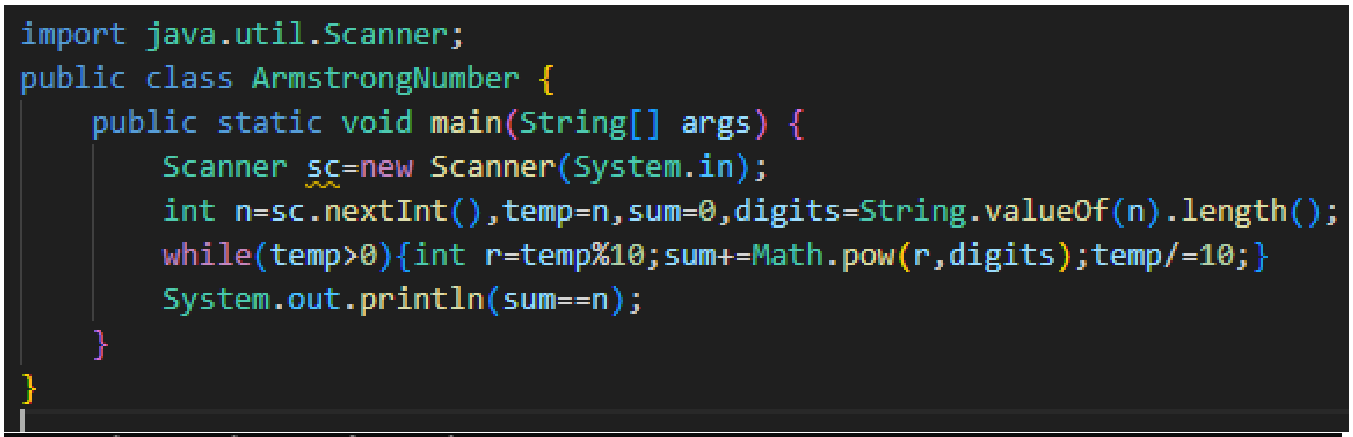
ADS
Assignment 1

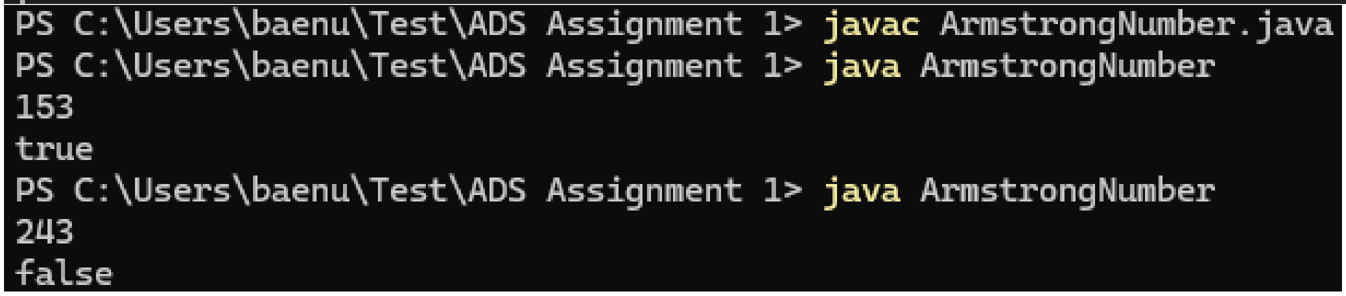
ANSWERS

Following things to be added in each question:
-Program
-Flow chart
-Output
Submission Date: 18/09/2025

1. Armstrong Number
Problem: Write a Java program to check if a given number is an Armstrong number.
Test Cases:
Input: 153
Output: true
Input: 123
Output: false

