Assignment 1a: Student Marks Analysis (Excel Automation)

You are given student marks data in the following format. The data is stored in Excel with Headers StudentID, Name, Math Physics Chemistry and Biology.

```
"StudentID": [101, 102, 103, 104, 105, 106, 107],
"Name": ["Alice", "Bob", "Charlie", "David", "Emma", "Frank", "Grace"],
"Math": [95, 72, 88, 55, 80, 99, 68],
"Physics": [89, 65, 91, 62, 77, 95, 72],
"Chemistry": [92, 70, 85, 58, 79, 97, 74],
"Biology": [88, 60, 90, 61, 83, 96, 70]
```

Tasks:

1. **Load the Data**
   - Convert the above dictionary into a Pandas DataFrame.
2. **Vectorized Computations (No Loops Allowed)**
   - Using **NumPy vectorized operations**:
     - Compute **total score** for each student.
     - Compute **average score** for each student.
     - Assign a **grade** based on the average:
       - A: avg ≥ 90
       - B: 75 ≤ avg < 90
       - C: 60 ≤ avg < 75
       - F: avg < 60
3. **Find Top Performers (per subject)**
   - Identify the **top 3 students** in each subject (Math, Physics, Chemistry, Biology).
4. **Data Visualization**
   - Plot a **bar chart** showing the **average marks per subject** across all students.
5. **Save Results Back to Excel**
   - Create a new Excel file results.xlsx with two sheets:
     - **Summary:** StudentID, Name, Total, Average, Grade
     - **Top Performers:** List of top 3 per subject

Input: Excel Worksheet (student.xlsx)

Output: Excel Worksheet (results.xlsx) which is created by Python Code, along with bar chart in excel dynamically created by code.

Note: Need to keep both worksheets in the same workbook.

Assignment 1b: Polygon Geometry using Vector Algebra

In **Architecture CAD Drawings**, polygons represent **land plots, obstacles, or regions**.
Your task is to compute **geometric properties** of a polygon using **vector algebra** (dot product, cross product, norms, etc.).

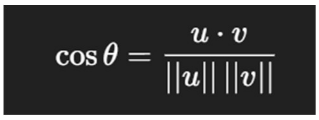
Input Data

You are given a list of polygon vertices (in order): (Hint create polygon using Shapely Library)

(9.05, 7.76) (12.5, 3.0) (10.0, 0.0)(5.0, 0.0)(2.5, 3.0)
Use this sample input for the program Or any other polygon vertices of your choice.

Tasks

- Represent polygon edges as vectors between consecutive vertices.
- Compute the polygon's area using the **shoelace formula / 2D cross product** and compare with **shapely.geometry.Polygon.**
- Compute the **length of each edge** using vector norms.
- For each vertex, compute the **interior angle** using the dot product formula:



$$\cos\theta = \frac{u \cdot v}{||u||\,||v||}$$

Verify if the polygon is **convex** (all angles < 180°).

- Compute the **centroid** (average of vertices) and compare with **shapely.Polygon.centroid.**
- Visualize the polygon using matplotlib:
    - Fill the polygon.
    - Label the vertices.
    - Mark the centroid with a red dot.
    - Annotate each interior angle.

Expected Output

The program should display in the **console**:

- Polygon Area: <value>
- Edge Lengths: [L1, L2, L3, ...]
- Interior Angles (degrees): [A1, A2, A3, ...]
- Is Convex: True/False
- Centroid: (x, y)

And produce a **Matplotlib plot** with:

- Polygon filled with light color.
- Vertices labeled (V1, V2, ...).
- Centroid marked in red.
- Angles annotated near vertices.

Assignment 2: Room Tiling with Squares (Spiral Fill Visualization)

You are given a **room of size M × N** and four types of square tiles:

- 1×1 → Red
- 2×2 → Blue
- 3×3 → Yellow
- 4×4 → Green

Tasks:

