Lab Questions: Lab Session 8

Deadline: 18.10.2017 11:59pm SGT

Complete all assignments below. For those questions that are marked with an asterisk *, i.e. **Questions 5 and ??**, create the script files as requested. Once you are done with it, submit the file(s) via `iNTU`. Remember to put plenty of comments for all assignments, as this helps us to better understand your program (which might give you higher marks).

Important!!! Make sure your scripts work properly, as we give 0 marks otherwise. Please name the scripts according to the requirements, and upload each file separately and not in a Zip file or similar. The submission system closes at the deadline. Hence after that, you will get no marks for your solution.

1. Transform these list comprehension statements into equivalent code using loops:

(a)
```
seven = [x for x in range(1,100) if x%7==0]
```

Solution:

```
seven = []
for x in range(1,100):    # for all integers in [1,99]
   if x%7==0:
      seven.append(x)
```

(b)
```
shift = [(x+4,y-5) for (x,y) in zip(range(40,100),reversed(range(40,100)))]
```

Solution:

```
shift = []
listx = list(range(40,100))
listy = list(reversed(range(40,100)))
for i in range(len(listx)):
    shift.append((listx[i]+4,listy[i]-5))
```

(c)
```
primes = [x for x in range(2,100) if sum([float(x/y).is_integer() for y in range
    (2,x)])==0]
```

Solution:

```
primes = []
for x in range(2,100):    # for all integers in [2,99]
   my_sum = 0
   for y in range(2,x):  # count how many times this number can be divided
      if float(x/y).is_integer():
          my_sum += 1

   if my_sum == 0:  # if it can't be divided, it is a prime number
      primes.append(x)
```

2. Transform these loops into an equivalent code using list comprehension:

(a)
```
squares = []
for x in range(10):
    squares.append(x**2)
```

Solution:

```
squares = [x**2 for x in range(10)]
```

(b)
```
combs = []
for x in [1,2,3]:
    for y in [3,1,4]:
        if x != y:
            combs.append((x, y))
```

Solution:

```
[(x, y) for x in [1,2,3] for y in [3,1,4] if x != y]
```

(c)
```
list1 = range(1,10)
list2 = range(1,20,2)
list3 = range(1,40,4)
L = []
for i in range(len(list1)):
    if list1[i]*list2[i]>list3[i]:
        L.append(list1[i]*list2[i]+list3[i])
```

Solution:

```
L = [x*y+z for (x,y,z) in zip(range(1,10),range(1,20,2),range(1,40,4)) if x*y>z]
```

3. Write a list comprehension that uses nested for-clauses to create a list `L` of 2-element tuples, containing all 36 different dice combinations from (1,1) to (6,6). Repeat the exercise, but this time generate only the 21 different unordered dice combinations from (1,1) to (6,6), in other words (1,2) and (2,1) are considered the same combination.

Solution:

```
L = [(x, y) for x in range(1,7) for y in range(1,7) ]  # 36 combinations
```

```
L = [(x, y) for x in range(1,7) for y in range(1,7) if x<=y ]  # 21 combinations
```

4. Create a script `temp.py` that continues asking the user to enter temperatures in degrees Celsius and store them in a list `L`. The process stop when the user enters the character 'e'. Then, using list comprehension, generate a new list `LT` of tuples, where each tuple has two values, a temperature in Celsius and its corresponding temperature in Farenheit (you can use the formula $F = 1.8C + 32$).

Solution:

temp.py

```
L=[]
while True:
    user_input = input('Please enter a temperature in Celsius degrees: ')
    if user_input == 'e':
       break
    else:
       L.append(float(user_input))

LT = [(c,1.8*c+32) for c in L]
print(LT)
```

5. * Assume that you have made some temperature measurements at 50 different points of time in three different places and you have stored the values in three lists `temp1`, `temp2`, `temp3` (each list corresponding to a place, and each i-th element in a list representing the temperature at time i). To simulate this situation, you can initialize the three lists with random elements in $[0, 40]$. Write a script <`YourMatricNo`>`_Lab8_comprehension.py`, that will initialize `temp1`, `temp2`, `temp3` and then compute three new lists (all using list comprehension):

   - `warm`: a list of 3-temperature tuples, containing the generally warm temperatures. More precisely, if at time i you measured temperatures $t1$, $t2$ and $t3$ such that $t1+t2+t3 > 75$, then the tuple $(t1, t2, t3)$ should be added to the list `warm`.

   - `all_cold`: a list of 3-temperature tuples, containing the all-cold temperatures. More precisely, if at time i you measured temperatures $t1$, $t2$ and $t3$ such that all three temperatures are strictly lower than 10 degrees, then the tuple $(t1, t2, t3)$ should be added to the list `all_cold`.

   - `similar`: a list of 3-temperature tuples, containing the similar temperatures. More precisely, if at time i you measured temperatures $t1$, $t2$ and $t3$ such that the temperature differences between $t1$ and $t2$, between $t1$ and $t3$, between $t2$ and $t3$, are strictly smaller than 5, then the tuple $(t1, t2, t3)$ should be added to the list `similar`.

   Solution:

comprehension.py

```
import random

temp1 = [random.randint(0,40) for x in range(50)]
temp2 = [random.randint(0,40) for x in range(50)]
temp3 = [random.randint(0,40) for x in range(50)]

hot   = [(x,y,z) for (x,y,z) in zip(temp1,temp2,temp3) if x+y+z>75]
all_cold = [(x,y,z) for (x,y,z) in zip(temp1,temp2,temp3) if x<10 and y<10 and z<10]
similar = [(x,y,z) for (x,y,z) in zip(temp1,temp2,temp3) if abs(x-y)<5 and abs(x-z)<5
     and abs(z-y)<5]
```