Day 14

Name : Siddarth

Date : 13/08/2024

1. Create a Java class with user defined exception handling

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)”

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

Answer:

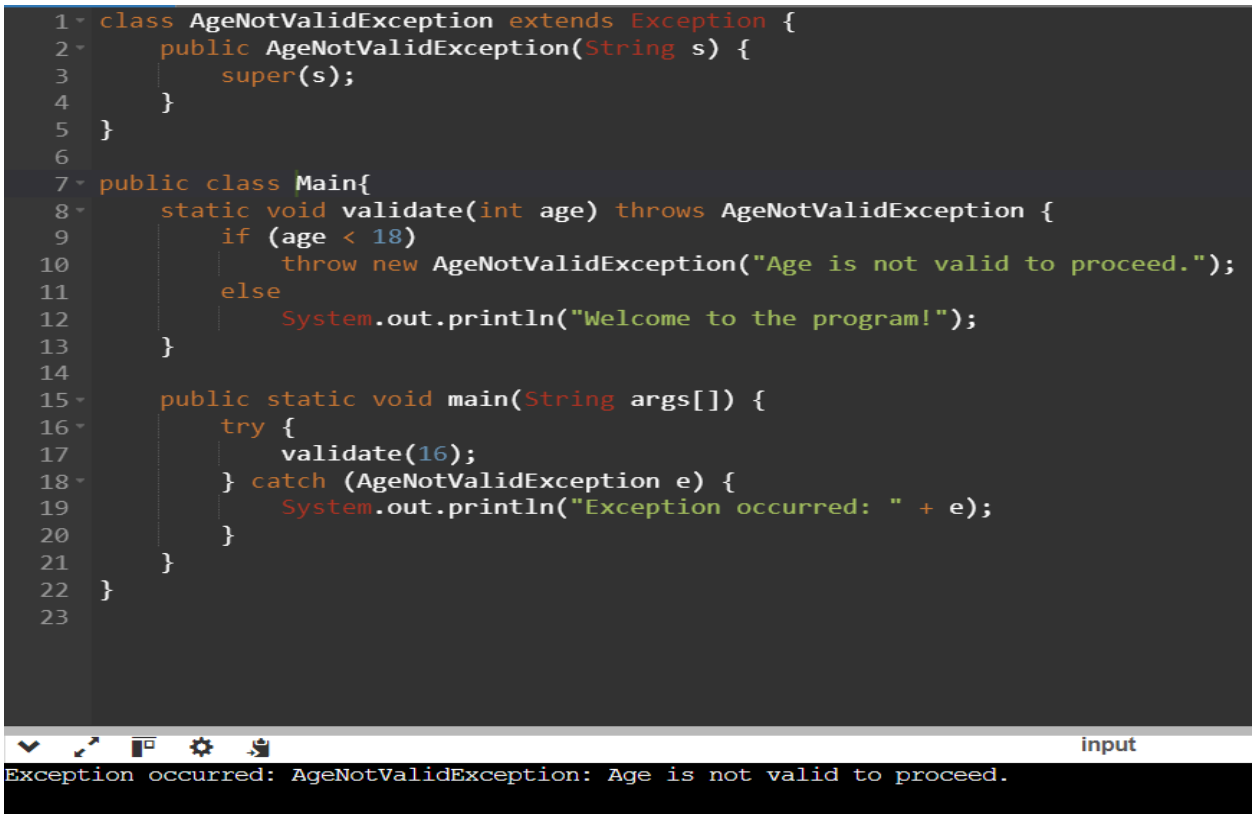
1.Java Class with User-Defined Exception Handling: To create a Java class with user-defined exception handling, you can define a custom exception class by extending the `Exception` class. Then, you can use this custom exception in your program wherever specific error handling is required.

code:

```
class AgeNotValidException extends Exception {
    public AgeNotValidException(String s) {
        super(s);
    }
}

public class UserDefinedExceptionHandling {
    static void validate(int age) throws AgeNotValidException {
        if (age < 18)
            throw new AgeNotValidException("Age is not valid to
proceed.");
        else
            System.out.println("Welcome to the program!");
    }
```

```
    public static void main(String args[]) {
        try {
            validate(16);
        } catch (AgeNotValidException e) {
            System.out.println("Exception occurred: " + e);
        }
    }
}
```