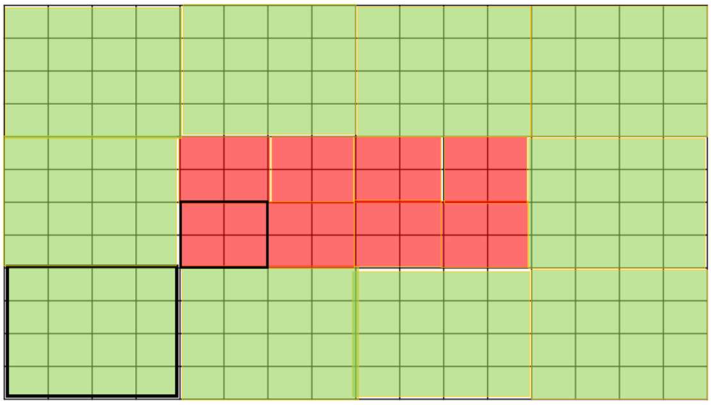
1. Fill the room using the **minimum number of tiles** (always try to place bigger tiles first).
2. The tiling should proceed in a **circular/spiral motion** starting from the center.
3. If after placing larger tiles, there is some **empty space left in the center**, that space must be filled **only with 1×1 tiles**.
4. Visualize the result with matplotlib (different colors for each tile size).

Input:

- Given width and height of room (dimensions in x and y direction)
- The sizes of the tiles are fixed, its internal input for program.

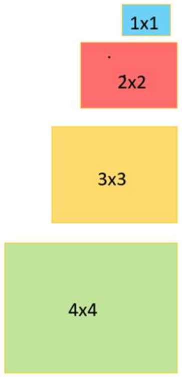
Output:

- A visual plot of the room showing all tiles along with number of tiles needs per size for a given room.



Room filled with minimum number of tiles

Tile sizes



Assignment 3: Bin Packing in Container (Row wise Placement)

You are given a **rectangular container** of size W × H and a list of **bins** (rectangles with width and height). Each bin has a **unique ID**.

Tasks:

1. Place bins **row by row** inside the container:
   - Left → Right,
   - then move to next row (Bottom → Top).
2. Only bins that **fit completely** in the container are placed.
3. If some bins remain unplaced, that is acceptable.
4. Every bin must be visualized with its **unique ID label**. (hint: Use python dictionary to use IDs as keys and rectangle bins as values.)

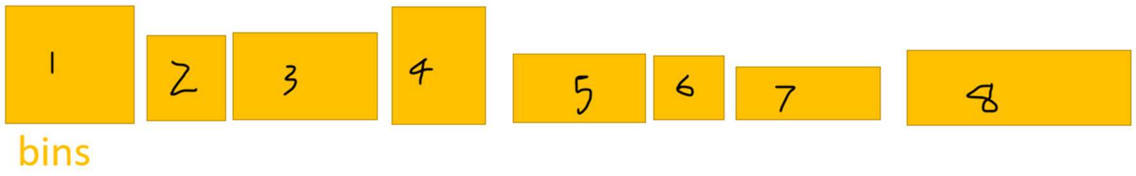
Input:

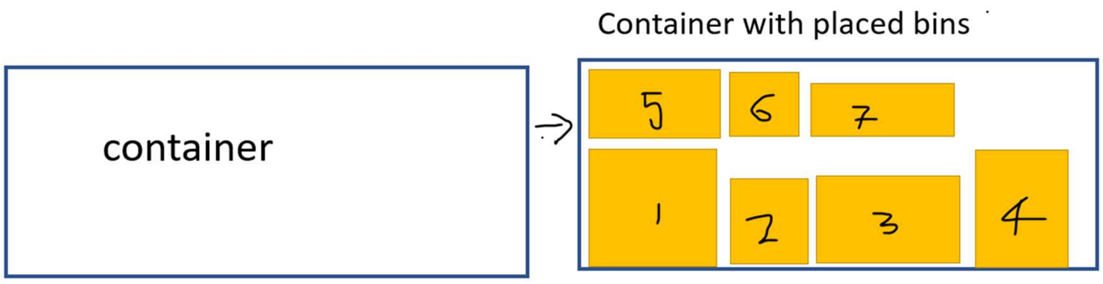
Given width and height of rectangle container (dimensions in x and y direction)

A set of rectangular bins with its width and height with their unique ids. Each bin will have width and height and Left-Bottom coordinate positions.

Expected Output

- A matplotlib visualization showing container outline and random colored bins inside and ids of the bins.
- A printed list of placed bin IDs and unplaced IDs.


1
2
3
4
5
6
7
8
bins


Container with placed bins
container
5
6
7
1
2
3
4