Practical Questions

CSCA08 Fall 2017 – Week 8

Question # 1

a) Trace the following code: What is printed?

```
a = 1
b = 2
c = 3
print (a, b, c)
(a, b, c) = (b, c, a)
print (c, a, b)
(b, a, c) = (b*c/a, a**c+b, a+b+c)
print (b, c, a)
```

b)

```
program_info = [
    ("John", ["CompSci", "Physics"]),
    ("Vusi", ["Maths", "CompSci", "Stats"]),
    ("Jess", ["CompSci", "Accounting", "Economics", "Management"]),
    ("Sarah", ["InfSys", "Accounting", "Economics", "CommLaw"]),
    ("Zuki", ["Sociology", "Economics", "Law", "Stats", "Music"])]
print(program_info[3][1][2])
```

-Change Sarah's program from InfSys to Information Systems and from CommLaw to Criminal Law

-Add data for a new student named Bruce who is studying Cell and Molecular Biology, Biomedical Toxicology and Mechatronics

c) Write a function that takes in one variable that contains a student's age, student number, name, street address and height, and returns the students name and their identifier ((age * height/ street number)+student number)

d) Trace the following code: What is printed?[1]

```
my_set = set([1,3,5,7,9])
print (my_set)
my_set.add(3)
print (my_set)
my_set.add(6)
print (my_set)
```

[1] When dealing with sets & dictionaries, we don't actually know the order in which the elements will be printed, when asked to trace them any possible correct output will be accepted.

e) Write a function that takes in a string and returns all the unique characters in the string using sets.

f) Write a function that takes in a string and returns only the characters that were repeated in the entered string, using sets.

g) Write a function that takes in two strings of characters and checks if all the characters in the first string are found in the second string, without using loops. (Hints: You may add a requirement that all characters in the second string have to be in the first string)

h)

```
my_dict ={“50’s”:6, “60’s”:18, “70’s”:37, “80’s”:16, “90’s”:5}
print (my_dict[“60’s”])
print (my_dict[“90’s”])
```

i)

```
my_dict ={“50’s”:38, “60’s”:18, “70’s”:2, “80’s”:16, “90’s”:39}
print( my_dict)
my_dict[“40’s”] = 6
print (my_dict)
my_dict[“80’s”] = 20
print (my_dict)
```

j)

```
my_dict ={“40’s”:17,“50’s”:6, “60’s”:14, “70’s”:2, “80’s”:37, “90’s”:5}
for key in my_dict:
    print (key, my_dict[key])
    my_dict[key] = my_dict[key]*2
print (my_dict)
del my_dict[“80’s”]
print (my_dict)
```

k) Write a function that takes in 6 lists(names, ages, student numbers, utorid, campus, department) and returns a dictionary that maps student numbers to the other data

l) **(Challenge)**
https://open.kattis.com/problems/babelfish

Discussion Questions

Question # 2

There are actually 2 types of sets, the mutable kind, like taught in lecture and an immutable kind, called the frozen set. The frozen set does not have add or delete methods, which means that new values cannot be added and values cannot be removed after the set has been defined. Why would you want to have a set that cannot be changed? What benefits come from this property? What could you have to do if you wanted to add or remove a value?

Question # 3

This week, you learned about 3 new ways to store data beside lists, tuples dictionaries and sets. Each of ways of storing data has their own strengths and drawbacks, which you, as a computer scientist, have to consider in order to determine which is most appropriate for your situation. Which might be the best container type to represent a print queue? A person's medical information? A business full of people? A text document? Is there always one right answer? Is there always any right answer?

Question #4

Dictionaries can't have lists, sets or other dictionaries as their keys. Why not? (hint: think about the memory model and mutability)

Question # 4, The “Logic Question” **(Challenge)**

The mad king has captured you once again. This time, they let you go for a ransom of 1000 gold coins[2]. The guards now have to decide how to divide the coins among themselves. There are 100 guards who are perfectly logical, perfectly obedient (they will always follow the rules, and trust that the other guards will as well), and perfectly greedy (they only care about maximizing their own gold share). They also won’t kill anyone unless it will increase their gold share (if two scenarios give them equal gold, they’d prefer the one that doesn’t involve killing).

The guards come up with rules for dividing the money. The most senior guard (there’s a very strict hierarchy) will propose a division (who gets what), and then they’ll all vote. If 50% or more of the guards accept the proposal, they divide the gold and all are happy. If fewer than 50% vote in favour, they will kill the senior guard, and then the next most senior guard gets to propose a division. This continues until someone gets 50% or more of the votes.

How much gold does each guard get?

[2] Maybe they wouldn’t have caught you if you weren’t slowed down by carrying around huge bags of coins everywhere.