Output ScreenShot:



Explanation:

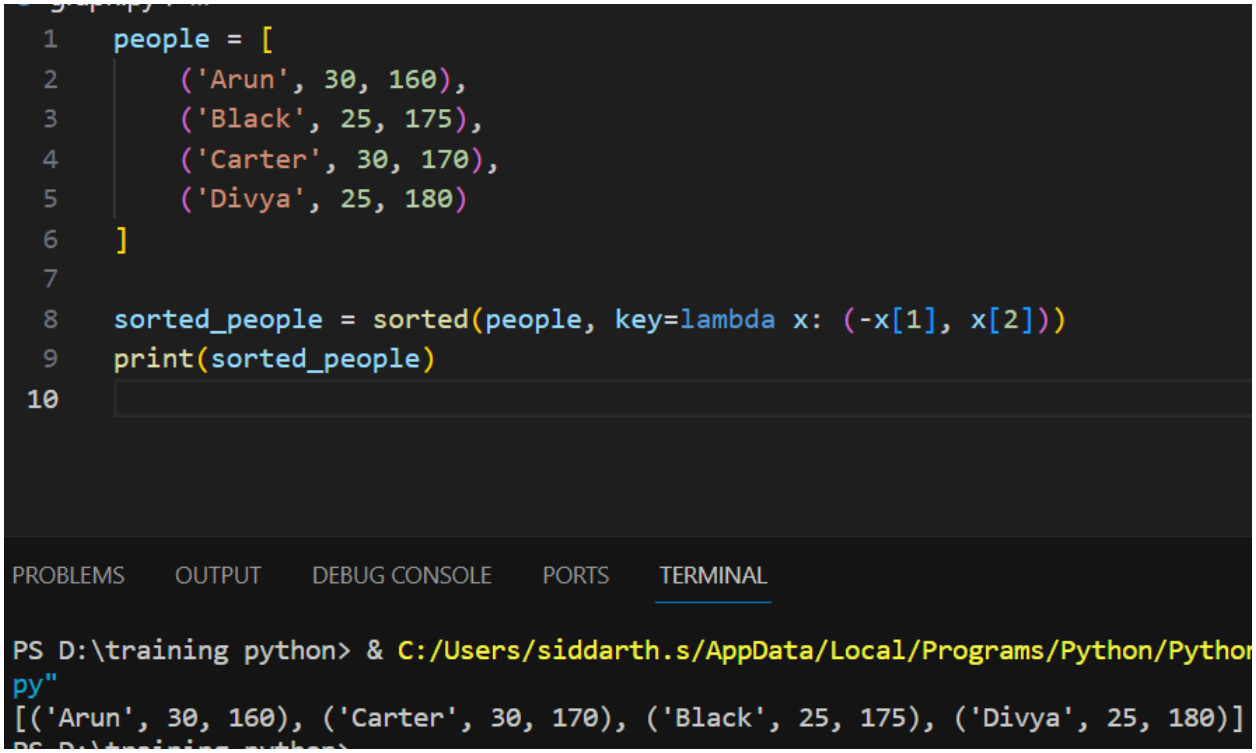
- This code creates a custom exception `AgeNotValidException`.
- The `validate` method throws this exception if the age is below 18.
- The exception is caught in the `main` method and an appropriate message is displayed.

2.Python Code to Sort List by Age (Descending) and Height (Ascending): The provided code is modified to sort the list of people by age in descending order and by height in ascending order.

code:

```
people = [
    ('Arun', 30, 160),
    ('Black', 25, 175),
    ('Carter', 30, 170),
    ('Divya', 25, 180)
]

sorted_people = sorted(people, key=lambda x: (-x[1], x[2]))
print(sorted_people)
```

Output ScreenShot



Explanation:

