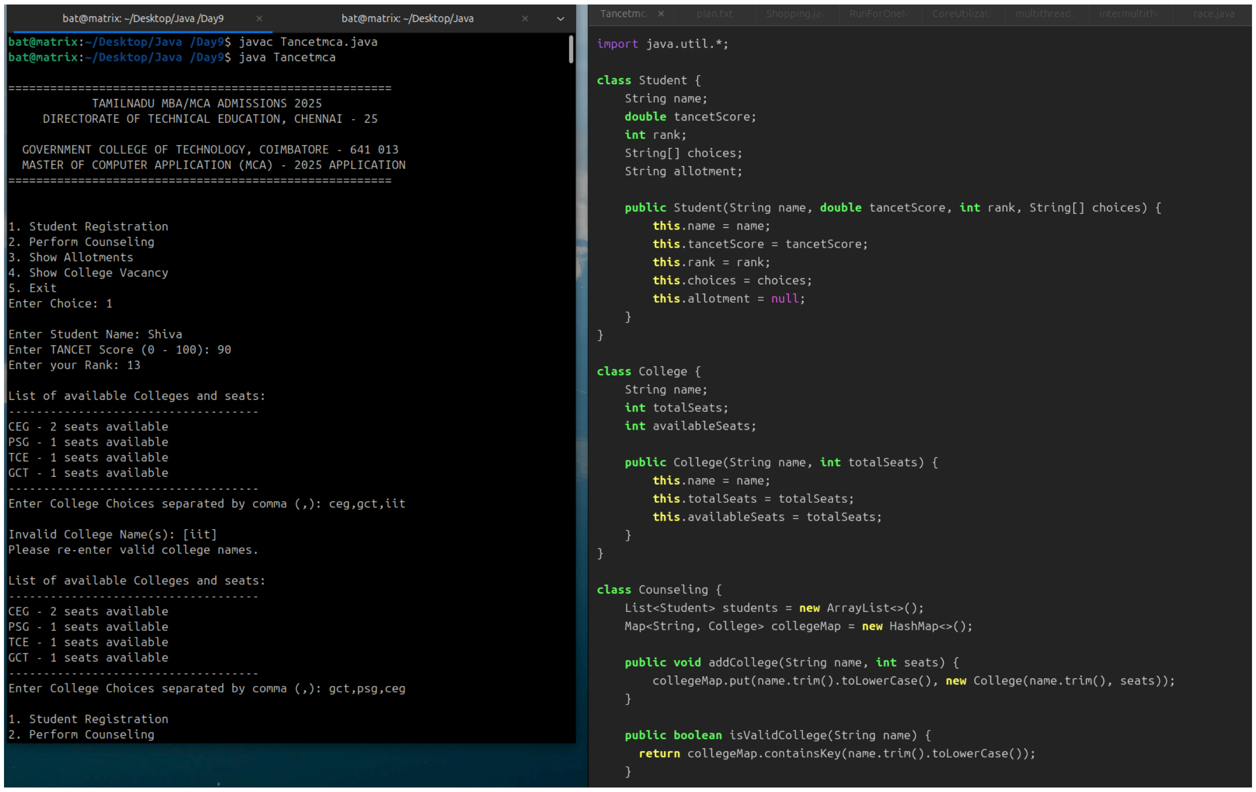
1.Java Program to perform tancet counseling

```
import java.util.*;

class Student {
    String name;
    double tancetScore;
    int rank;
    String[] choices;
    String allotment;

    public Student(String name, double tancetScore, int rank, String[] choices) {
        this.name = name;
        this.tancetScore = tancetScore;
        this.rank = rank;
        this.choices = choices;
        this.allotment = null;
    }
}

class College {
    String name;
    int totalSeats;
    int availableSeats;

    public College(String name, int totalSeats) {
        this.name = name;
        this.totalSeats = totalSeats;
        this.availableSeats = totalSeats;
    }
}

class Counseling {
    List<Student> students = new ArrayList<>();
    Map<String, College> collegeMap = new HashMap<>();

    public void addCollege(String name, int seats) {
        collegeMap.put(name.trim().toLowerCase(), new College(name.trim(), seats));
    }

    public boolean isValidCollege(String name) {
      return collegeMap.containsKey(name.trim().toLowerCase());
    }
    public void registerStudent(Student s) {
      students.add(s);
    }

    public void doCounseling() {
      students.sort(Comparator.comparingInt(s -> s.rank));

      for (Student s : students) {
        for (String choice : s.choices) {
          College c = collegeMap.get(choice.trim());
          if (c != null && c.availableSeats > 0) {
            c.availableSeats--;
            s.allotment = c.name;
            break;
          }
        }
      }
    }

    public void showAllotments() {

System.out.println("\n=========================================================")
;
      System.out.println("            TAMILNADU MBA/MCA ADMISSIONS 2025            ");
      System.out.println("     DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI - 25\n  ");
      System.out.println("  GOVERNMENT COLLEGE OF TECHNOLOGY, COIMBATORE - 641
013 ");
      System.out.println("      MASTER OF COMPUTER APPLICATION (MCA) - 2025 ");
      System.out.println("\n                    STUDENT ALLOTMENT\n");

System.out.println("=========================================================\n")
;
      for (Student s : students) {
        System.out.println("Name: " + s.name + " | Rank: " + s.rank);
        if (s.allotment != null) {
          System.out.println("Allotted College: " + s.allotment);
        } else {
          System.out.println("No college allotted.");
        }
        System.out.println("------------------------------------");
      }

System.out.println("=========================================================\n")
;
```

```
    }

    public void showVacancy() {
        System.out.println("-----------------------------------");
        for (College c : collegeMap.values()) {
            System.out.println(c.name + " - "+c.availableSeats+" seats available");
        }
        System.out.println("-----------------------------------");
    }
}

public class Tancetmca {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Counseling co = new Counseling();

        co.addCollege("CEG", 2);
        co.addCollege("GCT", 1);
        co.addCollege("PSG", 1);
        co.addCollege("TCE", 1);

System.out.println("\n=========================================================")
;
        System.out.println("            TAMILNADU MBA/MCA ADMISSIONS 2025            ");
        System.out.println("    DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI - 25\n  ");
        System.out.println("  GOVERNMENT COLLEGE OF TECHNOLOGY, COIMBATORE - 641
013 ");
        System.out.println("  MASTER OF COMPUTER APPLICATION (MCA) - 2025
APPLICATION");

