UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 2

Algorithm Analysis and Flowchart

Submitted by:
Enverzo, Kyke Andrey D.

Instructor:
Engr. Maria Rizette H. Sayo

August 02, 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

A. Explain algorithm and flowchart

An **algorithm** is a methodical process or equation used to carry out a task or solve a particular problem. It is written in a human-readable format and subsequently converted into a computer program.

Characteristics of an Algorithm

- **Finiteness**: It must terminate after a finite number of steps.
- **Definiteness**: Each step must be clearly and unambiguously defined.
- **Input**: It should have zero or more inputs.
- **Output**: It must produce at least one output.
- **Effectiveness**: Every step must be basic enough to be carried out.

A **flowchart** is an algorithm that is represented graphically. Various symbols are used to indicate different kinds of instructions, and arrows are used to illustrate how control moves from one step to the next.

Common Flowchart Symbols are:

- **Rectangle -** Process step (e.g., calculations or assignments)
- **Paralelogram** - Input/Output operations
- **Diamond** - Decision-making (e.g., Yes/No questions)
- **Oval** - Start/End

III. Results

B. 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:

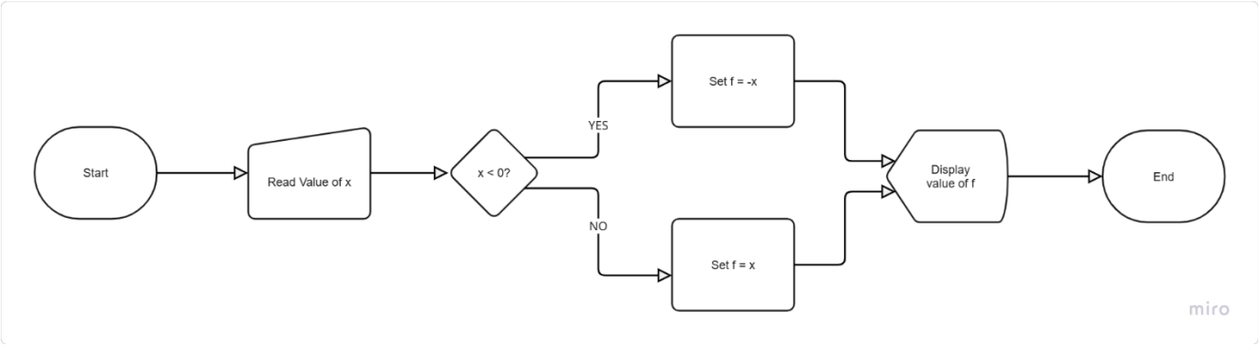

Figure 1: Screenshot of the flowchart of the program

```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 = [2, 21, 7, 4, 2024]
minimum, maximum = find_min_max(numbers)
print("Min:", minimum)
print("Max:", maximum)
```

```
Min: 2
Max: 2024
```

Figure 2: Source code and Output

IV. Conclusion

Since I was still getting used to my new surroundings, I initially found it difficult to create flowcharts during our PLD training. But I now have a better grasp of how algorithms and flowcharts help with problem-solving in a methodical way thanks to this lab exercise. Creating procedures and turning them into flowcharts made each step easier to follow and helped to clarify the reasoning. Particularly intriguing to me was the section on recursive functions, which showed how some issues may be resolved without the need of loops. All in all, this lab improved my ability to solve programming problems and highlighted how crucial careful thought is before creating actual code.

References

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.

A. Downey, *Think Python: How to Think Like a Computer Scientist*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2015.

J. Kleinberg and É. Tardos, *Algorithm Design*, 1st ed. Upper Saddle River, NJ, USA: Pearson, 2006.