

Master of Engineering in Internetworking

INWK 6312

Programming for Internetworking Applications

Lab 2

Iterations, Strings, and Word Play

Table of Contents

[Chapter 1 - Prerequisites 3](#_Toc493181803)

[Chapter 2 - Introduction 3](#_Toc493181804)

[Chapter 3 - Objectives of the lab 3](#_Toc493181805)

[Chapter 4 - Iterations 4](#_Toc493181806)

[Important concepts 4](#_Toc493181807)

[Task 1 4](#_Toc493181808)

[Task 2 4](#_Toc493181809)

[Task 3 5](#_Toc493181810)

[Chapter 5 - Strings 5](#_Toc493181811)

[Important concepts 5](#_Toc493181812)

[Task 4 6](#_Toc493181813)

[Task 5 6](#_Toc493181814)

[Task 6 6](#_Toc493181815)

[Task 7 6](#_Toc493181816)

[Task 8 7](#_Toc493181817)

[Chapter 6 - Word Play 7](#_Toc493181818)

[Important concepts 7](#_Toc493181819)

[Task 9 7](#_Toc493181820)

[Task 10 8](#_Toc493181821)

[Task 11 8](#_Toc493181822)

[Task 12 8](#_Toc493181823)

[Task 13 8](#_Toc493181824)

[Task 14 9](#_Toc493181825)

# Prerequisites

Ensure that you have completed the tasks mentioned in Lab 1. Also, ensure that you can achieve all the objectives of lab 1.

# Introduction

This lab is to introduces you to the Iterations, perhaps one of the most important part of programming. We will look at the “for” loop and the “while” loop. This is followed by understanding and using basic string manipulations which are essential for any programming task. Finally, world play enables us to put together all these concepts into practice.

The key concepts in this lab are: Encapsulation, Interface Design, Development Plan and Incremental Development.

# Objectives of the lab

At the end of this lab you will have learnt the following:

* How and Why to iterate
* To effectively use the for and the while loops
* Understands basics about strings
* Slice strings
* To read the string documentation and implement the string methods correctly
* Iterate through strings and manipulate them based on requirements

# Iterations

## Important concepts

**multiple assignment:** Making more than one assignment to the same variable during the execution of a program.

**update:** An assignment where the new value of the variable depends on the old.

**initialization:** An assignment that gives an initial value to a variable that will be updated.

**increment:** An update that increases the value of a variable (often by one)

**decrement:** An update that decreases the value of a variable.

**iteration:** Repeated execution of a set of statements using either a recursive function call or a loop.

**infinite loop:** A loop in which the terminating condition is never satisfied.

## Task 1

while True:

print x

y = (x + a/x) / 2

if y == x:

break

x = y

*Encapsulate this loop in a function called square\_root that takes****a****as a parameter, chooses a reasonable value of****x****, and returns an estimate of the square root of****a****.*

*Bonus: Replace the if statement (****y==x****) with “*abs(***y-x***) < *epsilon*”,

Where *epsilon* has a value like 0.0000001 that determines how close is close enough

## Task 2

*To test the square root algorithm in this task, you could compare it with math.sqrt. Write a function named test\_square\_root that prints a table like this:*

*1.0 1.0 1.0 0.0*

*2.0 1.41421356237 1.41421356237 2.22044604925e-16*

*3.0 1.73205080757 1.73205080757 0.0*

*4.0 2.0 2.0 0.0*

*5.0 2.2360679775 2.2360679775 0.0*

*6.0 2.44948974278 2.44948974278 0.0*

*7.0 2.64575131106 2.64575131106 0.0*

*8.0 2.82842712475 2.82842712475 4.4408920985e-16*

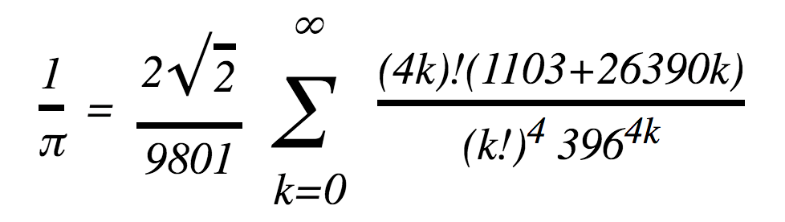
*9.0 3.0 3.0 0.0*

*The first column is a number, a; the second column is the square root of a computed with the function from question 1; the third column is the square root computed by math.sqrt; the fourth column is the absolute value of the difference between the two estimates.*

## Task 3

*The mathematician Srinivasa Ramanujan found an infinite series that can be used to generate a numerical approximation of 1 / π:*

|  |
| --- |
|  |

**

*Write a function called estimate\_pi that uses this formula to compute and return an estimate of π. It should use a while loop to compute terms of the summation until the last term is smaller than 1e-15 (which is Python notation for 10−15). You can check the result by comparing it to math.pi.*

