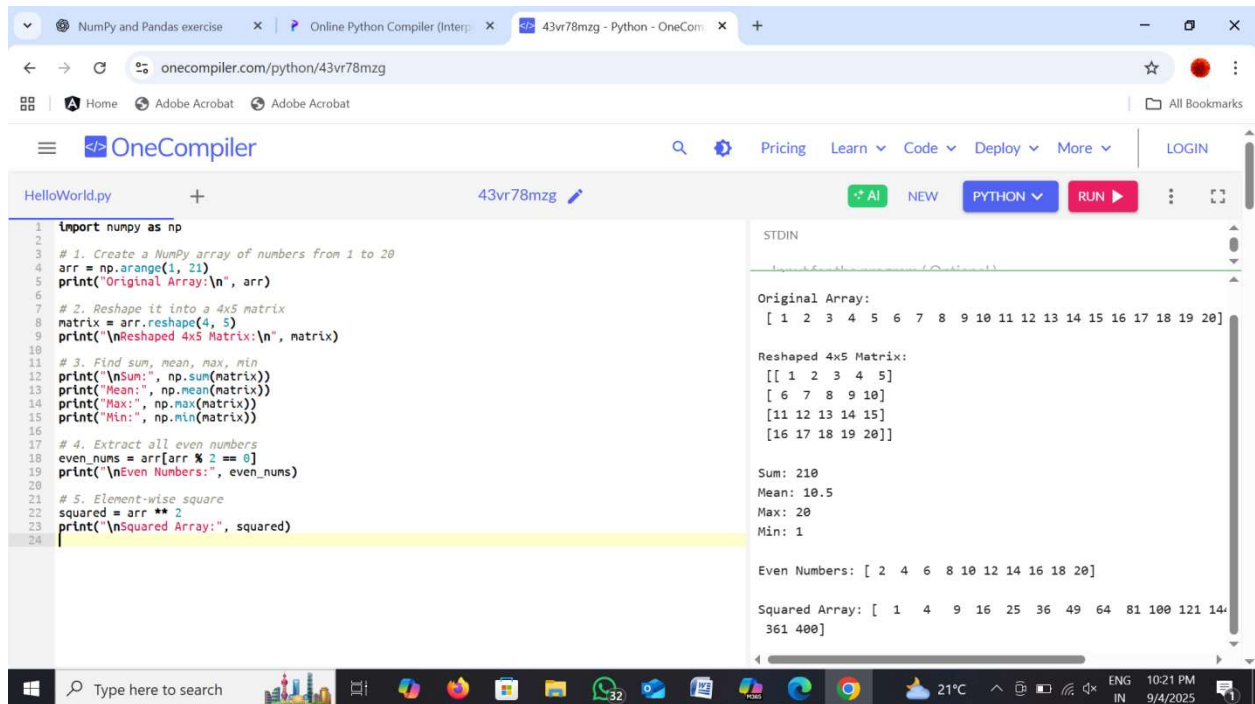


## Week 3: NumPy and Pandas for Data

Hands-On: Perform operations with NumPy and manipulate datasets with Pandas.



The screenshot shows the OneCompiler Python IDE with a file named 'HelloWorld.py'. The code performs several NumPy operations: creating an array from 1 to 20, reshaping it into a 4x5 matrix, calculating sum, mean, max, and min, extracting even numbers, and squaring each element. The output on the right shows the original array, the reshaped matrix, the calculated statistics, the even numbers, and the squared array.

```
1 import numpy as np
2
3 # 1. Create a NumPy array of numbers from 1 to 20
4 arr = np.arange(1, 21)
5 print("Original Array:\n", arr)
6
7 # 2. Reshape it into a 4x5 matrix
8 matrix = arr.reshape(4, 5)
9 print("\nReshaped 4x5 Matrix:\n", matrix)
10
11 # 3. Find sum, mean, max, min
12 print("\nSum:", np.sum(matrix))
13 print("Mean:", np.mean(matrix))
14 print("Max:", np.max(matrix))
15 print("Min:", np.min(matrix))
16
17 # 4. Extract all even numbers
18 even_nums = arr[arr % 2 == 0]
19 print("\nEven Numbers:", even_nums)
20
21 # 5. Element-wise square
22 squared = arr ** 2
23 print("\nSquared Array:", squared)
24
```

