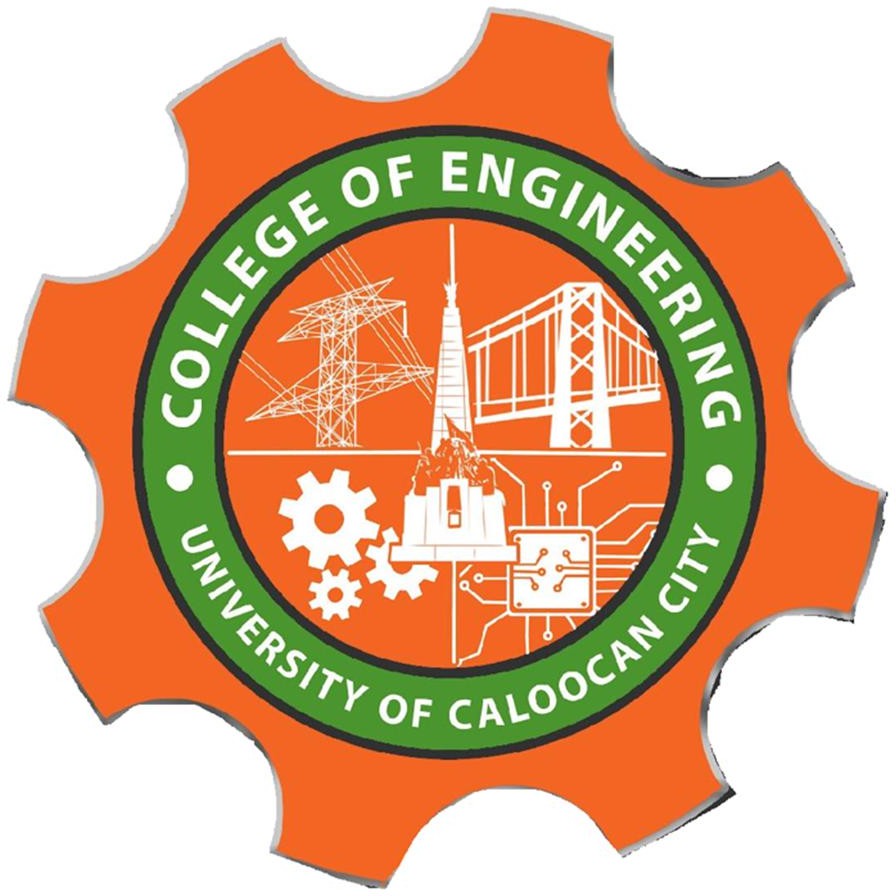
**UNIVERSITY OF CALOOCAN CITY COMPUTER ENGINEERING DEPARTMENT**

## Data Structure and Algorithm

Laboratory Activity No. 2

Algorithm Analysis and Flowchart

*Submitted by: Instructor:*

Calica, Ljay L. Engr. Maria Rizette H. Say

August 01, 2025



DSA UCC

## 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

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

x, x ≥ 0

flowchart

* 1. 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

* An algorithm is a finite sequence of well-defined, computer-implementable instructions to solve a class of problems or perform a computation. Algorithms are essential to computer science as they form the basis for all computational problem-solving.

**Key Characteristics of Effective Algorithms:**

1. **Correctness**: Produces the right output for all valid inputs
2. **Efficiency**: Minimizes resource usage (time and space)
3. **Clarity**: Each step is unambiguous and precisely defined
4. **Robustness**: Handles unexpected inputs gracefully
5. **Termination**: Guaranteed to complete in finite time

A **flowchart** is a type of diagram that represents a workflow or process using standardized symbols connected by arrows to show the direction of process flow.

### Flowchart Symbols and Their Meanings:

* **Terminator (Oval)**: Start/End points
* **Process (Rectangle)**: Any processing function
* **Decision (Diamond)**: Conditional branching
* **Input/Output (Parallelogram)**: Data input or output
* **Flow Lines (Arrows)**: Direction of process flow
* **Connector (Circle)**: Connection between separated parts

