Week 2

Ideas:

- Using a **physics experiment** with a lot of numerical data that needs to be processed (for example radiation experiment or measurements of the same thing which would create an opportunity for calculations of standard deviations and averages etc), create a program which takes in a number of measurements and calculates certain statistical results.
- E-commerce or inventory analysis: Using a dataset from an ecommerce website (or Kaggle) students could leverage loops to analyse trends over time, like best-selling products each month or how inventory levels change over specific time periods.
- Weather trend analysis: Given historical weather data from multiple cities, students can apply iteration to determine patterns such as the hottest/coldest months in each city, average rainfall in specified periods, or the frequency of certain weather events.

Iteration: Physics Experiment (Radiation Exposure Analysis)

[Case Study Story] --> Dr. Eleanor, a renowned physicist, is conducting an experiment to measure the average radiation levels in various environments with high amounts of radioactive waste. Over several weeks, she collects a vast amount of data from different locations, ranging from urban areas to dense forests. She also takes measurements at varying distances from the known sources of the radiation so she is able to account for any background or ambient, naturally occurring radiation. Each location has multiple measurements taken at different times of the day to account for variations.

To process this extensive data, Dr. Eleanor needs a program that can efficiently handle the numerous measurements. The program should allow her to input the radiation measurements for each location and then calculate essential statistical insights. These insights include the average radiation level, the standard deviation to understand the variability, and other relevant metrics that can help her identify patterns or anomalies in the data.

Given the repetitive nature of the data input and the calculations required for each set of measurements, the program must utilise loops effectively. For instance, a *for* loop can iterate over each location's data set, while nested loops can process individual measurements within those sets. Additionally, the program

should provide an option for Dr. Eleanor to continue inputting data until she decides to stop, making use of a *while* loop.

Topics to be consolidated

- Fundamentals of *for* loops in Python.
- Designing nested *for* loops for multi-layered data processing.
- Using trace tables to monitor variable values during iterations.
- Advantages of loop structures in reducing code redundancy and errors.
- Differentiating between *for* and *while* loops and their applications.
- Implementing '*break*' and '*continue*' within loops for enhanced control.
- Practical applications of loops: processing extensive data sets and calculating statistical metrics.
- Debugging techniques within loop structures using IDE tools.