LONDON
METROPOLITAN
UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title
CC4002NA Information Systems

Assessment Weightage & Type
20% Individual Coursework

Year and Semester
2018-19 autumn

Student Name: Suraksha Shrestha
London Met ID:
College ID: NP01NT4A180002
Assignment Due Date: 12/21/2018
Assignment Submission Date:12/21/2018

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents:

List of figures:

Introduction:

Bubble sort algorithm is a kind of program that is used to sort the numbers of list in either ascending or descending order based on the idea of repeatedly comparing the adjacent element and swapping them if they bare in wrong order. It follows certain path and algorithm. This program utters repeatedly through the list and compares the consecutive pair of elements. Because of this nature, bubble sort algorithm is not considered to be efficient. To arrange a large number of elements in a list, it consumes a lot of time. But in case of shorter list, it is very efficient. (Sehgal, feb 11)

Firstly, we use loop in which we compare the adjacent elements and swap them if it is necessary. In one loop, the program iterates through each element which is called as one pass. We cannot determine exactly how many passes can be taken in a bubble sort algorithm but we can say it does not exceed the length of the list. In more precise way, if we take n numbers of elements in a list then the number of pass is considered to be n times.

While comparing if the given condition becomes true, it swaps the consecutive elements. In first pass, after comparing each adjacent elements, the list ends and again the program enters into the loop. If we are going to arrange the elements in ascending order, after the first pass, the greatest element takes the last place in the list. In this way again the second greatest element reach second last position and so on. Each time the elements are swapped, the variable(number of swaps) is increased by 1. Since, the computer does not know whether the arrangement is done or not, so it again enters the loop and this time the number of swaps becomes 0. Alternatively, the program comes out of the loop after the number of swaps becomes 0.

Illustration of a bubble sort program is given below:

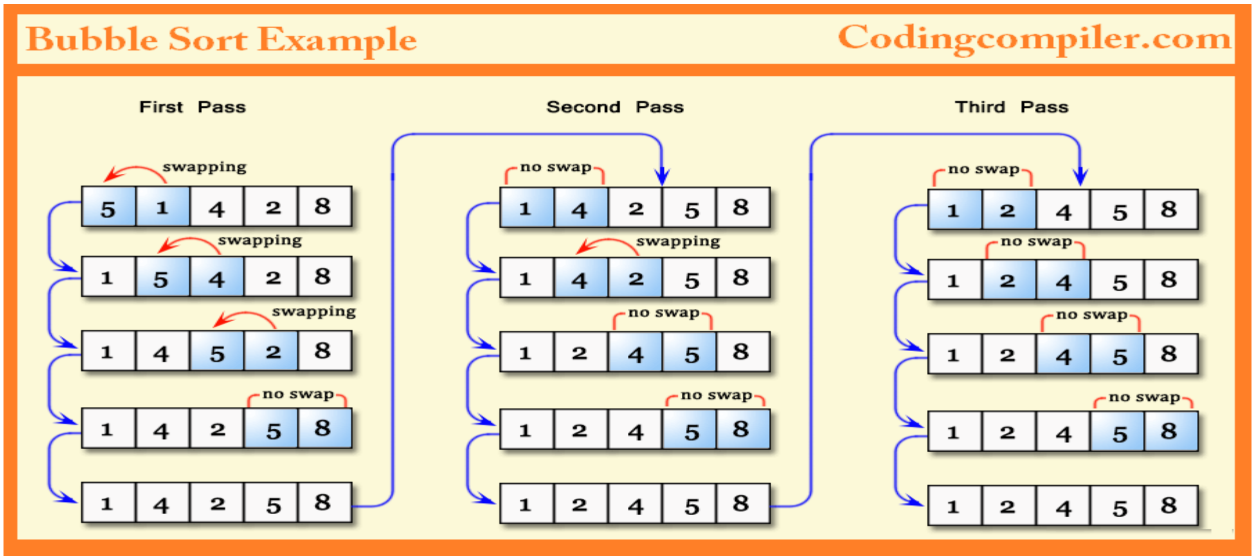


Figure 1:Bubble sort algorithm (Anon., 2017)

