PSG College of Technology

Department of Applied Mathematics and Computational Sciences

Python - Laboratory

**Problem Sheet – Basics**

1. Write a program that accepts a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically. Suppose the following input is supplied to the program:

hello world and practice makes perfect and hello world again

Then, the output should be:

again and hello makes perfect practice world

1. By using list comprehension, please write a program to print the list after removing the value 24 in [12,24,35,24,88,120,155].
2. With two given lists [1,3,6,78,35,55] and [12,24,35,24,88,120,155], write a program to make a list whose elements are intersection of the above given lists.
3. Write a Python program to find maximum and the minimum value in a set.
4. Write a Python program to find the number of elements in a set.
5. Write a Python program to clear a set.
6. Write a Python program to test whether every element in A is in B and every element in B is in A. If so, print “The two sets A and B are equal”, if not, store the elements in B which are not in A in the name B1 and store the elements in A which are not in B in the name B2. Print both B1 and B2.
7. Get two sets {} and {} as user input in the name A and B respectively. Write a program to find the resultant set which is (i) intersection of A and B (ii) Union of A and B (iii) A-B (difference: Formally A–B = {s | s A and s B}) (iv) AB (symmetric difference: Formally AB=(A-B) (B-A)).
8. Write a Python program to create an empty set, add the members “White, Red, Blue, Green” in the set, remove the member “Red”, check the presence of a member “yellow” in the list. (Print “Yes present” if it is present and “No, not present” if it is not).
9. Write a program concerns the problem of making an index for a book. A related problem is making a concordance for a document. A concordance lists every word that occurs in the document, and for each word it gives the line number of every line in the document where the word occurs. Input the document as a multi-line string with each line starting with line numbers.
10. Write a program to carry out commands of the form "let variable = expression" or "print expression". That program can handle expressions that contain variables, numbers, operators, and parentheses.
11. Let’s say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]. Write one line of Python that takes this list ‘a’ and makes a new list that has only the even elements of this list in it. (Hint: use list comprehension)
12. Let’s continue building Hangman. In the game of Hangman, a clue word is given by the program that the player has to guess, letter by letter. The player guesses one letter at a time until the entire word has been guessed. (In the actual game, the player can only guess 6 letters incorrectly before losing).

Let’s say the word the player has to guess is “EVAPORATE”. For this exercise, write the logic that asks a player to guess a letter and displays letters in the clue word that were guessed correctly. For now, let the player guess an infinite number of times until they get the entire word. As a bonus, keep track of the letters the player guessed and display a different message if the player tries to guess that letter again. Remember to stop the game when all the letters have been guessed correctly! Don’t worry about choosing a word randomly or keeping track of the number of guesses the player has remaining - we will deal with those in a future exercise.

An example interaction can look like this:

>>> Welcome to Hangman!

\_ \_ \_ \_ \_ \_ \_ \_ \_

>>> Guess your letter: S

Incorrect!

>>> Guess your letter: E

E \_ \_ \_ \_ \_ \_ \_ E

...

And so on, until the player gets the word.

1. Write a program (function!) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates.
2. Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your code in a main method.