Data Structure and Algorithm

Laboratory Activity No. 2

Algorithm Analysis and Flowchart

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Monoy, Justin Rhey A. | Engr. Maria Rizette H. Sayo |

July, 26, 2025

# Objectives

Introduction

Data structure is a systematic way of organizing and accessing data, and an algorithm is a step-by-step procedure for performing some task in a finite amount of time. These concepts are central to computing, but to be able to classify some data structures and algorithms as “good,” we must have precise ways of analyzing them.

This laboratory activity aims to implement the principles and techniques in:

* Writing a well-structured procedure in programming
* Writing algorithm that best suits to solve computing problems to improve the efficiency of computers
* Convert algorithms into flowcharting symbols

# Methods

* 1. Explain algorithm and flowchart

-x, x<0

x, x ≥ 0

* 1. Write algorithm to find the result of equation: f (x) = and draw its flowchart
  2. Write a short recursive Python function that finds the minimum and maximum values in a sequence without using any loops

# Results

A.

Algorithm  
  
 A programming algorithm is a procedure or formula used for solving a problem. It is based on conducting a sequence of specified actions in which these actions describe how to do something, and your computer will do it exactly that way every time. An algorithm works by following a procedure, made up of inputs. Once it has followed all the inputs, it will see a result, also known as output.  
  
Flowchart

An algorithm, workflow, or process can be represented by a flowchart. The steps are represented by boxes of different types in the flowchart, and arrows are used to connect the boxes to indicate their order. This diagrammatic representation shows a model for solving a particular problem. Flowcharts are used in many different sectors for process or program analysis, design, documentation, and management.

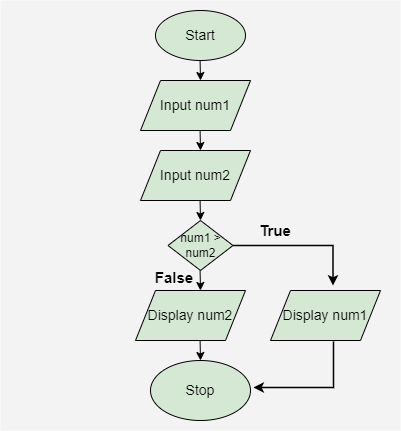


Figure 1. Flow chart

B.

Algorithm

Step 1: Start

Step 2: Input the value of x

Step 3: If x < 0, then f = -x

Step 4: Else f = x

Step 5: Output the value of f

Step 6: End

Flowchart

1. Start

2. Input xIs x < 0?

3. If Yes, go to step 4

4. If No, go to step 5

5. f = -x

6. f = x

7. Output f

9. End

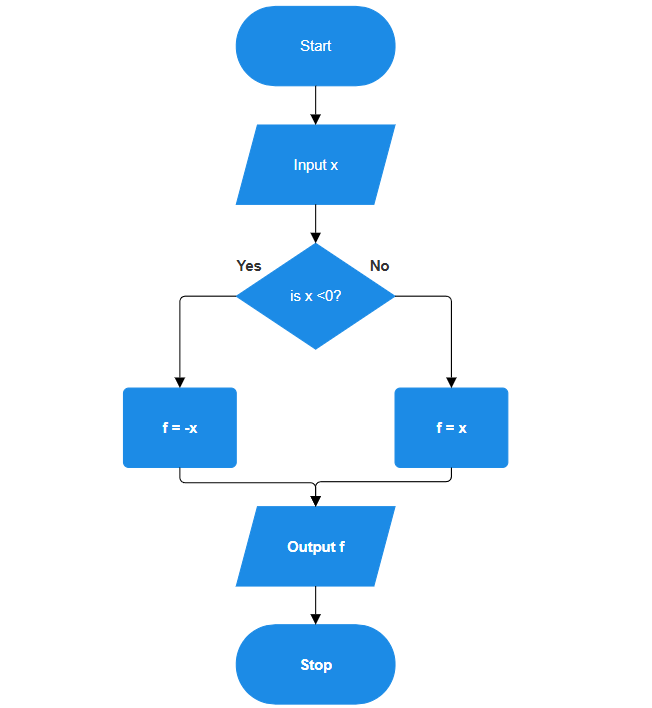


Figure 2. Flow chart

C.

def find\_min\_max(seq):

if len(seq) == 1:

return seq[0], seq[0]

else:

min\_rest, max\_rest = find\_min\_max(seq[1:])

return min(seq[0], min\_rest), max(seq[0], max\_rest)

numbers = [11,9,23,6,5,3]

minimum, maximum = find\_min\_max(numbers)

print("Min:", minimum)

print("Max:", maximum)

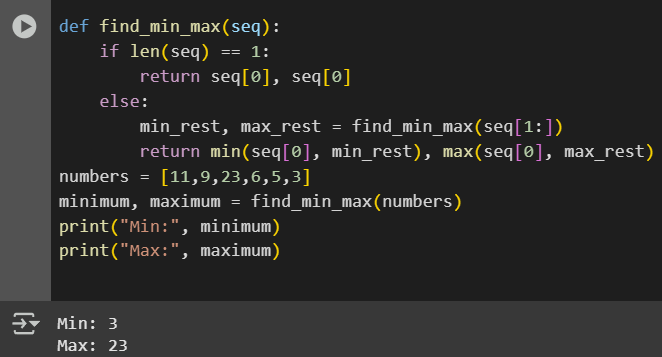


Figure 3. Source code

# Conclusion

Through this laboratory activity, I was able to understand more clearly how algorithms and flowcharts help in solving problems step by step. Writing procedures and translating them into flowcharts made it easier to follow and check the logic. I also found the recursive function part interesting because it showed how problems can be solved without using loops. Overall, this lab helped me improve my thinking when it comes to solving problems in programming, and I learned how important it is to plan before writing actual code.

**References**

[1] “What is a programming algorithm? Data defined - indicative,” Indicative, Sep. 15, 2021. <https://www.indicative.com/resource/programming-algorithm/>

[2] K. L. Busbee, “Flowcharts,” Programming Fundamentals, Dec. 15, 2018. <https://press.rebus.community/programmingfundamentals/chapter/flowcharts/>

[3]“W3Schools.com.” <https://www.w3schools.com/python/gloss_python_function_recursion.asp>