



UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT



Data Structure and Algorithm

Laboratory Activity No. 3

Translating Algorithm to Program

Submitted by:
Disomnong, Jalilah M.

Instructor:
Engr. Maria Rizette H. Sayo

08-02-2025

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 tasks 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
- Writing an efficient Python program from translated algorithms

II. Methods

- Design an algorithm and the corresponding flowchart (Note: You may use LucidChart or any application) for adding the test scores as given below if the number is even: 26,49,98,87,62,75
- Translate the algorithm to a Python program (using Google Colab)
- Save your source codes to GitHub

III. Results

Algorithm

Step 1: Start

Step 2: Input test scores list: 26, 49, 98, 87, 62, 75

Step 3: If first test score is even add to sum

else if test score is odd proceed to next test score

If next test score is even add to sum

else if test score is odd proceed to next test score

If next test score is even add to sum

else if test score is odd proceed to next test score

Step 4: Print Sum of the even scores

Step 5: End

The algorithm begins by initializing the process and accepting a predefined list of test scores: 26, 49, 98, 87, 62, and 75. It then checks each score one by one to determine whether it is even. Starting with the first score, 26, the algorithm verifies that it is an even number and adds it

to the cumulative sum. The next score, 49, is identified as odd, so it is skipped, and the algorithm proceeds to the following number. It encounters 98, which is even, so it is added to the sum as well. Continuing the process, the algorithm skips 87 because it is odd and adds 62 to the sum because it is even. Finally, it checks 75, which is odd, and skips it. After all elements have been evaluated, the algorithm prints the total sum of the even scores.

Flowchart

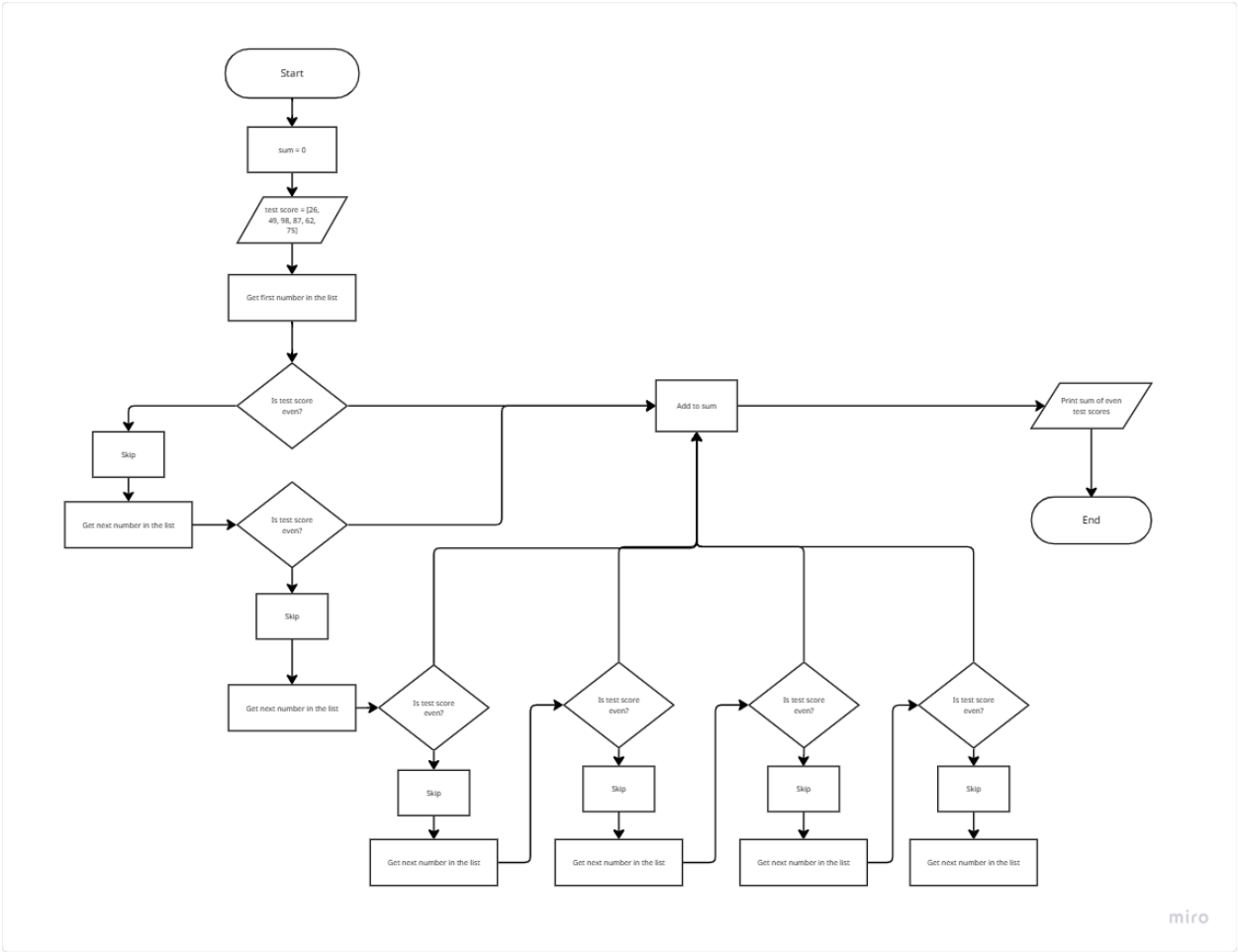
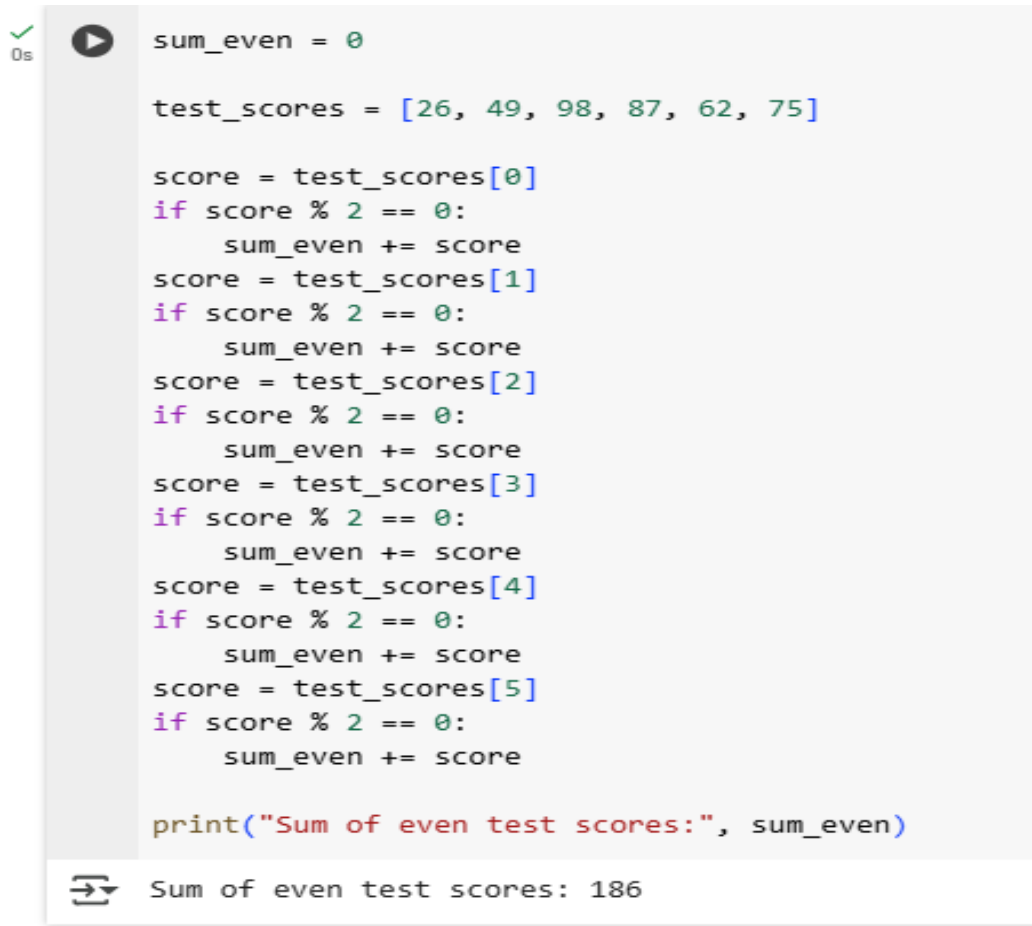