System.out.println("=========================================================\n")
;

        while (true) {

            System.out.println("\n1. Student Registration\n2. Perform Counseling\n3. Show
Allotments\n4. Show College Vacancy\n5. Exit");
            System.out.print("Enter Choice: ");
            int choice = sc.nextInt();
            sc.nextLine();

            switch (choice) {
                case 1:
                    System.out.print("\n Enter Student Name: ");
                    String name = sc.nextLine();
                    double score;
                    while (true) {
                      System.out.print("Enter TANCET Score (0 - 100): ");
                      score = sc.nextDouble();
                      if (score >= 0 && score <= 100) {break;}
                      else
                      {System.out.println("Invalid score. Please enter a value between 0 and 100.");}
                    }
                    System.out.print("Enter your Rank: ");
                    int rank = sc.nextInt();
                    sc.nextLine();
                    String[] choices;
                    while (true) {
                      System.out.println("\n List of available Colleges and seats:");
                      co.showVacancy();
                      System.out.print("Enter College Choices separated by comma (,): ");
                      String input = sc.nextLine();
                      choices = input.split(",");

                      boolean allValid = true;
                      List<String> invalidNames = new ArrayList<>();

                      for (String c : choices) {
                        if (!co.isValidCollege(c)) {
                            allValid = false;
                            invalidNames.add(c.trim());
                        }
                      }
                      if (allValid) break;
                      System.out.println("\n Invalid College Name(s): " + invalidNames);
                      System.out.println("Please re-enter valid college names.");
                    }
                    co.registerStudent(new Student(name, score, rank, choices));
                    break;

                case 2:
                    co.doCounseling();
                    System.out.println("Counseling Completed.");
                    break;

                case 3:
                    co.showAllotments();
                    break;
```

```
                case 4:
                    co.showVacancy();
                    break;

                case 5:
                    System.out.println("**** Thank You! Happy Collegian ****");
                    return;

                default:
                    System.out.println("Enter valid option only.");
            }
        }
    }
}
```



```
bat@matrix:~/Desktop/Java /Day9$ javac Tancetmca.java
bat@matrix:~/Desktop/Java /Day9$ java Tancetmca

=======================================================
            TAMILNADU MBA/MCA ADMISSIONS 2025
     DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI - 25

  GOVERNMENT COLLEGE OF TECHNOLOGY, COIMBATORE - 641 013
  MASTER OF COMPUTER APPLICATION (MCA) - 2025 APPLICATION
=======================================================


1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 1

Enter Student Name: Shiva
Enter TANCET Score (0 - 100): 90
Enter your Rank: 13

List of available Colleges and seats:
------------------------------------
CEG - 2 seats available
PSG - 1 seats available
TCE - 1 seats available
GCT - 1 seats available
------------------------------------
Enter College Choices separated by comma (,): ceg,gct,iit

Invalid College Name(s): [iit]
Please re-enter valid college names.

