

''' Output
    Hello
    Hello
'''
```

Data Types and Variables (TVar.py)

```
# Test variables: No declaration
# Note: Python is a "Loosely Typed Language"
#       In Python a variable does not need to be
#       declared before being set.
# Program-ID:   TVar.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TVar.py
#    $ python3  TVar.py

i1 = 1
i2 = 2
f1 = 3.3
f2 = 4.4
c = 'a'
s = "bcd"

f1 = i1  # implicit casting
i2 = f2  # no type checking
c = c + " " + s + " efg" # Concatenation Operator (+)
s = len(s)

print("i1 = ", i1)
print("i2 = ", i2)
print("f1 = ", f1)
print("f2 = ", f2)
print("c  = ",  c)
print("s  = ",  s)

''' Output
    i1 =  1
    i2 =  4.4
    f1 =  1
    f2 =  4.4
    c  =  a bcd efg
    s  =  3
'''
```

Selection Statements (TSel.py)

```
# Test Selections:      if, if-else, nested if-else
# Logical Operators:    and, or, not (invalid: &&, ||, !)
# Relational Operators: <, >, ==, <=, >=, !=
# Program-ID:   TSel.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TSel.py
#    $ python3  TSel.py

i1=1
i2=2
i3=3
i4=4
i5=5
i6=6

#Test a simple if
if i4 > i1:
    print("i4 >  i1")

# Test if-else
if (i5 < i2) and (i3 >= i2):
    print("(i5 <  i2) and (i3 >= i2)")
else:
    print("(i5 >= i2) or (i3 <  i2)")

# Test nested if-else
if i1 != i2:
    print("(i1 != i2)")
elif (i4 == i5) or (i5 != i6):
    print("(i1 == i2) and ((i4 == i5) or (i5 != i6))")

''' Output
    i4 >  i1
    (i5 >= i2) or (i3 <  i2)
    (i1 != i2)
'''
```

Loops (TLoop.py)

```
# Test Loop: while, for, nested loops (Lists, Tuples, Sets, and Dictionaries)
# Program-ID:   TLoop.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TLoop.py
#    $ python3  TLoop.py

a = [1, 2, 'elephant', 1] # list: a sequence of mutable objects
b = (1, 2, 3)             # tuple: a sequence of immutable objects
c = set(a)                # set: an unordered collection, no duplication
d = {'apple':'a', 'banana':'b', 'carrot':'c'} # dictionary: associative array

print ("Test while loop: list = ", a)
i = 0
while i < 4:
    print ("a[", i, "] =", a[i])
    i = i + 1
print ("Test for loop range(): tuple = ", b)
for i in range(len(b)):
    print ("b[", i, "] = ", b[i])
print ("Test for loop: set = ", c)
for v in c:
    print (v)
print ("Test for loop: dictionary = ", d)
for k, v in d.items():
    print (k, v)

e = [[10, 20, 30], [40, 50, 60], [70, 80, 90]]
print ("Test nested loop: No array in Python but similar to 2D array = ", e)
for i in range(len(e)):
    for j in range(len(e[i])):
        print (e[i][j])

print ("Test list: concatenation")
f = a + ['b', 'c']
print (f)
print ("Test list: insert")
f.insert(2, 'a')
print (f)
print ("Test list: remove")
f.remove('b')
print (f)
```

```
''' Output
     Test while loop: list =  [1, 2, 'elephant', 1]
     a[ 0 ] = 1
     a[ 1 ] = 2
     a[ 2 ] = elephant
     a[ 3 ] = 1
     Test for loop range(): tuple =  (1, 2, 3)
     b[ 0 ] =  1
     b[ 1 ] =  2
     b[ 2 ] =  3
     Test for loop: set =  {1, 2, 'elephant'}
     1
     2
     elephant
     Test for loop: dictionary =  {'apple': 'a', 'carrot': 'c', 'banana': 'b'}
     apple a
     carrot c
     banana b
     Test nested loop: No array in Python but similar to 2D array =  [[10, 20,
     30], [40, 50, 60], [70, 80, 90]]
     10
     20
     30
     40
     50
     60
     70
     80
     90
     Test list: concatenation
     [1, 2, 'elephant', 1, 'b', 'c']
     Test list: insert
     [1, 2, 'a', 'elephant', 1, 'b', 'c']
     Test list: remove
     [1, 2, 'a', 'elephant', 1, 'c']
'''
```

Subprograms (TSub.py)

```
# Test Subprograms: Call by Value and Call by Reference
# Program-ID:   TSub.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TSub.py
#    $ python3  TSub.py


def func1(i):
    i = i + 1
def func2(i):
    i = i + 2
    return i
def func3(i, j):
    i = i + 3
    j[0] = j[1] + 4

print("Test call by value")
m = 1
func1(m)
print("func1() m = ", m)
n = func2(m)
print("func2() n = ", n)

print("Test call by reference: list")
a = [10, 20, 30]
func3(a[1], a)
print("func3() a = ", a)

''' Output
    Test call by value
    func1() m =  1
    func2() n =  3
    Test call by reference: list
    func3() a =  [24, 20, 30]
'''
```

Other concerns: Test File I/O (TFile.py)

```
# Test File I/O: Read, Write
# An example program which
#     reads (TFileIn.dat) records from a input file,
#     displays the records,
#     and writes (TFileOut.dat)them to a sequential file.
# Program-ID:   TFile.py
# Author:       Kuo-pao Yang
# OS:           Ubuntu 18
# Interpreter:  Python 3
# Note:
# The following instructions are used to
#      edit and run this program
#    $ nano     TFile.py
#    $ python3  TFile.py

fIn = open("TFileIn.dat", "r")
fOut = open("TFileOut.dat", "w")
for line in fIn.read().splitlines():
    print(line)
    fOut.write(line + "\n")
fIn.close()
fOut.close()

''' Output
# TFileIn.dat
    abcd
    efg
# TFileOut.dat
    abcd
    efg
# Screen Outputs
    abcd
    efg
'''
```