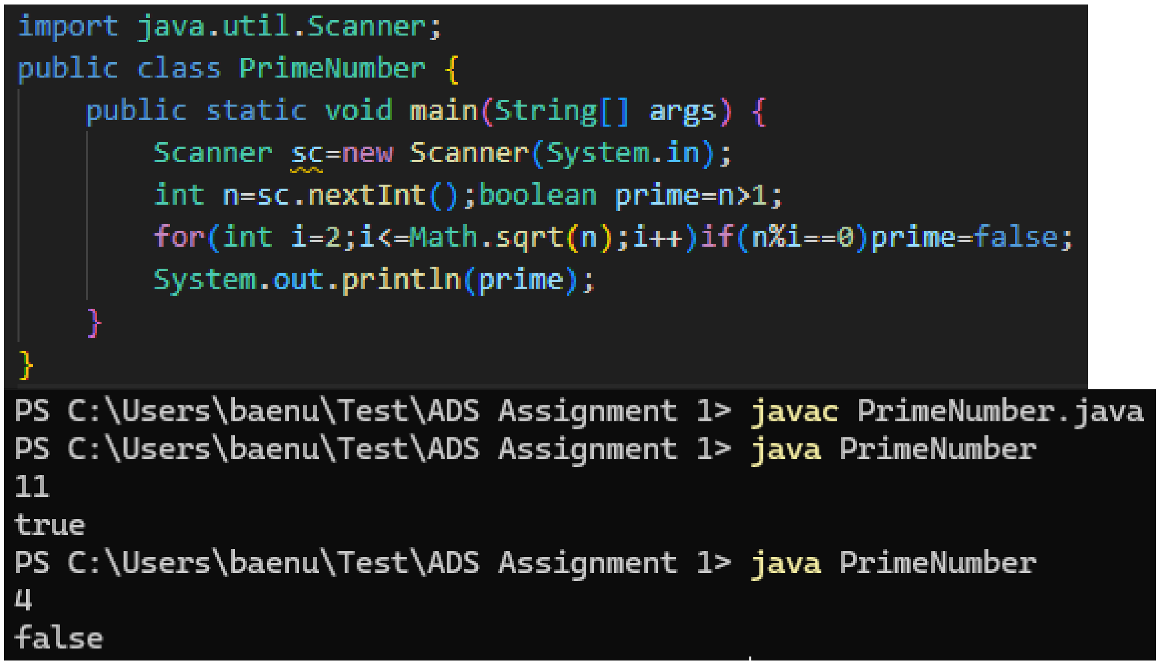


```java
import java.util.Scanner;
public class ArmstrongNumber {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt(),temp=n,sum=0,digits=String.valueOf(n).length();
        while(temp>0){int r=temp%10;sum+=Math.pow(r,digits);temp/=10;}
        System.out.println(sum==n);
    }
}
```



```
PS C:\Users\baenu\Test\ADS Assignment 1> javac ArmstrongNumber.java
PS C:\Users\baenu\Test\ADS Assignment 1> java ArmstrongNumber
153
true
PS C:\Users\baenu\Test\ADS Assignment 1> java ArmstrongNumber
243
false
```