List of available Colleges and seats:
------------------------------------
CEG - 2 seats available
PSG - 1 seats available
TCE - 1 seats available
GCT - 1 seats available
------------------------------------
Enter College Choices separated by comma (,): gct,psg,ceg

1. Student Registration
2. Perform Counseling
```

```java
import java.util.*;

class Student {
    String name;
    double tancetScore;
    int rank;
    String[] choices;
    String allotment;

    public Student(String name, double tancetScore, int rank, String[] choices) {
        this.name = name;
        this.tancetScore = tancetScore;
        this.rank = rank;
        this.choices = choices;
        this.allotment = null;
    }
}

class College {
    String name;
    int totalSeats;
    int availableSeats;

    public College(String name, int totalSeats) {
        this.name = name;
        this.totalSeats = totalSeats;
        this.availableSeats = totalSeats;
    }
}

class Counseling {
    List<Student> students = new ArrayList<>();
    Map<String, College> collegeMap = new HashMap<>();

    public void addCollege(String name, int seats) {
        collegeMap.put(name.trim().toLowerCase(), new College(name.trim(), seats));
    }

    public boolean isValidCollege(String name) {
      return collegeMap.containsKey(name.trim().toLowerCase());
    }
```

Output:

```
=======================================================
        TAMILNADU MBA/MCA ADMISSIONS 2025
   DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI - 25

 GOVERNMENT COLLEGE OF TECHNOLOGY, COIMBATORE - 641 013
```

```
  MASTER OF COMPUTER APPLICATION (MCA) - 2025 APPLICATION
===========================================================


1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 1

Enter Student Name: Shiva
Enter TANCET Score (0 - 100): 90
Enter your Rank: 13

List of available Colleges and seats:
-----------------------------------
CEG - 2 seats available
PSG - 1 seats available
TCE - 1 seats available
GCT - 1 seats available
-----------------------------------
Enter College Choices separated by comma (,): ceg,gct,iit

Invalid College Name(s): [iit]
Please re-enter valid college names.

List of available Colleges and seats:
-----------------------------------
CEG - 2 seats available
PSG - 1 seats available
TCE - 1 seats available
GCT - 1 seats available
-----------------------------------
Enter College Choices separated by comma (,): gct,psg,ceg

1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 1

Enter Student Name: Balan
Enter TANCET Score (0 - 100): 99
Enter your Rank: 1

List of available Colleges and seats:
-----------------------------------
CEG - 2 seats available
PSG - 1 seats available
TCE - 1 seats available
GCT - 1 seats available
-----------------------------------
Enter College Choices separated by comma (,): gct,ceg,psg

1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 2
Counseling Completed.