Figure 1 Flowchart

The flowchart illustrates a simple algorithm designed to calculate the sum of even numbers from a predefined list of test scores. It begins with the start of the process and proceeds to input the list of scores. Each score is then evaluated individually through a series of decision points that determine whether the number is even or odd. If the score is even, it is added to a running total; if it is odd, the algorithm skips it and moves to the next score. This process repeats until all scores in the list have been checked. Finally, the flowchart concludes with the output of the total sum of even scores and marks the end of the algorithm.

Source Code



```
sum_even = 0

test_scores = [26, 49, 98, 87, 62, 75]

score = test_scores[0]
if score % 2 == 0:
    sum_even += score
score = test_scores[1]
if score % 2 == 0:
    sum_even += score
score = test_scores[2]
if score % 2 == 0:
    sum_even += score
score = test_scores[3]
if score % 2 == 0:
    sum_even += score
score = test_scores[4]
if score % 2 == 0:
    sum_even += score
score = test_scores[5]
if score % 2 == 0:
    sum_even += score

print("Sum of even test scores:", sum_even)
```

Sum of even test scores: 186

Figure 2 Screenshot of source code link:https://colab.research.google.com/drive/1k8wFdMrhElr-FBpSEkfVs_B6ffddAucE#scrollTo=9cMbpNSVSYvV

In figure 2, the program checks whether each score is even by using the modulus operator (%). If a score is divisible by 2 (i.e., even), it is added to a running total called sum_even. This process is repeated for all six scores using sequential if statements without loops.

IV. Conclusion

In this laboratory activity, we performed an activity on how to apply the basic concepts of algorithms, flowcharts, and source code in solving a problem. We started by creating an algorithm that shows the step-by-step process for adding even numbers from a list of test scores. Then, we used a flowchart to visualize the flow of the algorithm, which made the logic easier to follow. Finally, we translated the algorithm into a Python program, showing how source code is written based on a planned solution. Through this activity, we learned how important it is to first plan and understand the logic before writing the actual code.

References

[1] Google. (2024). Google Colaboratory. Retrieved April 18, 2024, from <https://colab.research.google.com/>