

Assignment – Module 5

1.Create a Java class with user defined exception handling?

Code:-

```
class AgeException extends Exception {
    public AgeException(String message) {
        super(message);
    }
}

// Main class
public class CustomExceptionExample {
    public static void main(String[] args) {
        try {
            validateAge(15); // This will throw the exception
        } catch (AgeException e) {
            System.out.println("Caught the exception: " + e.getMessage());
        }
    }

    // Method that throws the custom exception
    public static void validateAge(int age) throws AgeException {
        if (age < 18) {
            throw new AgeException("Age must be 18 or above.");
        }
        System.out.println("Age is valid.");
    }
}
```

Output:-

```
11 public static void main(String[] args) {
12     try {
13         validateAge(15); // This will throw the exception
14     } catch (AgeException e) {
15         System.out.println("Caught the exception: " + e.getMessage());
16     }
17 }
18
19 // Method that throws the custom exception
20 public static void validateAge(int age) throws AgeException {
21     if (age < 18) {
22         throw new AgeException("Age must be 18 or above.");
23     }
24     System.out.println("Age is valid.");
25 }
```

<terminated> CustomException [Java Application] C:\Users\tapan.k\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.1
Caught the exception: Age must be 18 or above.

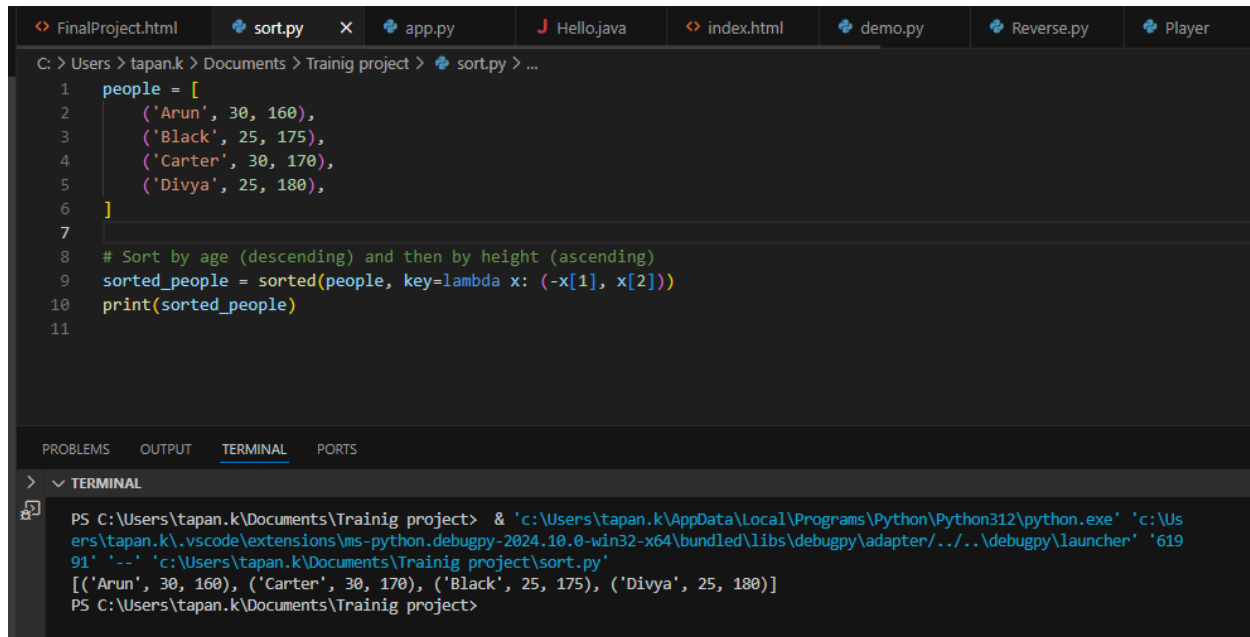
2.Modify below sorted list of user with name, age and height such that age can be descending and height as ascending using python

```
“people = [  
(‘Arun’, 30, 160),  
(‘Black’, 25, 175),  
(‘Carter’, 30, 170),  
(‘Divya’, 25, 180),  
]  
# Sort by age (ascending) and then by height (descending)  
sorted_people = sorted(people, key=lambda x: (x[1], -x[2]))  
print(sorted_people)”
```