1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 3

=====================================================
          TAMILNADU MBA/MCA ADMISSIONS 2025
    DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI - 25

  GOVERNMENT COLLEGE OF TECHNOLOGY, COIMBATORE - 641 013
     MASTER OF COMPUTER APPLICATION (MCA) - 2025

                 STUDENT ALLOTMENT

=====================================================

Name: Balan | Rank: 1
Allotted College: GCT
-----------------------------------
Name: Shiva | Rank: 13
Allotted College: PSG
-----------------------------------
```

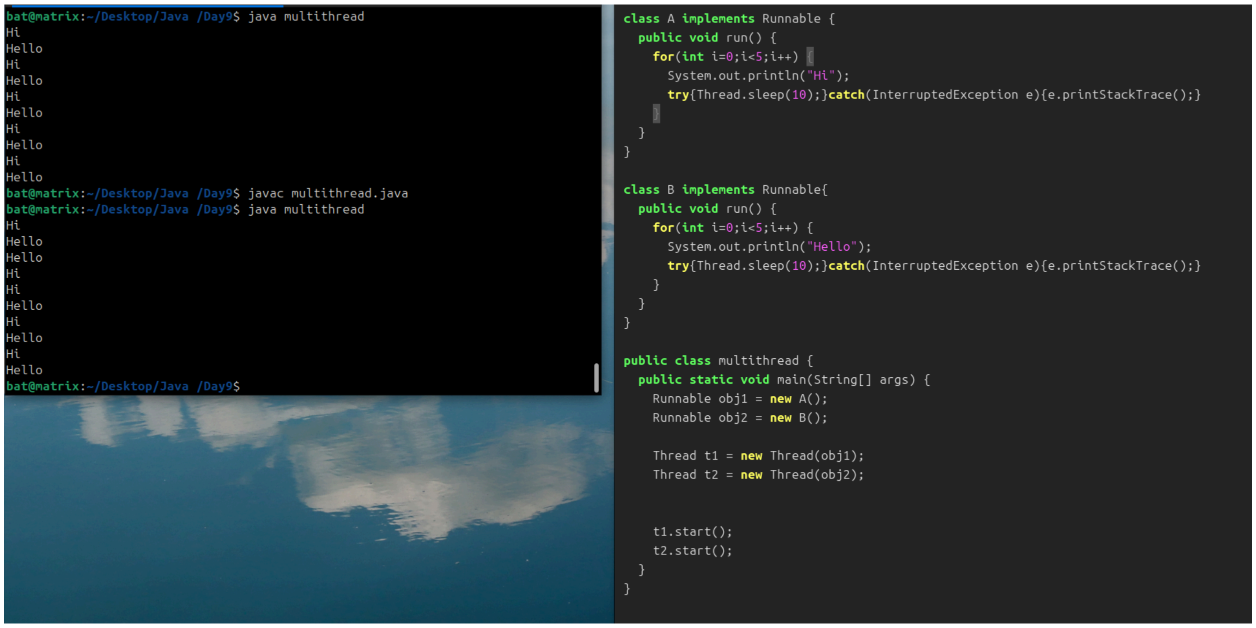
```
==========================================================

1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 4
-----------------------------------
CEG - 2 seats available
PSG - 0 seats available
TCE - 1 seats available
GCT - 0 seats available
-----------------------------------

1. Student Registration
2. Perform Counseling
3. Show Allotments
4. Show College Vacancy
5. Exit
Enter Choice: 5
**** Thank You! Happy Collegian ****
```

2.Multithreading Interface normal

```
class A implements Runnable {
	public void run() {
		for(int i=0;i<5;i++) {
			System.out.println("Hi");
			try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
		}
	}
}

class B implements Runnable{
	public void run() {
		for(int i=0;i<5;i++) {
			System.out.println("Hello");
			try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
		}
	}
}

public class multithread {
        public static void main(String[] args) {
                Runnable obj1 = new A();
                Runnable obj2 = new B();

                Thread t1 = new Thread(obj1);
                Thread t2 = new Thread(obj2);


                t1.start();
    t2.start();
        }
}
```

Output:

```
Hi
Hello
Hello
Hi
Hi
Hello
Hi
Hello
Hi
Hello
```



```
bat@matrix:~/Desktop/Java /Day9$ java multithread
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
bat@matrix:~/Desktop/Java /Day9$ javac multithread.java
bat@matrix:~/Desktop/Java /Day9$ java multithread
Hi
Hello
Hello
Hi
Hi
Hello
Hi
Hello
Hi
Hello
bat@matrix:~/Desktop/Java /Day9$
```

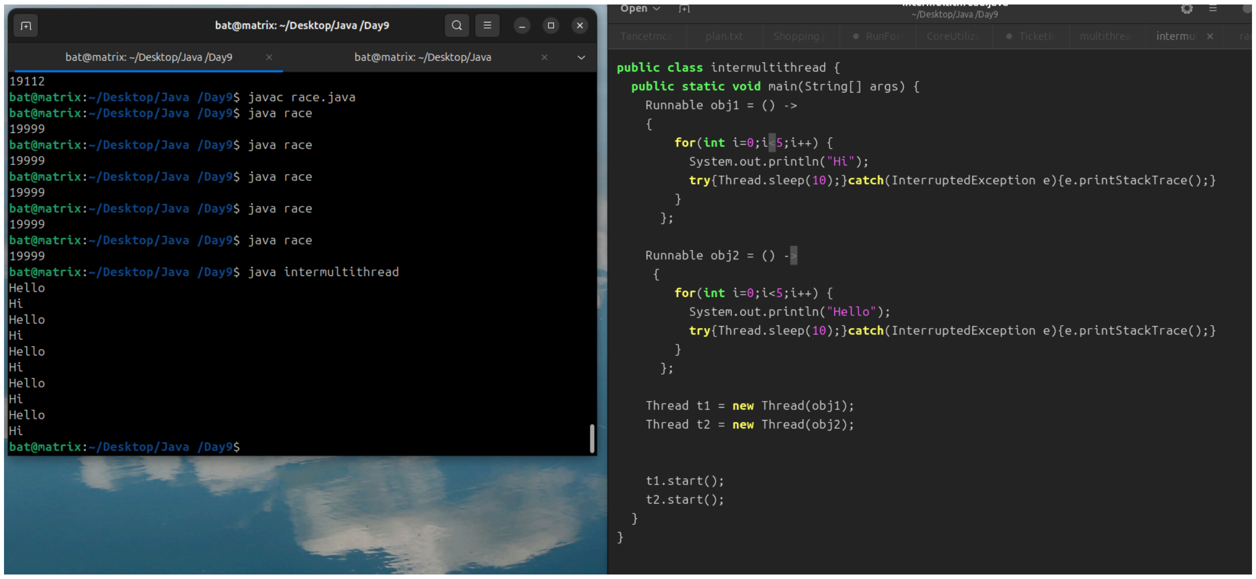
```java