The Bubble sort algorithm:

This algorithm iterates through each element in the list using loop and arranges them in the format you want. The procedures are briefly described in the steps below:

- Step 1: Start.
- Step 2: Input a list.
- Step 3: Declare a flag.
  i.e. sorted==False
- Step 4: If sorted=False, got to step 5 otherwise got to step 13.
- Step 5: Declare a counter for counting the number of swaps.
  num_swaps=0
- Step 6: Assign the value for i.
  i = 0
- Step 7: Use loop where value of i goes upto length of list.
  If i <= len (list)
- Step 8: If the condition becomes true, go to step 9, else go to step 4.
- Step 9: Again, if L[i-1]>L[i], go to step 10 else go to step 11.
- Step 10: Increase the counter, num_swaps by 1:
  num_swaps=num_swaps+1
- Step 11: Increase i by 1 and go to step 7.
  i+=1
- Step 12: If num_swaps=0, change the flag to True and go to step 4.
- Step 13: Print list
- Step 14: Stop

Graphical representation of the algorithm:

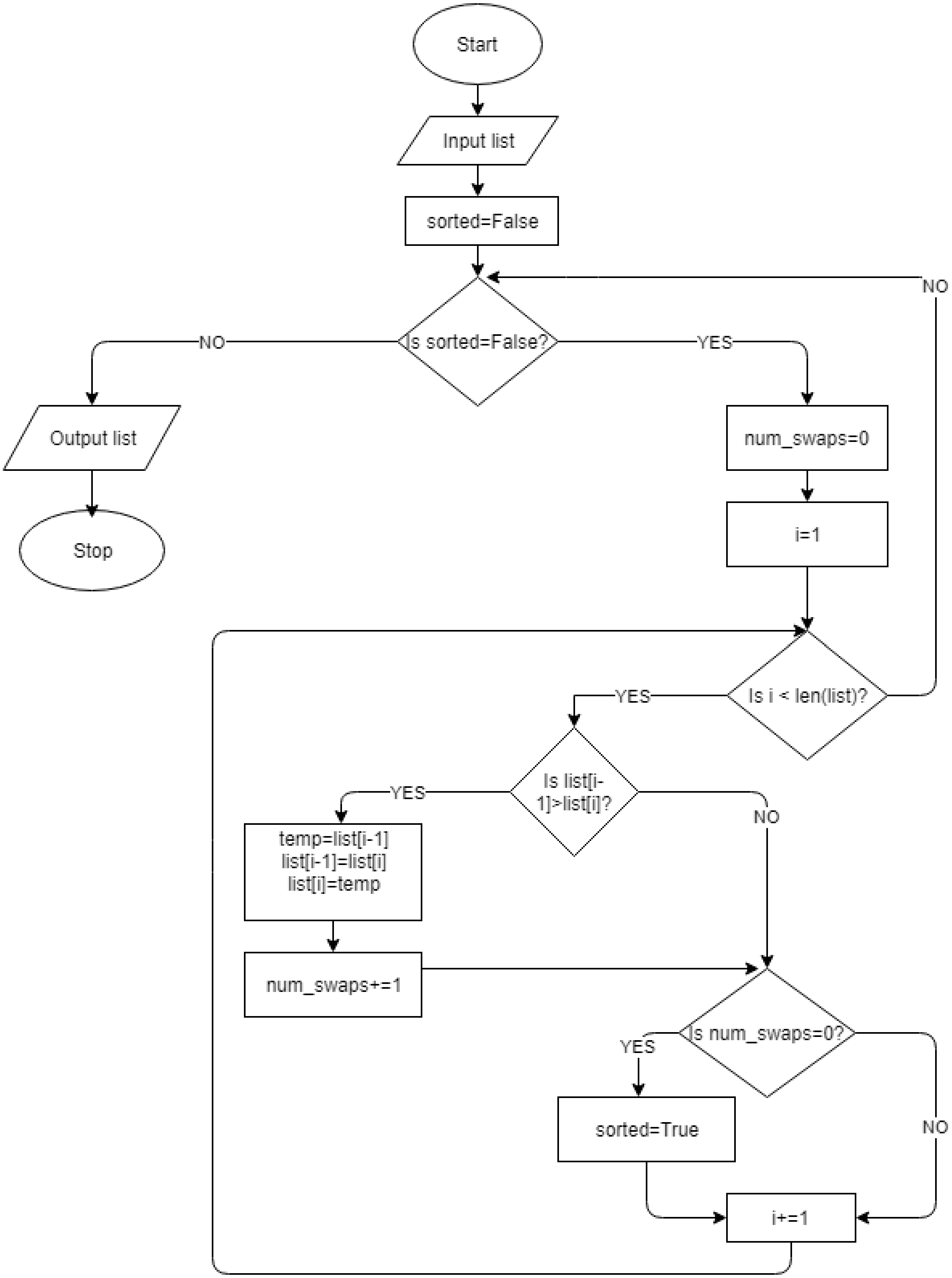


Figure 2:Flowchart of the algorithm

Above flowchart is briefly described in steps as below:

Working process:

- Step 1: Firstly, a list is assigned to the program by the user. For now, let us assume a list (L) given as below:

  L= [5,2,4,3]

- Step 2: Now we iterate through each item of the list plugging it into a loop. Here, a variable is assigned by the value 'False' and if it becomes 'True', the program comes out of the loop. The passes of the list during this loop is described briefly as below:

  - Pass 1:

➢ Compares the first and second elements only and swaps if L[1] > L[2]:
i.e. from [**5**,**2**,4,3] to [**2**,**5**,4,3]

➢ Compares the second and third element:
i.e. from [2,**5**,**4**,3] to [2,**4**,**5**,3]

➢ Compares the third and fourth element:
i.e. from [2,4,**5**,**3**] to [2,4,**3**,**5**]

Therefore, in the first pass, the greatest number of element if placed at last position.

  - Pass 2:

➢ Compares the first and second element:
[**2**,**4**,3,5] to [**2**,**4**,3,5]