Original Array:  
[ 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20]

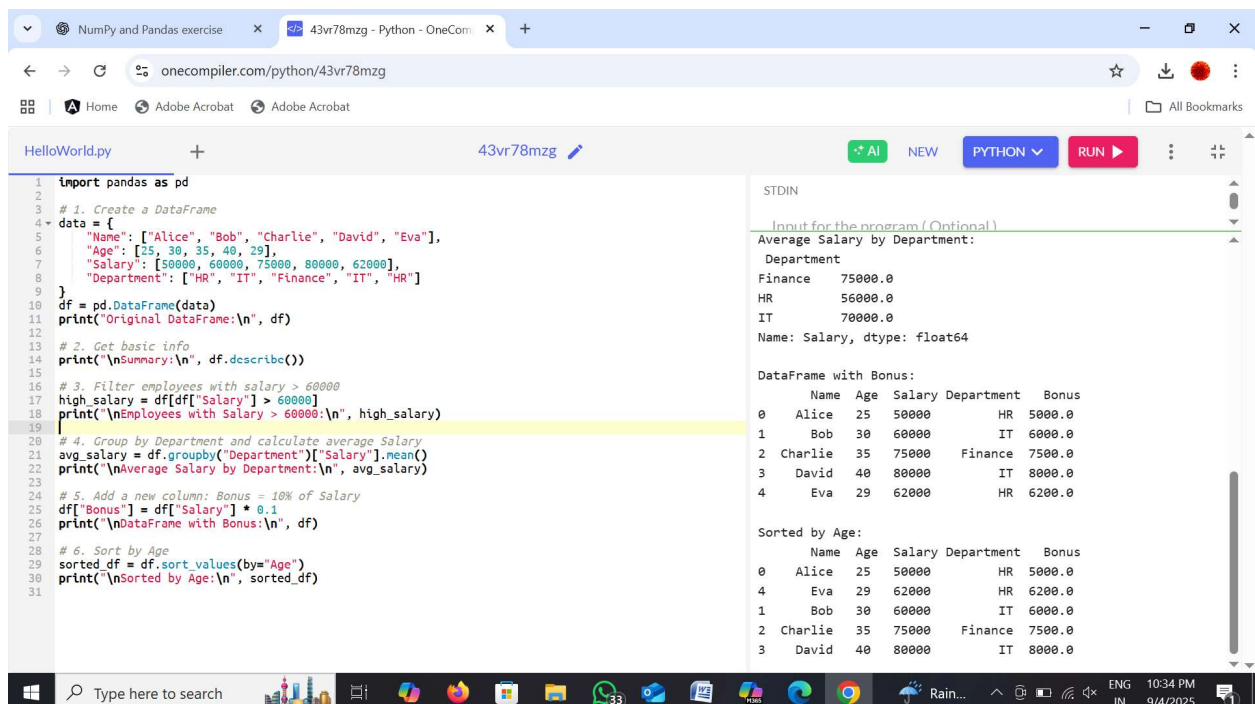
Reshaped 4x5 Matrix:  
[[ 1 2 3 4 5]  
[ 6 7 8 9 10]  
[11 12 13 14 15]  
[16 17 18 19 20]]

Sum: 210  
Mean: 10.5  
Max: 20  
Min: 1

Even Numbers: [ 2 4 6 8 10 12 14 16 18 20]

Squared Array: [ 1 4 9 16 25 36 49 64 81 100 121 144 169 196 225 256 289 324 361 400]

Client Project: Clean and aggregate a dataset (e.g., remove missing values, calculate averages).