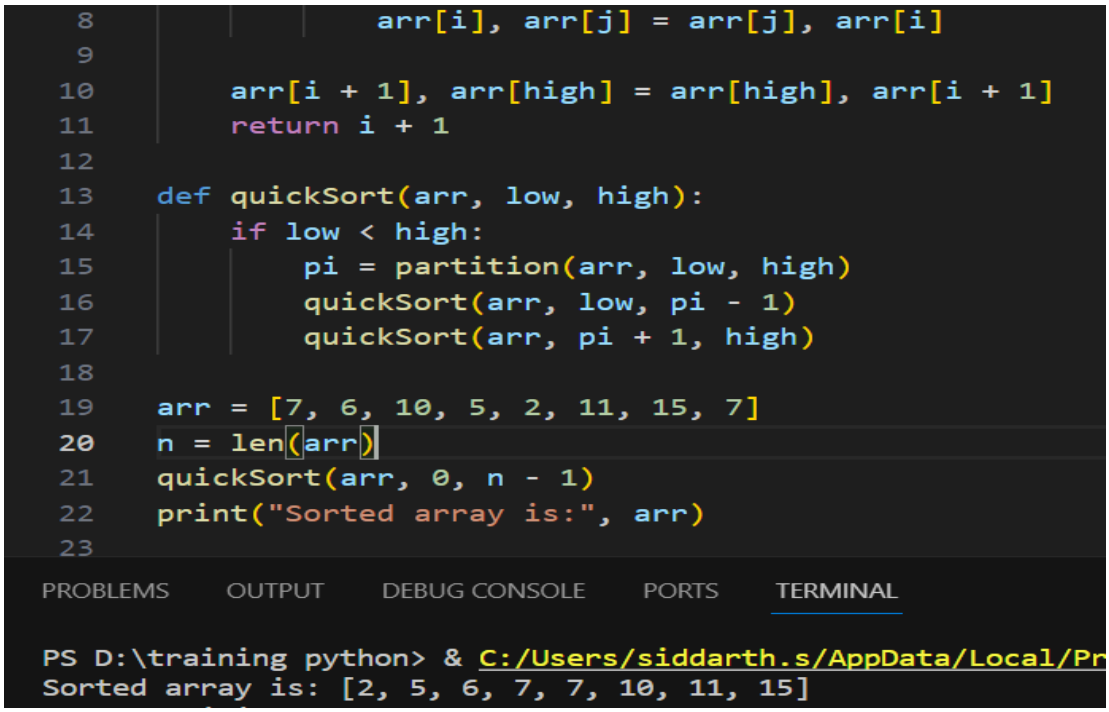
- The `key` parameter in the `sorted` function is modified to first sort by age in descending order (`-x[1]`) and then by height in ascending order (`x[2]`).

3.Quick Sort Implementation: Here's an implementation of Quick Sort in Python to sort the array `[7, 6, 10, 5, 2, 11, 15, 7]`:

code:

```
def partition(arr, low, high):
    i = low - 1
    pivot = arr[high]

    for j in range(low, high):
        if arr[j] <= pivot:
            i = i + 1
            arr[i], arr[j] = arr[j], arr[i]

    arr[i + 1], arr[high] = arr[high], arr[i + 1]
    return i + 1

def quickSort(arr, low, high):
    if low < high:
        pi = partition(arr, low, high)
        quickSort(arr, low, pi - 1)
        quickSort(arr, pi + 1, high)

arr = [7, 6, 10, 5, 2, 11, 15, 7]
n = len(arr)
quickSort(arr, 0, n - 1)
print("Sorted array is:", arr)
```

Output:



```
            arr[i], arr[j] = arr[j], arr[i]

    arr[i + 1], arr[high] = arr[high], arr[i + 1]
    return i + 1

def quickSort(arr, low, high):
    if low < high:
        pi = partition(arr, low, high)
        quickSort(arr, low, pi - 1)
        quickSort(arr, pi + 1, high)

arr = [7, 6, 10, 5, 2, 11, 15, 7]
n = len(arr)
quickSort(arr, 0, n - 1)
print("Sorted array is:", arr)

PROBLEMS  OUTPUT  DEBUG CONSOLE  PORTS  TERMINAL

PS D:\training python> & C:/Users/siddarth.s/AppData/Local/Pr
Sorted array is: [2, 5, 6, 7, 7, 10, 11, 15]
```