➢ Compares second and third element:
[2,**4**,**3**,5] to [2,**3**,**4**,5]

➢ No swapping at last because it is already in order.

  - Pass 3:

➢ Compares similarly as above. No swapping occurs.

Since no swapping occurs, the value of variable becomes 'True' and it ultimately comes out of the loop.

- Step 3: At last, the list is printed in the sorted format:
  i.e. L = [2,3,4,5]

The program ends here by sorting the elements into ascending order.

Data types in Python:

In python, we store the values in their own data type such as int for integers, float for decimal, etc. There as generally two types of data types, we use in python which are briefly described as below:

1. Primitive datatype: According to the type of element we are going to use in the program, there are some basic data types and they are int, float, double, boolean, etc. These data types cannot be broken down because they are not made up of other data types.

2. Collection datatype: When we need to store more than one information of an object, then we usually use collection data type. Here, we can store different types of data, it can be integer or string. So, this data type is complex. For example: list, tuple, dictionary, 2D list, etc.

2.Collection datatype:

- **List:** While storing different data, list is very versatile. Commonly, in different collection datatype, list is used. List stores the data between two curly brackets{ } and separated by commas. The examples of list are:

  List1 = [1,2,3,4,5]
  List 2 = ["Kathmandu", "Dhading", "Gorkha"]
  List 3 = ['a', 'b', 'c']

  The elements of the list can be accessed using indexes. The index starts from 0, so the last element of the list goes in the index N-1 if the number if the number of elements in the list is N. Every element has their own indexing and it is denoted by L[N]. For example:

  L = [2,5,3,9]
  Here,
  L [0] = 2, L [1] = 5 and so on.

  If we count from the last, the indexes start from -1. For example, in the list above:
  L [-1] =9, L [-2] =3, L [-3] =5 and so on.

