Practical - Introduction & Revisiting Arrays and Files

Updated: 10 February 2025

Objective

This practical session serves as an introduction to the Data Structures and Algorithms unit. During this session, students will revisit the fundamental concepts of arrays and file handling, exploring their applications in problem-solving.

There are two hands-on activities designed for this session, where students will work in pairs to practice operations using Python or Java. No submission is required for this session. These activities will allow students to apply their knowledge of arrays and file manipulation to solve real-world problems, reinforcing key concepts while enhancing their problem-solving and programming skills.

Note:
Java Students: Please adhere to the Java Coding Standard document located under *Links and Resources on Blackboard.*

Python Students: Please adhere to the PEP8 Coding Standard.

Activities

1. Imagine you are working as a data analyst for a weather monitoring system. The system collects daily temperature data for a city. You need to find the hottest and coldest days of the week based on the temperature readings. Given an array of recorded temperatures, write a program to determine the maximum (hottest day) and minimum (coldest day) temperature.

Example Input & Output:
Input: [22, 30, 25, 28, 35, 31, 27]
Output: Max Temperature: 35°C, Min Temperature: 22°C

Pseudocode:

1. Define a function to find maximum and minimum elements in an array:
   - INPUT: An array of numbers (temperature readings)
   - OUTPUT: Maximum and Minimum values in the array

2. Initialize two variables:
   - max_value ← first element of array
   - min_value ← first element of array

3. Iterate through each element in the array:
   a. IF current element > max_value THEN
      - Update max_value ← current element
   b. IF current element < min_value THEN
      - Update min_value ← current element

4. Return max_value and min_value

2. In real-world applications, data is often stored in files rather than being hardcoded in the program because files can handle large amounts of data efficiently. Files make it possible to store multiple entries of temperature data, especially when the data is generated over a long period (e.g., over months or years). Now imagine, you are working for a climate monitoring agency that collects daily temperature data. The data is typically recorded in a file (temperatures.csv) for later processing. Your task is to read the temperature data from the file, process it, and determine the hottest and coldest days of the year.

Example Input & Output:

Input: temperatures_365_days.csv

Output: Hottest day: 1/05/2024 with temperature 41.0°C

Coldest day: 27/01/2024 with temperature 15.0°C

Pseudocode:

```
BEGIN

  FUNCTION Main
    SET filePath = "temperatures_365_days.csv"
    DECLARE dates[365]  // Array to store the dates
    DECLARE temperatures[365]  // Array to store the temperatures
    CALL readTemperatureData(filePath, dates, temperatures)
    CALL findExtremeDays(dates, temperatures)
    PRINT "Hottest day: " + maxDay + " with temperature " + maxTemp + "°C"
    PRINT "Coldest day: " + minDay + " with temperature " + minTemp + "°C"
  END

  FUNCTION readTemperatureData(filePath, dates, temperatures)
    OPEN file at filePath for reading
    SKIP header line
    SET i = 0  // Index for the arrays
    WHILE there are more lines in the file
      READ a line
      SPLIT the line by ',' to get date and temperature
      SET dates[i] = date  // Store date in the array
      SET temperatures[i] = temperature  // Store temperature in the array
      INCREMENT i by 1  // Move to the next index
    END WHILE
  END

  FUNCTION findExtremeDays(dates, temperatures)
    SET maxTemp = temperatures[0]  // Initialize maxTemp with the first
temperature
    SET minTemp = temperatures[0]  // Initialize minTemp with the first
temperature
    SET maxDay = dates[0]  // Initialize maxDay with the first date
    SET minDay = dates[0]  // Initialize minDay with the first date

    FOR i = 1 to 364  // Loop through the remaining temperatures
      IF temperatures[i] > maxTemp
        SET maxTemp = temperatures[i]
        SET maxDay = dates[i]
      END IF
      IF temperatures[i] < minTemp
        SET minTemp = temperatures[i]
        SET minDay = dates[i]
      END IF
    END FOR

    PRINT "Hottest day: " + maxDay + " with temperature " + maxTemp + "°C"
    PRINT "Coldest day: " + minDay + " with temperature " + minTemp + "°C"
  END

END
```

End of Worksheet