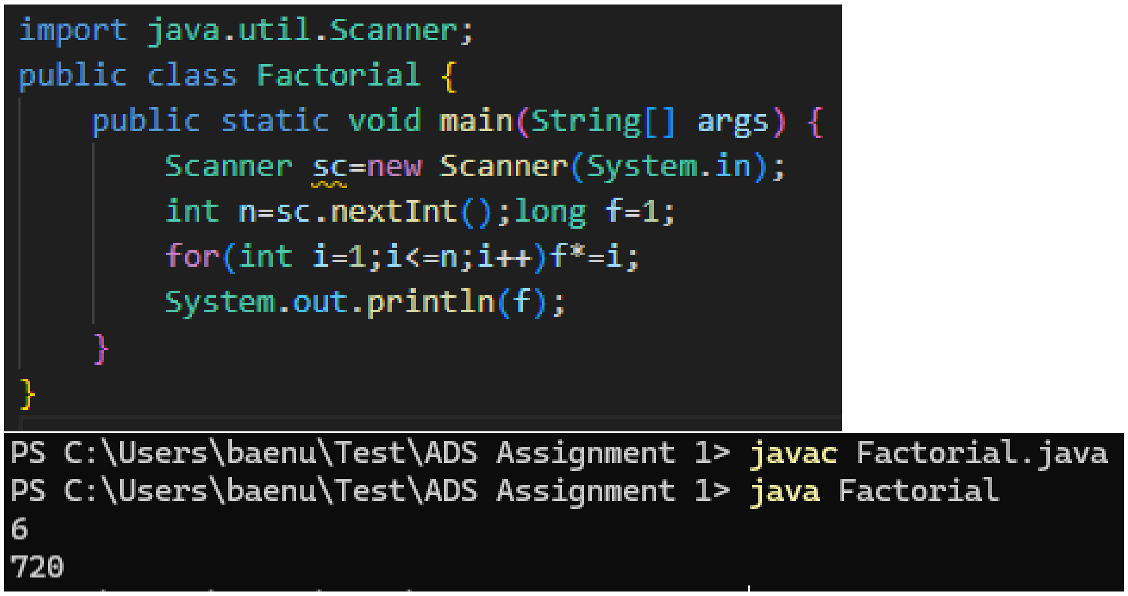
2. Prime Number
Problem: Write a Java program to check if a given number is prime.
Test Cases:
Input: 29
Output: true
Input: 15
Output: false



```java
import java.util.Scanner;
public class PrimeNumber {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();boolean prime=n>1;
        for(int i=2;i<=Math.sqrt(n);i++)if(n%i==0)prime=false;
        System.out.println(prime);
    }
}
```

```
PS C:\Users\baenu\Test\ADS Assignment 1> javac PrimeNumber.java
PS C:\Users\baenu\Test\ADS Assignment 1> java PrimeNumber
11
true
PS C:\Users\baenu\Test\ADS Assignment 1> java PrimeNumber
4
false
```

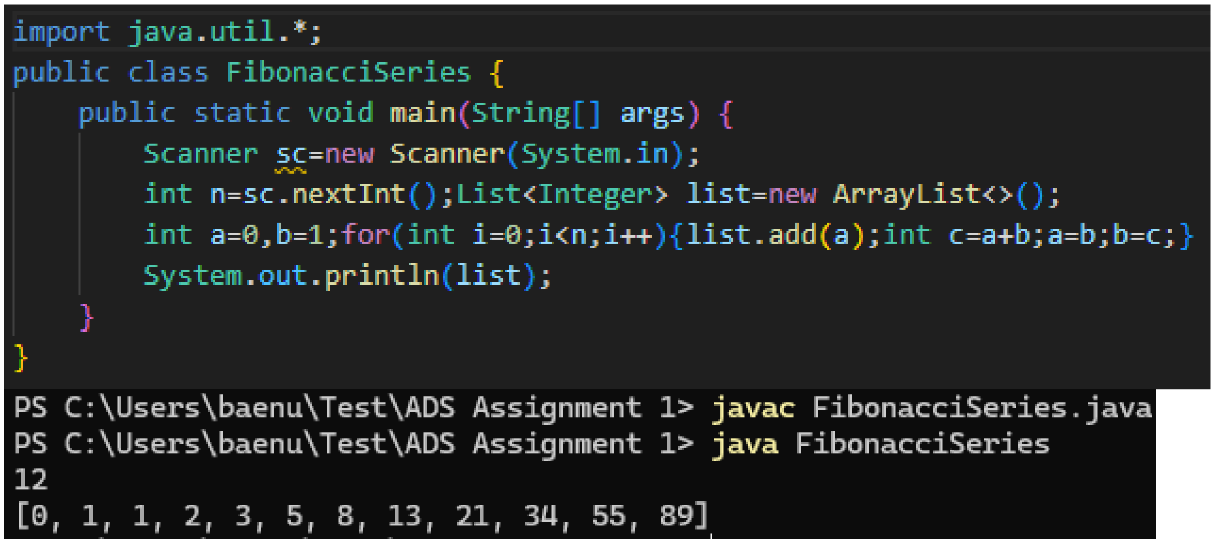
3. Factorial
Problem: Write a Java program to compute the factorial of a given number.
Test Cases:
Input: 5
Output: 120
Input: 0
Output: 1



```java
import java.util.Scanner;
public class Factorial {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();long f=1;
        for(int i=1;i<=n;i++)f*=i;
        System.out.println(f);
    }
}
```