class A implements Runnable {
  public void run() {
    for(int i=0;i<5;i++) {
      System.out.println("Hi");
      try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
    }
  }
}

class B implements Runnable{
  public void run() {
    for(int i=0;i<5;i++) {
      System.out.println("Hello");
      try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
    }
  }
}

public class multithread {
  public static void main(String[] args) {
    Runnable obj1 = new A();
    Runnable obj2 = new B();

    Thread t1 = new Thread(obj1);
    Thread t2 = new Thread(obj2);


    t1.start();
    t2.start();
  }
}
```

3.Multithreading using Interface and lambda expression

```
public class intermultithread {
        public static void main(String[] args) {
                Runnable obj1 = () ->
         {
                   for(int i=0;i<5;i++) {
                           System.out.println("Hi");
                           try{Thread.sleep(10);}catch(InterruptedException
e){e.printStackTrace();}
                   }
           };

                Runnable obj2 = () ->
          {
                   for(int i=0;i<5;i++) {
                           System.out.println("Hello");
                           try{Thread.sleep(10);}catch(InterruptedException
e){e.printStackTrace();}
                   }
           };

                Thread t1 = new Thread(obj1);
                Thread t2 = new Thread(obj2);


                t1.start();
   t2.start();
        }
}
```

Output:

```
bat@matrix:~/Desktop/Java /Day9$ java intermultithread
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
```



```
19112
bat@matrix:~/Desktop/Java /Day9$ javac race.java
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java intermultithread
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
bat@matrix:~/Desktop/Java /Day9$
```

```java
public class intermultithread {
  public static void main(String[] args) {
    Runnable obj1 = () ->
    {
        for(int i=0;i<5;i++) {
          System.out.println("Hi");
          try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
        }
      };

    Runnable obj2 = () ->
     {
        for(int i=0;i<5;i++) {
          System.out.println("Hello");
