Exercise 3 – Flow Control

Objective

To use the flow control structures of Python and to gain familiarity in coding based on indentation! That does take a little practice. We'll also be using a couple of modules from the Python standard library.

Questions

1. Using a for loop, display the files in your home directory, with their size.
    a) Use the skeleton file Ex3.py
    b) Get the directory name from the environment using os.environ, HOMEPATH on Windows HOME on Linux (we've done that part for you, notice the test of system.platform).
    c) Construct a portable wildcard pattern using os.path.join (we've done that part for you as well).
    d) Use the glob.glob() function to obtain the list of filenames.
    e) Use os.path.getsize() to find each file's size.
    f) Add a test to only display files that are not zero length.
    g) Use os.path.basename() to remove the leading directory name(s) from each filename before you print it.

2. Write a Python program that emulates the high-street bank mechanism for checking a PIN. Keep taking input from the keyboard (see below) until it is identical to a password number which is hard-coded by you in the program.

    To output a prompt and read from the keyboard:

    ```
    supplied_pin = input("Enter your PIN: ")
    ```

    Restrict the number of attempts to three (be sure to use a variable for that, we may wish to change it later), and output a suitable message for success and failure. Be sure to include the number of attempts in the message.

Optional extension

Passwords, and PINs, would not normally be displayed (*echoed*) to the screen for security reasons. So, now we will add the functionality to hide the characters typed. That could be a lot of work, but one of the advantages of using a language like Python is that "there's a module for it".

You'll need to **import** a module called **getpass**, which is part of the standard library.

Instead of **input** use **getpass.getpass**, in the same place in the program, with the same parameters.

Note you will have to run your program at the Windows or Linux command prompt (and not the Python shell) to test if it works!

3. Write a Python program to display a range of numbers by steps of -2.

    a) Prompt the user at the keyboard for a positive integer using:
    var = input ("Please enter an integer: ")

    b) Validate the input **(var)** to make sure that the user entered an integer using the **isdecimal()** method. If the user entered an invalid value, output a suitable error message and exit the program.

    c) Use a loop to count down from this integer in steps of 2, displaying each number on the screen until either 1 or 0 is reached. For example, if the integer 16 (validated) is entered, the output would be:

    16
    14
    12
    10
    8
    6
    4
    2
    0

    And if 7 is entered, the output would be:

    7
    5
    3
    1

    You will need to look-up the **range()** built-in in the online documentation, pay particular attention to the *stop* parameter.

If time allows...

4. If a year is exactly divisible by 4 but not by 100, the year is a leap year. There is an exception to this rule. Years exactly divisible by 400 are leap years. The year 2000 is a good example.

   Write a program that asks the user for a year and reports either a leap year or *not* a leap year. (*Hint*: x % y is zero if x is exactly divisible by y.) Test with the following data:

1984	is a leap year	1981	is NOT a leap year
1904	is a leap year	1900	is NOT a leap year
2000	is a leap year	2010	is NOT a leap year

   Use the following to ask the user for a year:

   ```
   year = int(input('Please enter a year: '))
   ```

5. Examine the code template provided in weekday.py. Complete the program to ask for a date in DD/MM/YYYY format and print out the day of the week for this date.

   There is a formula, called *Zeller's Congruence*, which calculates the day of the week from a given day, month and year. Zeller's congruence is defined as follows:

   ```
   z = (1 + d + (m*2)
        + (3 * (m+1)/5)
        + y + y/4 - y/100 + y/400) % 7
   ```

   where d, m and y are day, month, year and z are an integer (0 = Sun, 6 = Sat).

   Add the following adjustments *before* using in the formula:

   If month is 1 or 2 and year is a leap year, subtract 2 from day.
   If month equals 1 or 2 and year is not a leap year, subtract 1 from day.
   If month is 1 or 2, add 12 to month.

   Your program should print out the name of the day (e.g. Monday), e.g.:

1/1/1980	Tuesday	9/8/1982	Monday
25/12/1983	Sunday	31/5/1989	Wednesday
2/2/1990	Friday	29/2/1992	Saturday

Solutions

Question 1

Here's our solution:

```
import sys
import glob
import os

# Get the directory name.
if sys.platform == 'win32':
    hdir = os.environ['HOMEPATH']
else:
    hdir = os.environ['HOME']

# Construct a portable wildcard pattern.
pattern = os.path.join(hdir, '*')

# Use the glob.glob() function to obtain the list of filenames.
for filename in glob.glob(pattern):

    # Use os.path.getsize to find each file's size.
    size = os.path.getsize(filename)

    # Only display files that are not zero length.
    if size > 0:
        print(os.path.basename(filename), size, 'bytes')
```

Question 2

There are several valid ways to write this code. Here's one solution:

```
import sys

PIN = '0138'
LIMIT = 4

for tries in range(1, LIMIT):
    supplied_pin = input('Enter your PIN: ')
    if supplied_pin == PIN:
        print('Well done, you remembered it!')
```

```
        print('... and after only', tries, 'attempts')
        break
# Note the else: is indented with the for loop, not the if!
else:
    print('You had', tries, 'tries and failed!')
```

Note that we used **uppercase** as a convention for constants, and we took advantage of the **else** on a **for** loop that is *not* executed on a **break**.

Optional extension to question 2

Using **getpass**, which is part of the standard library:

```
import sys
import getpass

PIN = '0138'
LIMIT = 4

for tries in range(1, LIMIT):
    supplied_pin = getpass.getpass('Enter your PIN: ')
    if supplied_pin == PIN:
        print('Well done, you remembered it!')
        print('... and after only' , tries, 'attempts')
        break
# Note the else: is indented with the for loop, not the if!
else:
    print('You had', tries, 'tries and failed!')
```

Why didn't we use **getpass** in the main question? Because making the input invisible makes debugging more difficult.

Question 3

Here's one simple solution using the range function:

```
var = input("Please enter an integer: ")

if not var.isdecimal():
    print("Invalid integer:", var)
    exit(1)
```

```
for var in range(int(var), -1, -2):
    print(var)
```

Question 4

Here's our solution to test for leap years:

```
y = int(input('Please enter a year: '))

if y%4 == 0 and (y%400 == 0 or y%100 != 0):
    print("Leap Year")
else:
    print("NOT a leap year")
```

Question 5

Here's our solution to print out the day of the week using Zellar's congruence:

```
# Code for reading in the date.
date = input('Please enter date (DD/MM/YYYY): ')
d, m, y = date.split('/')
d = int(d)
m = int(m)
y = int(y)

if m == 1 or m == 2:
    m += 12
    if y%4 == 0 and (y%400 == 0 or y%100 != 0):
        d -= 2
    else:
        d -= 1

z = 1 + d + (m*2) + (3 * (m+1)//5) + y + y//4 - y//100 + y//400
z %= 7

days = ['Sun', 'Mon', 'Tues', 'Wednes', 'Thurs', 'Fri', 'Satur']
print(days[z] + 'day')
```