## Results

1. Algorithm and Flowchart for Absolute Value Function:

**Problem Statement**: Implement the absolute value function:

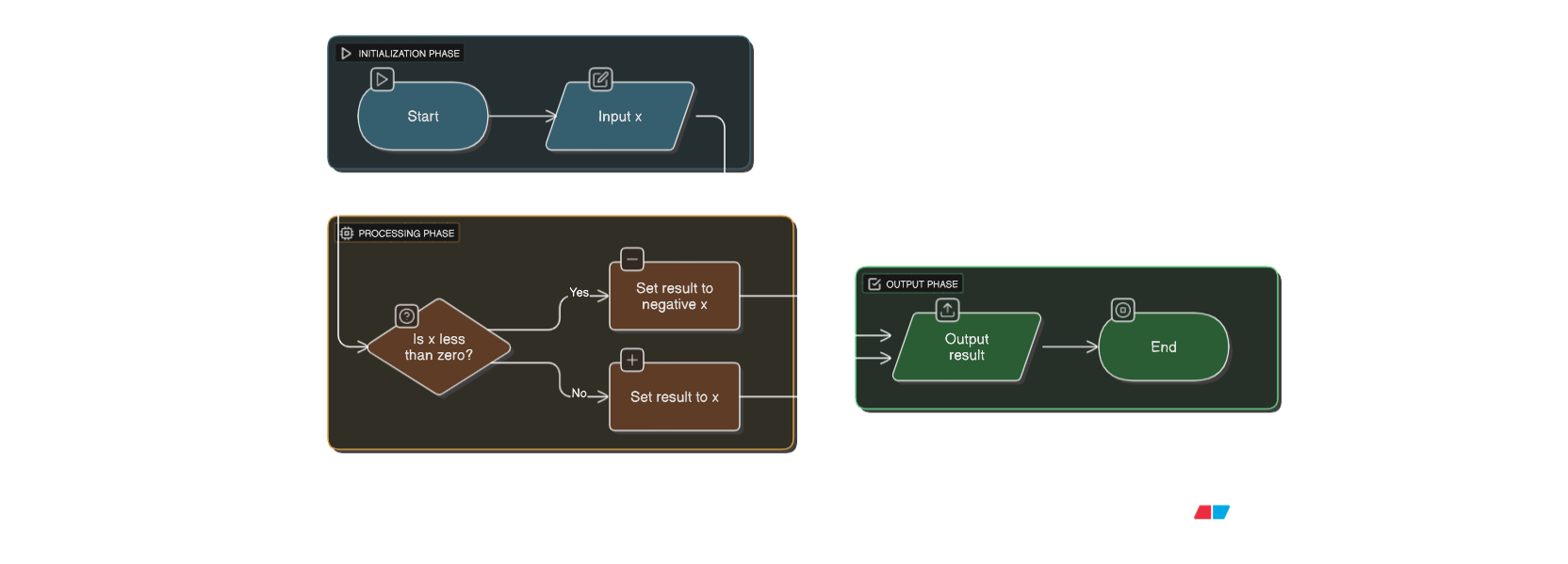
f (x) = { − *x* , x < 0

{ *x ,* ​ x > or equal 0

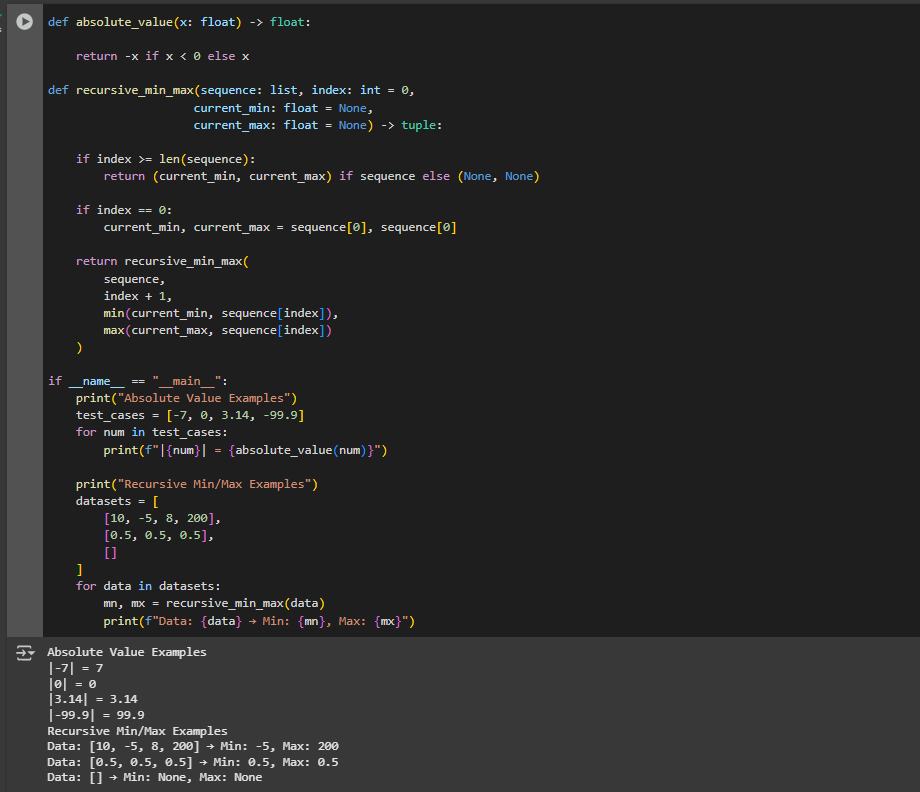
### Algorithm:

1. **Start** the procedure
2. **Input** a numerical value for x
3. **Evaluate** the condition: Is x less than 0?
   * If TRUE: Set result = -x
   * If FALSE: Set result = x
4. **Output** the computed result
5. **End** the procedure

### Flowchart:



*Figure 1: Screenshot of the flowchart of the program*

**

*Figure 2: Source code and Output*

## Conclusion

This lab completely changed how I think about coding. At first, I just wanted to jump straight into writing code, but forcing myself to plan with flowcharts for the absolute value problem saved me so much time. I could actually see my logic before running into errors. The recursive min/max function was tough at first I kept getting lost in all the function calls. But when it finally clicked, I realized recursion is just solving smaller versions of the same problem. It felt like unlocking a secret level in a game! Now I understand why experienced programmers don't just start typing. Whether it's sketching a quick flowchart or breaking problems into recursive cases, planning really does make everything easier. I'm still not perfect at it, but I'm trying to slow down and think before coding.

# References

Cormen, T. H., et al. (2022). *Introduction to Algorithms* (4th ed.). MIT Press.

"Flowchart Design Guidelines". IEEE Computer Society, 2023.

<https://www.computer.org/education/flowcharts>