          try{Thread.sleep(10);}catch(InterruptedException e){e.printStackTrace();}
        }
      };

    Thread t1 = new Thread(obj1);
    Thread t2 = new Thread(obj2);


    t1.start();
    t2.start();
  }
}
```

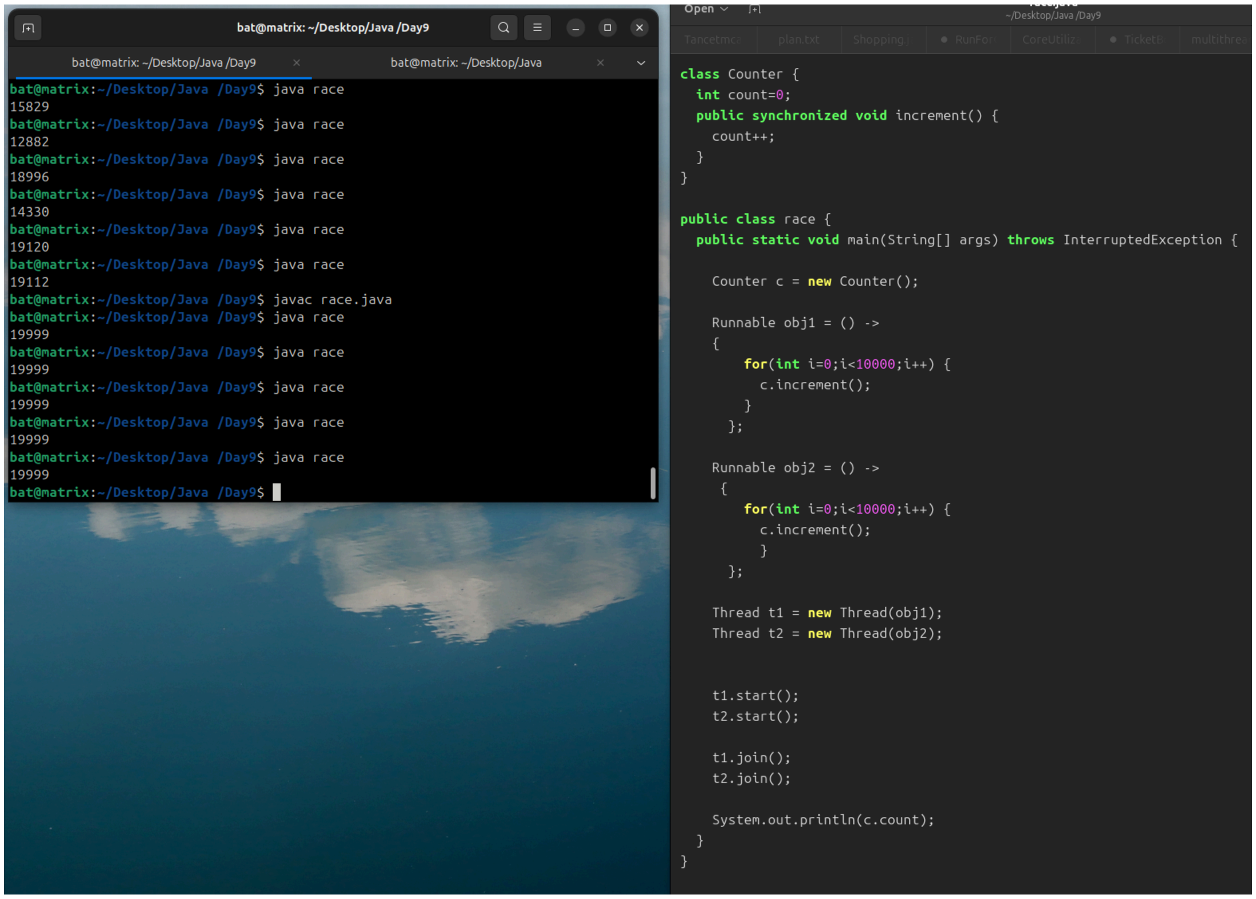
4.Race Condition:

```java
class Counter {
  int count=0;
  public synchronized void increment() {
    count++;
  }
}

public class race {
        public static void main(String[] args) throws InterruptedException {

          Counter c = new Counter();

                Runnable obj1 = () ->
          {
                    for(int i=0;i<10000;i++) {
                            c.increment();
                    }
            };

                Runnable obj2 = () ->
           {
                    for(int i=0;i<10000;i++) {
                      c.increment();
                            }
            };
```

```
                        Thread t1 = new Thread(obj1);
                        Thread t2 = new Thread(obj2);


                        t1.start();
    t2.start();

    t1.join();
    t2.join();

    System.out.println(c.count);
            }
}
```

Output:

```
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
```

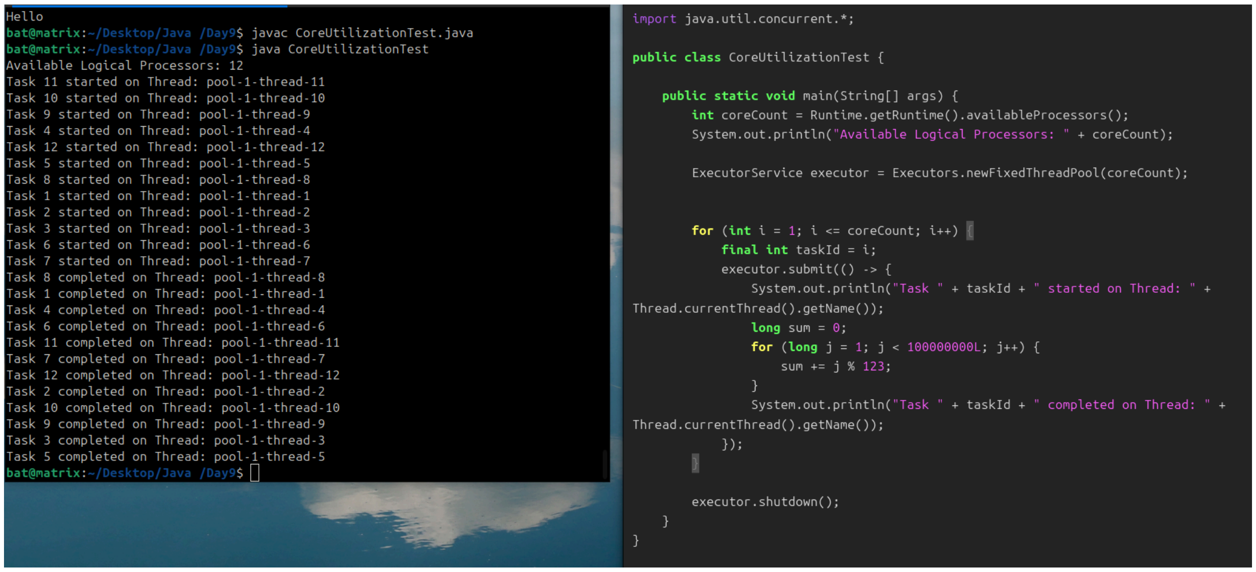

bat@matrix: ~/Desktop/Java /Day9
bat@matrix: ~/Desktop/Java /Day9
bat@matrix: ~/Desktop/Java
Open
~/Desktop/Java /Day9

```
bat@matrix:~/Desktop/Java /Day9$ java race
15829
bat@matrix:~/Desktop/Java /Day9$ java race
12882
bat@matrix:~/Desktop/Java /Day9$ java race
18996
bat@matrix:~/Desktop/Java /Day9$ java race
14330
bat@matrix:~/Desktop/Java /Day9$ java race
19120
bat@matrix:~/Desktop/Java /Day9$ java race
19112
bat@matrix:~/Desktop/Java /Day9$ javac race.java
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$ java race
19999
bat@matrix:~/Desktop/Java /Day9$
```

```java
class Counter {
  int count=0;
  public synchronized void increment() {
    count++;
  }
}

public class race {
  public static void main(String[] args) throws InterruptedException {

    Counter c = new Counter();

    Runnable obj1 = () ->
    {
        for(int i=0;i<10000;i++) {
          c.increment();
        }
      };

    Runnable obj2 = () ->
     {
        for(int i=0;i<10000;i++) {