- **Tuple:** Tuple is similar as a list but the data stored in it cannot be changed or it is immutable. The data is stored in between the small brackets (). Different type of data can be stored in tuple as well. The example is given below:

  T= (1,2,3,4)
  T= ('a', 'b', 'c', 'd')

Indexing in tuple is same as in list, i.e. it starts from 0 to len(L) -1. We can access the elements by using the same function i.e. L[N].

- **Dictionaries:** It is unordered collection of the data, so there is no concept of indexing. The data in dictionaries can be easily fetched and are mutable as well. In more precise way, the data can be added to deleted easily. Dictionaries are denoted by curly brackets { }.The data does not have definite order. For example:

  D={‘ vehicle’ : car, ‘color’ : red, ‘brand’ : Suzuki}

  Dictionaries store data in the form of key-value, the keys are unique. Here, vehicle, color and brand are the unique keys and car, red and Suzuki are the values.

- **Set:** Set are also unordered collection of data. Data does not follow definite order, thus set does not have the concept of indexing. Set is denoted by curly brackets { }. The element in set must be unique i.e. no duplicates can be used. In case, if we put the duplicates, it does not include the other duplicates in the list. For example:

  a = {1, 2, 3, 4}
  b = {3, 5, 7, 6}

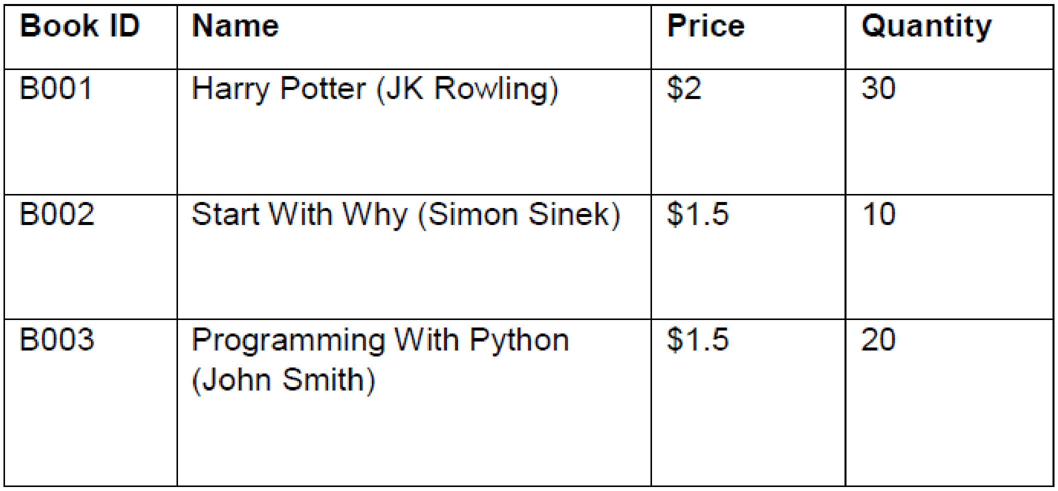
  With the help of set, we can find the operations like intersection, union, compliment, etc of two sets. A demonstration of syntax is given below:

```
print (a.union (b) )              >>> { 1, 2, 3, 4, 5, 7, 6}
print (a.intersection (b) )       >>> {3}
print (a.difference (b) )         >>> {1, 2, 4}
```

These are some examples of operations. We can also perform some other tasks using sets.

Question:

Below is a table consisting of some data related to a book store. Chose a suitable collection data type for representing and storing that data in a python program and justify your choice in terms of convenience, economy and efficiency.



Book ID	Name	Price	Quantity
B001	Harry Potter (JK Rowling)	$2	30
B002	Start With Why (Simon Sinek)	$1.5	10
B003	Programming With Python (John Smith)	$1.5	20

Figure 3: Data related to book store

In my sense, I think storing the given data in dictionary is more convenient in comparison to other because there are different pieces data for 1 entry, i.e. we need to store book ID, name, price and the quantity, and dictionary stores the data using key-value pair. We know, in dictionary a large data can be stored in a single variable and that data can be fetched and changed easily. After saving these different data of the book, if we have to add or remove any data history, we can easily have an access. On the one hand, since list stores the data in single line format, if we store the data in the form of the list then it would be confusing to know which data indicates which key. For example: if we store the given data in list the result may come as below i.e.

