Data Structure and Algorithm

Laboratory Activity No. 2

Algorithm Analysis and Flowchart

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Poliño, Justine | Engr. Maria Rizette H. Sayo |

August 2 , 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,Algorithms & Flowcharts in Python**

**Algorithm**

A step-by-step problem-solving procedure.

**Key Features:**

* Finiteness - Must terminate after limited steps
* Clarity - Each step is unambiguous
* Input/Output - Processes inputs to generate outputs

**Flowchart**

Visual algorithm representation using symbols:

**Symbols:**

* Oval - Start/End points
* Parallelogram - Input/Output operations
* Rectangle - Processing steps
* Diamond - Decision points

**Absolute Value Example:**

* Start → Input x → Check if x < 0 →
  + Yes: Set f = -x → Output f → End
  + No: Set f = x → Output f → End

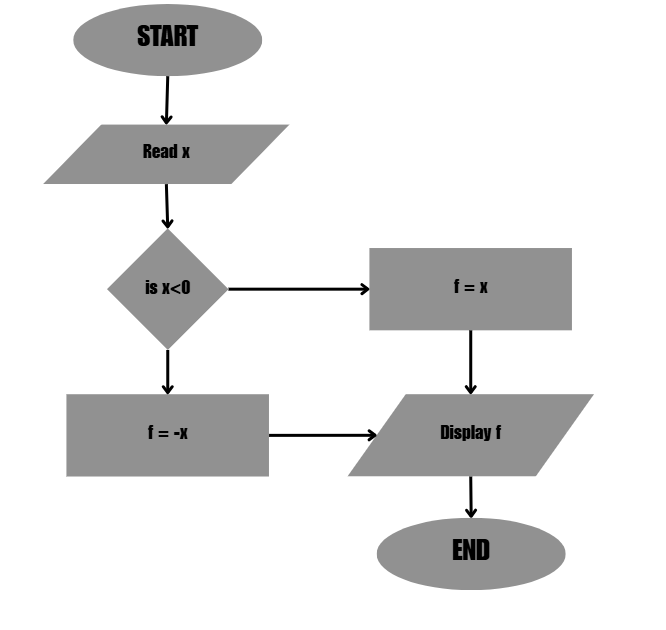
**Purpose:**

* Algorithms define logic precisely
* Flowcharts visualize program flow

**B. ALGORITHM AND FLOWCHART**

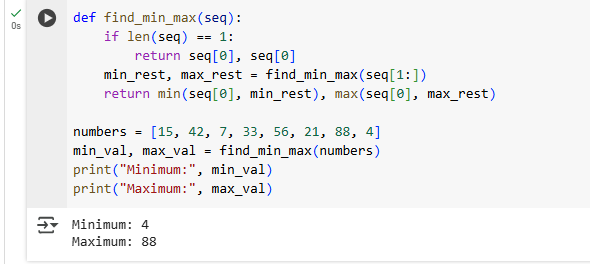
**ALGORITHM**

1. **START**
2. **Input** the value of x
3. **Check condition**:
   * If x < 0:
     + Set f = -x
   * Else:
     + Set f = x
4. **Display** the value of f
5. **END**

**FLOWCHART**   


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

**C. Find the minimum and maximum values in a sequence without using any loops.**

****

*Figure 2: Source code and Output*

# Conclusion

This laboratory activity provided valuable insights into the core concepts of algorithm design and flowchart development in Python programming. Through practical implementation, we achieved three key objectives: (1) constructing a functional algorithm to calculate absolute values through systematic logical steps, (2) accurately converting this algorithm into a visual flowchart representation, and (3) creating an efficient recursive solution for identifying minimum and maximum values without iterative loops. The exercise highlighted the complementary nature of algorithms and flowcharts, while algorithms offer precise, step-by-step computational instructions, flowcharts provide an intuitive visual framework for understanding program flow and decision logic. Initial difficulties in flowchart creation underscored the importance of methodical problem decomposition in programming tasks. These fundamental skills serve as critical building blocks for developing optimized solutions to more complex computational challenges. The recursive implementation particularly demonstrated how alternative programming paradigms can yield elegant solutions to specific problem types. Future work could focus on performance analysis and optimization techniques for handling larger datasets while maintaining code clarity and efficiency.

**References**

[1] G. Drozdek, Data Structures and Algorithms in Python, 1st ed. Boston: Cengage, 2022.