          c.increment();
          }
      };

    Thread t1 = new Thread(obj1);
    Thread t2 = new Thread(obj2);


    t1.start();
    t2.start();

    t1.join();
    t2.join();

    System.out.println(c.count);
  }
}
```

5. Utilizing all cores in the machine

```
import java.util.concurrent.*;

public class CoreUtilizationTest {

    public static void main(String[] args) {
        int coreCount = Runtime.getRuntime().availableProcessors();
        System.out.println("Available Logical Processors: " + coreCount);

        ExecutorService executor = Executors.newFixedThreadPool(coreCount);


        for (int i = 1; i <= coreCount; i++) {
            final int taskId = i;
            executor.submit(() -> {
                System.out.println("Task " + taskId + " started on Thread: " +
Thread.currentThread().getName());
                long sum = 0;
                for (long j = 1; j < 100000000L; j++) {
                    sum += j % 123;
                }
                System.out.println("Task " + taskId + " completed on Thread: " +
Thread.currentThread().getName());
            });
        }

        executor.shutdown();
    }
}
```



```
Hello
bat@matrix:~/Desktop/Java /Day9$ javac CoreUtilizationTest.java
bat@matrix:~/Desktop/Java /Day9$ java CoreUtilizationTest
Available Logical Processors: 12
Task 11 started on Thread: pool-1-thread-11
Task 10 started on Thread: pool-1-thread-10
Task 9 started on Thread: pool-1-thread-9
Task 4 started on Thread: pool-1-thread-4
Task 12 started on Thread: pool-1-thread-12
Task 5 started on Thread: pool-1-thread-5
Task 8 started on Thread: pool-1-thread-8
Task 1 started on Thread: pool-1-thread-1
Task 2 started on Thread: pool-1-thread-2
Task 3 started on Thread: pool-1-thread-3
Task 6 started on Thread: pool-1-thread-6
Task 7 started on Thread: pool-1-thread-7
Task 8 completed on Thread: pool-1-thread-8
Task 1 completed on Thread: pool-1-thread-1
Task 4 completed on Thread: pool-1-thread-4
Task 6 completed on Thread: pool-1-thread-6
Task 11 completed on Thread: pool-1-thread-11
Task 7 completed on Thread: pool-1-thread-7
Task 12 completed on Thread: pool-1-thread-12
Task 2 completed on Thread: pool-1-thread-2