L = [“B001”, 2, 30, “Harry Potter”, “B002”, “Start With Why”, 1.5, 10]

By looking the list above, we cannot know which value or string represents what information about the book. So, using list is not convenient.

Similarly, on the other hand, when we use tuple, if in future, we have to add or remove any information related to the book then we cannot fetch it because tuple is immutable. Thus, tuple is also not so convenient.

Talking about sets, the data stored in set is unordered. So, whenever we want to fetch a data, we would not know the exact position of it. Therefore, it would be difficult to manipulate the data.

Lastly, with the merits of storing data in key-value pair and ordered position, the above data can be stored using dictionary. The illustration is given as below:

Data = {'Book ID': ['B001', 'B002', 'B003'], 'Name': ['Harry Potter (JK Rowling)', 'Start With Why (Simon Sinek), 'Programming With Python (John Smith)', 'Price':['$2', '$1.5', '$1.5'], 'Quantity':[30, 10, 20] }

Now, we can easily fetch the items or data of the book store which are mutable.

Learning reflection:

Before I started this module, I had studied java programming on my own. So, I knew bits and pieces of programming. I learned some syntaxes of this language. So, it was a bit easy to catch up the things in the lecture classes. When this module started and as the classes went on, those bits and pieces started coming together.

The expectation from this module is high, I think that we will be able to develop applications from python which will be used to run the networking devices. Since python is a versatile programming language (can be used for hacking, developing websites, ethical hacking, developing software and games), I hoped to have a number of working fields and scopes. Having great scope, I thought writing a program using python would be tough for me but due to the format of our college i.e. lecture, tutorial and lab classes, I could understand the module in more precise way. When I started the module, I found python program to be easier than java in many cases, considering the fact that they are best at their own fields.

Considering these expectations, I think we have still a long way to go in order to develop applications. Looking back to the classes and labs that we attended, still these expectations have not been met, it is rude to say this, but honestly speaking till now we have studied the basics of the python programming and recently we learned to make our custom functions, so only this much knowledge is not enough to develop an application. But I think we will reach those heights as the module progresses. It is the matter of time, we have just started and the knowledge we have gathered is also an impressive feat. The best part is the feeling of competition we have while solving the problems given by the lecturers. Slowly and steadily, we are learning and inventing the same program in different ways. It helps us to clear the concept and understand the module. If we can continue our classes in same effective and energetic way, the module can be more interesting. Till now, I am pretty sure that I would meet my expectations if I continue my classes in same energetic manner. The requirements that this module requires from a student is slowly building up.

I am coping with the requirements keeping one mind set i.e. ‘Practice makes a man perfect’. If you practice the programs on your own, then you develop a concept in your mind regarding it. Whenever I practice the program, I try to find the other way to solve it and whenever I do it, I find this programming language more interesting. I take programming as fun, where we play with functions and variables to construct a program we want. I research and watch videos through which my future classes become easier to understand. While doing this activity, I came to know about other sorting algorithm as well. I check google classes provided by the college to learn some new ideas. I revise the lecture classes and try to perform the task related it. Practice is the only thing that can help you to conquer your queries and problems. I manage my time for all the subjects in which most of the time I practice programming. For now, it’s the basic thing we are learning, so I am preparing myself for the advanced study.

Since, we are going through the base, I have not countered any difficulties in this module. Right now, by looking the syntax error, we understand the mistake we have done in the program. So, its quite easy to find out the fault in our program and improve it. May be when we enter the main course, I would face some difficulties but I am positive to get help from my lecturers.

Bibliography:

Anon., 2017. *Bubble Sort Program In C Using Function.* [Online]
Available at: https://codingcompiler.com/bubble-sort-program-in-c-using-function/
[Accessed 20 12 2018].
Sehgal, K., feb 11. *An Introduction to Bubble Sort.* [Online]
Available at: https://medium.com/karuna-sehgal/an-introduction-to-bubble-sort-d85273acfcd8
[Accessed 7 12 2018].