The screenshot shows the OneCompiler Python IDE with a file named 'HelloWorld.py'. The code performs several Pandas operations: creating a DataFrame with employee data, getting basic info, filtering employees with salary > 60000, grouping by department to calculate average salary, adding a new column 'Bonus' (10% of salary), and sorting by age. The output on the right shows the average salary by department, the DataFrame with bonus, and the sorted DataFrame.

```
1 import pandas as pd
2
3 # 1. Create a DataFrame
4 data = {
5     "Name": ["Alice", "Bob", "Charlie", "David", "Eva"],
6     "Age": [25, 30, 35, 40, 29],
7     "Salary": [50000, 60000, 75000, 80000, 62000],
8     "Department": ["HR", "IT", "Finance", "IT", "HR"]
9 }
10 df = pd.DataFrame(data)
11 print("Original DataFrame:\n", df)
12
13 # 2. Get basic info
14 print("\nSummary:\n", df.describe())
15
16 # 3. Filter employees with salary > 60000
17 high_salary = df[df["Salary"] > 60000]
18 print("\nEmployees with Salary > 60000:\n", high_salary)
19
20 # 4. Group by Department and calculate average salary
21 avg_salary = df.groupby("Department")["Salary"].mean()
22 print("\nAverage Salary by Department:\n", avg_salary)
23
24 # 5. Add a new column: Bonus = 10% of Salary
25 df["Bonus"] = df["Salary"] * 0.1
26 print("\nDataFrame with Bonus:\n", df)
27
28 # 6. Sort by Age
29 sorted_df = df.sort_values(by="Age")
30 print("\nSorted by Age:\n", sorted_df)
31
```

Average Salary by Department:

Department	Average Salary
Finance	75000.0
HR	56000.0
IT	70000.0

DataFrame with Bonus:

	Name	Age	Salary	Department	Bonus
0	Alice	25	50000	HR	5000.0
1	Bob	30	60000	IT	6000.0
2	Charlie	35	75000	Finance	7500.0
3	David	40	80000	IT	8000.0
4	Eva	29	62000	HR	6200.0

Sorted by Age:

	Name	Age	Salary	Department	Bonus
0	Alice	25	50000	HR	5000.0
4	Eva	29	62000	HR	6200.0
1	Bob	30	60000	IT	6000.0
2	Charlie	35	75000	Finance	7500.0
3	David	40	80000	IT	8000.0

Output:

Original DataFrame:

	Name	Age	Salary	Department
0	Alice	25	50000	HR
1	Bob	30	60000	IT
2	Charlie	35	75000	Finance
3	David	40	80000	IT
4	Eva	29	62000	HR

Summary:

	Age	Salary
count	5.000000	5.000000
mean	31.800000	65400.000000
std	5.80517	12074.767078
min	25.000000	50000.000000
25%	29.000000	60000.000000
50%	30.000000	62000.000000
75%	35.000000	75000.000000
max	40.000000	80000.000000

Employees with Salary > 60000:

	Name	Age	Salary	Department
2	Charlie	35	75000	Finance
3	David	40	80000	IT
4	Eva	29	62000	HR

Average Salary by Department:

Department	
Finance	75000.0
HR	56000.0
IT	70000.0

Name: Salary, dtype: float64

DataFrame with Bonus:

	Name	Age	Salary	Department	Bonus
0	Alice	25	50000	HR	5000.0
1	Bob	30	60000	IT	6000.0
2	Charlie	35	75000	Finance	7500.0
3	David	40	80000	IT	8000.0
4	Eva	29	62000	HR	6200.0

Sorted by Age:

	Name	Age	Salary	Department	Bonus
0	Alice	25	50000	HR	5000.0
4	Eva	29	62000	HR	6200.0
1	Bob	30	60000	IT	6000.0
2	Charlie	35	75000	Finance	7500.0
3	David	40	80000	IT	8000.0