UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 2

Algorithm Analysis and Flowchart

Submitted by:
Luminario, Venice Lou Gabrielle M.

Instructor:
Engr. Maria Rizette H. Sayo

August 01, 2025

I. 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

II. Methods

A. Explain algorithm and flowchart

B. Write algorithm to find the result of equation: $f(x) = \begin{cases} -x, & x<0 \\ x, & x \geq 0 \end{cases}$ and draw its flowchart

C. Write a short recursive Python function that finds the minimum and maximum values in a sequence without using any loops

Explanation of Algorithm and Flowchart in Python

An **algorithm** is a structured sequence of steps used to solve a particular problem or complete a task. In Python programming, an algorithm outlines the logic that can be converted into actual code.

Key Features of an Algorithm:

- **Finiteness**: The steps should eventually come to an end.
- **Clarity**: Each instruction must be clearly defined with no ambiguity.
- **Input**: It can take zero or more input values.
- **Output**: It should generate at least one output result.
- **Effectiveness**: Every step should be simple enough to be performed using basic Python operations.

A **flowchart** visually represents the logic of an algorithm. It's often used in Python development to plan and organize code structure before writing the actual program. Flowcharts use specific symbols to illustrate the type of operation and show how control flows through the program.

Common Flowchart Symbols:

- **Rectangle**: Represents a process or action, like a calculation or a variable assignment.
- **Parallelogram**: Stands for input (e.g., `input()`) or output (e.g., `print()`).
- **Diamond**: Indicates a decision point, such as `if` statements in Python.
- **Oval**: Used to mark the start or end of the flowchart.

III. Results

A. Write algorithm to find the result of equation: f (x) = and draw its flowchart

Algorithm:

1. Start
2. Read Value of x
3. Check if x < 0
4. If YES, set f = -x
5. If NO, set f= x
6. Display value of f
7. End

Flowchart:

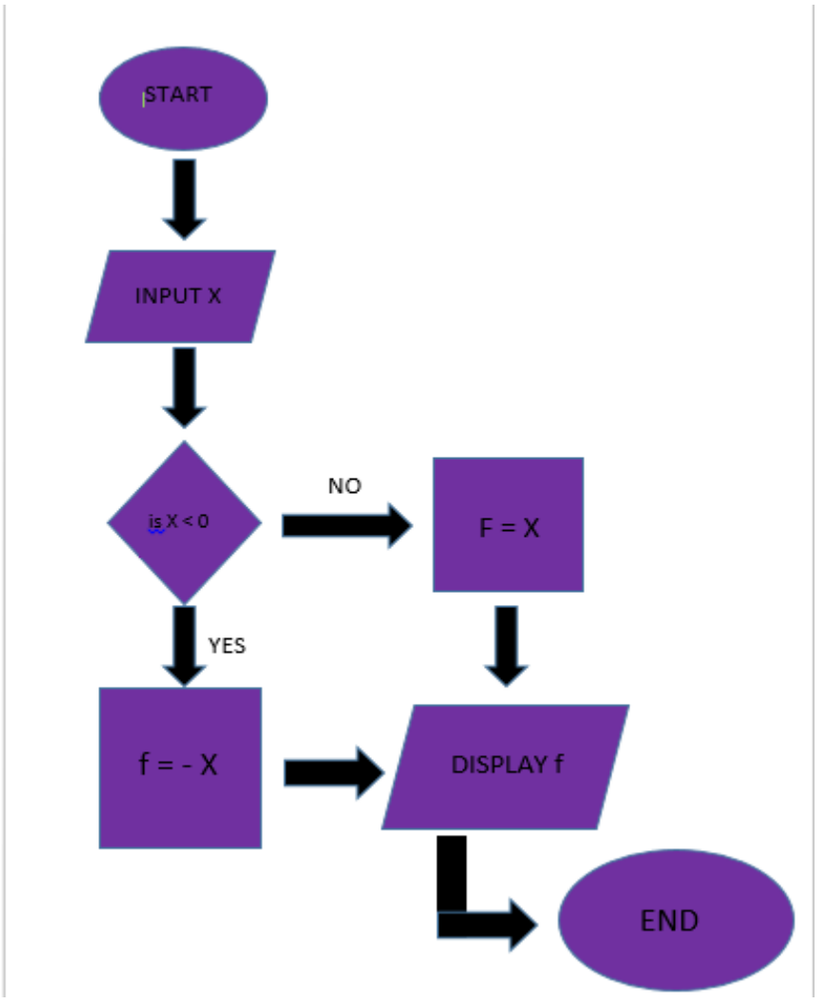

Figure 1: Screenshot of the flowchart of the program

```python
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 = [29, 20, 88, 42, 90, 60]
minimum, maximum = find_min_max(numbers)

print("Min:", minimum)
print("Max:", maximum)
```

```
Min: 20
Max: 90
```

Figure 2: Source code and Output

IV. Conclusion

I struggled initially to make flowcharts because it was applying my “new thinking” in a new place. But I understood better after participating this lab activity, how algorithms and flowcharts facilitate solving a problem in small steps. The process of writing them out and converting into flowcharts made the logic easier to follow. I was particularly interested in the section about recursive function.

References

S. S. Bhamare, S. S. Patil and M. D. Patil, "Understanding Algorithms and Flowcharts: A Comparative Study," *International Journal of Computer Applications*, vol. 176, no. 23, pp. 1–5, June 2020. [Online]. Available: https://www.ijcaonline.org/archives/volume176/number23/31283-2020920164

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