Code:-

```
people = [  
    ('Arun', 30, 160),  
    ('Black', 25, 175),  
    ('Carter', 30, 170),  
    ('Divya', 25, 180),  
]  
  
# Sort by age (descending) and then by height (ascending)  
sorted_people = sorted(people, key=lambda x: (-x[1], x[2]))  
print(sorted_people)
```

Output:-



The screenshot shows a VS Code editor with a file explorer at the top displaying several files: FinalProject.html, sort.py, app.py, Hello.java, index.html, demo.py, Reverse.py, and Player. The main editor area shows the content of sort.py, which defines a list of people and sorts them by age (descending) and then by height (ascending). The terminal at the bottom shows the command to run the script and the resulting output.

```
C: > Users > tapan.k > Documents > Trainig project > sort.py > ...  
1  people = [  
2      ('Arun', 30, 160),  
3      ('Black', 25, 175),  
4      ('Carter', 30, 170),  
5      ('Divya', 25, 180),  
6  ]  
7  
8  # Sort by age (descending) and then by height (ascending)  
9  sorted_people = sorted(people, key=lambda x: (-x[1], x[2]))  
10 print(sorted_people)  
11  
PROBLEMS  OUTPUT  TERMINAL  PORTS  
> TERMINAL  
PS C:\Users\tapan.k\Documents\Trainig project> & 'c:\Users\tapan.k\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\tapan.k\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundle\libs\debugpy\adapter\..\..\debugpy\launcher' '61991' '--' 'c:\Users\tapan.k\Documents\Trainig project\sort.py'  
[('Arun', 30, 160), ('Carter', 30, 170), ('Black', 25, 175), ('Divya', 25, 180)]  
PS C:\Users\tapan.k\Documents\Trainig project>
```

3.Implement quick sort and display sorted values for [7,6,10,5,9,2,1,15,7] using java or python?

Code:-

```
import java.util.Arrays;
```

```
public class QuickSortExample {  
    public static void main(String[] args) {  
        int[] arr = {7, 6, 10, 5, 9, 2, 1, 15, 7};  
        quickSort(arr, 0, arr.length - 1);  
        System.out.println(Arrays.toString(arr));  
    }  
}
```

```
public static void quickSort(int[] arr, int low, int high) {  
    if (low < high) {  
        int pi = partition(arr, low, high);  
        quickSort(arr, low, pi - 1);  
        quickSort(arr, pi + 1, high);  
    }  
}
```

```

    }

    public static int partition(int[] arr, int low, int high) {
        int pivot = arr[high];
        int i = (low - 1);
        for (int j = low; j < high; j++) {
            if (arr[j] <= pivot) {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        int temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;
        return i + 1;
    }
}

```

Output:

The screenshot shows the Eclipse IDE with the 'QuickSort.java' file open. The code in the editor is as follows:

```

1 package Daily;
2
3 import java.util.Arrays;
4
5 public class Quicksort {
6     public static void main(String[] args) {
7         int[] arr = {7, 6, 10, 5, 9, 2, 1, 15, 7};
8         quickSort(arr, 0, arr.length - 1);
9         System.out.println(Arrays.toString(arr));
10    }
11
12    public static void quickSort(int[] arr, int low, int high) {
13        if (low < high) {
14            int pi = partition(arr, low, high);
15            quickSort(arr, low, pi - 1);
16            quickSort(arr, pi + 1, high);
17        }
18    }
19
20    public static int partition(int[] arr, int low, int high) {
21        int pivot = arr[high];
22        int i = (low - 1);
23        for (int j = low; j < high; j++) {

```

The console output at the bottom shows the result of the program execution:

```

<terminated> Quicksort [Java Application] C:\Users\tapan.k.p2\poo\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_22.0.1.v202404
[1, 2, 5, 6, 7, 7, 9, 10, 15]

```