Name : Prathik Balaji N
Date : 13-08-2024

Module 5

1.Create a Java class with user defined exception handling.
Code :

```
package Samp;
import java.util.*;
class AgeException extends Exception {
 public AgeException(String message) {
    super(message);
 }
}

public class UserDefinedExceptionExample {

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

 public static void main(String[] args) {

          Scanner in = new Scanner(System.in);

    try {
          System.out.println("Enter Your Age : ");
        validateAge(in.nextInt());
```

```
    } catch (AgeException e) {
        System.out.println("Caught Exception: " + e.getMessage());
    }
 }
}
```

Output :

```
Enter Your Age :
20
Age is valid.
```

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),
]

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

Result :

```
PS C:\Users\Prathik.b\Desktop\Python-Prac> python basic.py
[('Arun', 30, 160), ('Carter', 30, 170), ('Black', 25, 175), ('Divya', 25, 180)]
```

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

Code :

```
package Samp;

import java.util.Arrays;

public class Quicksort {

    public static void main(String[] args) {

        int a[] = { 7,6,10,5,9,2,1,15,7 };
        Quicksort sort = new Quicksort();
        sort.quicksort(a, 0, (a.length - 1));

        for (int i = 0; i < a.length; i++) {
            System.out.println(a[i]);
        }

    }

    public int partition(int a[], int lb, int ub) {
```

```
    int pivot = lb;
    int i = lb;
    int j = ub;

    while (i < j) {
        while (i <= ub && a[i] <= a[pivot]) {
            i++;
        }
        while (j > lb && a[j] > a[pivot]) {
            j--;
        }

        if (i < j) {
            int c = a[i];
            a[i] = a[j];
            a[j] = c;
        }
    }
    int c = a[pivot];
    a[pivot] = a[j];
    a[j] = c;

    return j;
}

public void quicksort(int a[], int lb, int ub) {
    if (lb < ub) {
        int locMiddle = partition(a, lb, ub);
        System.out.println("loc:" + locMiddle + " " + Arrays.toString(a));
        quicksort(a, lb, locMiddle - 1);
        quicksort(a, locMiddle + 1, ub);
    }
```

```
    }
}
```

Result :

```
loc:5 [2, 6, 7, 5, 1, 7, 9, 15, 10]
loc:1 [1, 2, 7, 5, 6, 7, 9, 15, 10]
loc:4 [1, 2, 6, 5, 7, 7, 9, 15, 10]
loc:3 [1, 2, 5, 6, 7, 7, 9, 15, 10]
loc:6 [1, 2, 5, 6, 7, 7, 9, 15, 10]
loc:8 [1, 2, 5, 6, 7, 7, 9, 10, 15]
1
2
5
6
7
7
9
10
15
```