Task 10 completed on Thread: pool-1-thread-10
Task 9 completed on Thread: pool-1-thread-9
Task 3 completed on Thread: pool-1-thread-3
Task 5 completed on Thread: pool-1-thread-5
bat@matrix:~/Desktop/Java /Day9$
```

```java
import java.util.concurrent.*;

public class CoreUtilizationTest {

    public static void main(String[] args) {
        int coreCount = Runtime.getRuntime().availableProcessors();
        System.out.println("Available Logical Processors: " + coreCount);

        ExecutorService executor = Executors.newFixedThreadPool(coreCount);


        for (int i = 1; i <= coreCount; i++) {
            final int taskId = i;
            executor.submit(() -> {
                System.out.println("Task " + taskId + " started on Thread: " +
Thread.currentThread().getName());
                long sum = 0;
                for (long j = 1; j < 100000000L; j++) {
                    sum += j % 123;
                }
                System.out.println("Task " + taskId + " completed on Thread: " +
Thread.currentThread().getName());
            });
        }

        executor.shutdown();
    }
}
```

Output:

```
Available Logical Processors: 12
Task 11 started on Thread: pool-1-thread-11
Task 10 started on Thread: pool-1-thread-10
Task 9 started on Thread: pool-1-thread-9
Task 4 started on Thread: pool-1-thread-4
Task 12 started on Thread: pool-1-thread-12
Task 5 started on Thread: pool-1-thread-5
Task 8 started on Thread: pool-1-thread-8
Task 1 started on Thread: pool-1-thread-1
Task 2 started on Thread: pool-1-thread-2
Task 3 started on Thread: pool-1-thread-3
Task 6 started on Thread: pool-1-thread-6
Task 7 started on Thread: pool-1-thread-7
Task 8 completed on Thread: pool-1-thread-8
Task 1 completed on Thread: pool-1-thread-1
Task 4 completed on Thread: pool-1-thread-4
Task 6 completed on Thread: pool-1-thread-6
Task 11 completed on Thread: pool-1-thread-11
Task 7 completed on Thread: pool-1-thread-7
Task 12 completed on Thread: pool-1-thread-12
Task 2 completed on Thread: pool-1-thread-2
Task 10 completed on Thread: pool-1-thread-10
Task 9 completed on Thread: pool-1-thread-9
Task 3 completed on Thread: pool-1-thread-3
Task 5 completed on Thread: pool-1-thread-5
```