# Strings

## Important concepts

**sequence:** An ordered set; that is, a set of values where each value is identified by an integer index.

**item:** One of the values in a sequence.

**index:**  An integer value used to select an item in a sequence, such as a character in a string.

**slice:** A part of a string specified by a range of indices.

**empty string:** A string with no characters and length 0, represented by two quotation marks.

**immutable:** The property of a sequence whose items cannot be assigned.

**traverse:** To iterate through the items in a sequence, performing a similar operation on each.

**search:** A pattern of traversal that stops when it finds what it is looking for.

**counter:** A variable used to count something, usually initialized to zero and then incremented.

**method:** A function that is associated with an object and called using dot notation.

**invocation:** A statement that calls a method.

***Read the documentation of the string methods at***<http://docs.python.org/2/library/stdtypes.html#string-methods>*.*

*You might want to experiment with some of them to make sure you understand how they work.  strip and replace are particularly useful.*

*The documentation uses a syntax that might be confusing. For example, in find(sub[, start[, end]]), the brackets indicate optional arguments. So sub is required, but start is optional, and if you include start, then end is optional.*

## Task 4

*Write a function that takes a string as an argument and displays the letters backward, one per line.*

## Task 5

*There is a string method called count . Read the documentation of this method and write an invocation that counts the number of a(s)’ in 'banana'.*

## Task 6

*Write a Python program to add 'ing' at the end of a given string (length should be at least 3). If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string is less than 3, leave it unchanged.*

## Task 7

*A string slice can take a third index that specifies the “step size;” that is, the number of spaces between successive characters. A step size of 2 means every other character; 3 means every third, etc.*

*>>> fruit = 'banana'*

*>>> fruit[0:5:2]*

*'bnn'*

*A step size of -1 goes through the word backwards, so the slice [::-1] generates a reversed string.*

*Use this idiom to write a one-line version of is\_palindrome  from lab 2.*

## Task 8

*ROT13 is a weak form of encryption that involves “rotating” each letter in a word by 13 places. To rotate a letter means to shift it through the alphabet, wrapping around to the beginning if necessary, so ’A’ shifted by 3 is ’D’ and ’Z’ shifted by 1 is ’A’.*

*Write a function called rotate\_word that takes a string and an integer as parameters, and that returns a new string that contains the letters from the original string “rotated” by the given amount.*

*For example, “cheer” rotated by 7 is “jolly” and “melon” rotated by -10 is “cubed”.*

*You might want to use the built-in functions ord, which converts a character to a numeric code, and chr, which converts numeric codes to characters.*

*Potentially offensive jokes on the Internet are sometimes encoded in ROT13. If you are not easily offended, find and decode some of them.*

# Word Play

## Important concepts

**file object:** A value that represents an open file.

**problem recognition:** A way of solving a problem by expressing it as an instance of a previously-solved problem.

**special case:** A test case that is atypical or non-obvious (and less likely to be handled correctly)

## Task 9

*Write a program that reads words.txt and prints only the words with more than 20 characters (not counting whitespace).*

## Task 10

*Write a function called has\_no\_e that returns True if the given word doesn’t have the letter “e” in it.*

*Modify your program from the previous section to print only the words that have no “e” and compute the percentage of the words in the list have no “e.”*

## Task 11

*Write a function named avoids that takes a word and a string of forbidden letters, and that returns True if the word doesn’t use any of the forbidden letters.*

*Modify your program to prompt the user to enter a string of forbidden letters and then print the number of words that don’t contain any of them. Can you find a combination of 5 forbidden letters that excludes the smallest number of words?*

## Task 12

*Write a function named uses\_all that takes a word and a string of required letters, and that returns True if the word uses all the required letters at least once. How many words are there that use all the vowels aeiou? How about aeiouy?*

## Task 13

*Write a function called is\_abecedarian that returns True if the letters in a word appear in alphabetical order (double letters are ok). How many abecedarian words are there?*

## Task 14

*This question is based on a Puzzler that was broadcast on the radio program Car Talk (*<http://www.cartalk.com/content/puzzlers>*):*

*Give me a word with three consecutive double letters. I’ll give you a couple of words that almost qualify, but don’t. For example, the word committee, c-o-m-m-i-t-t-e-e. It would be great except for the ‘i’ that sneaks in there. Or Mississippi: M-i-s-s-i-s-s-i-p-p-i. If you could take out those i’s it would work. But there is a word that has three consecutive pairs of letters and to the best of my knowledge this may be the only word. Of course there are probably 500 more but I can only think of one. What is the word?*

*Write a program to find it.*