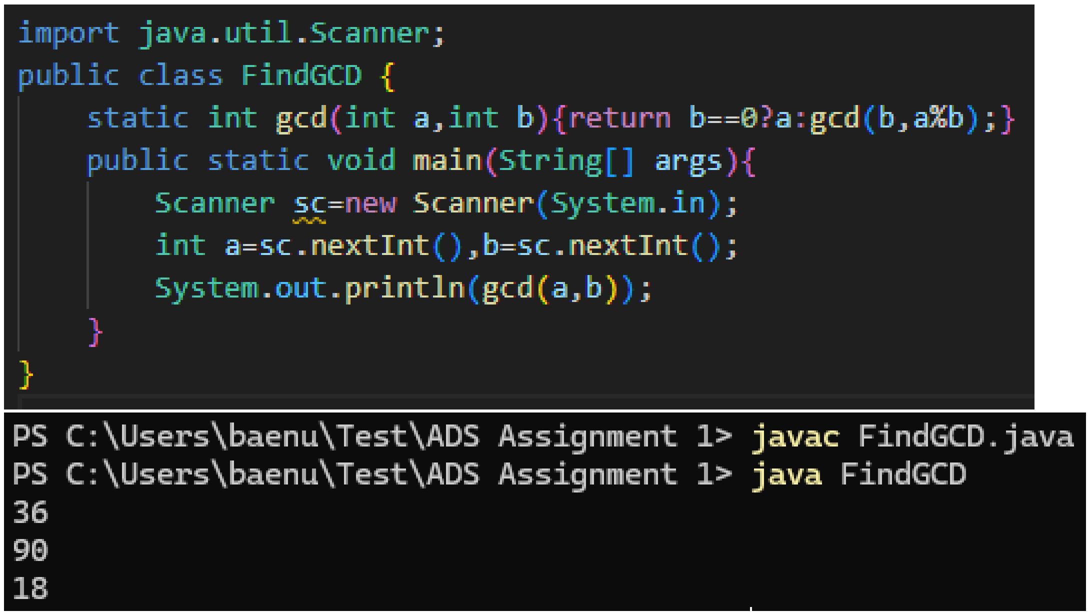
```
PS C:\Users\baenu\Test\ADS Assignment 1> javac Factorial.java
PS C:\Users\baenu\Test\ADS Assignment 1> java Factorial
6
720
```

4. Fibonacci Series
Problem: Write a Java program to print the first n numbers in the Fibonacci series.
Test Cases:
Input: n = 5
Output: [0, 1, 1, 2, 3]
Input: n = 8
Output: [0, 1, 1, 2, 3, 5, 8, 13]


```java
import java.util.*;
public class FibonacciSeries {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();List<Integer> list=new ArrayList<>();
        int a=0,b=1;for(int i=0;i<n;i++){list.add(a);int c=a+b;a=b;b=c;}
        System.out.println(list);
    }
}
```

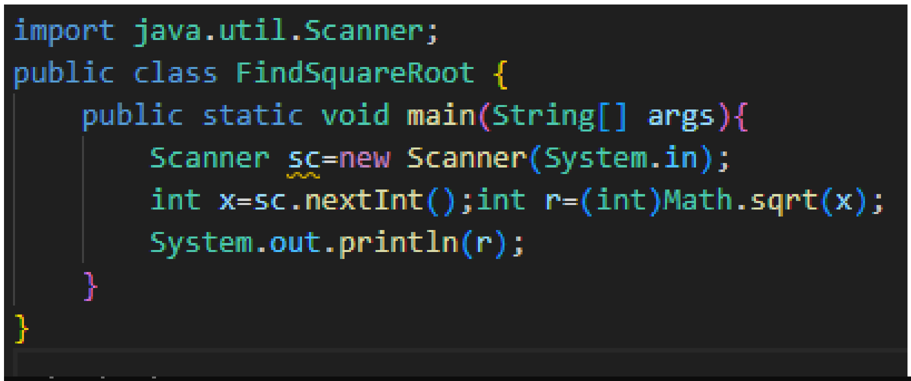
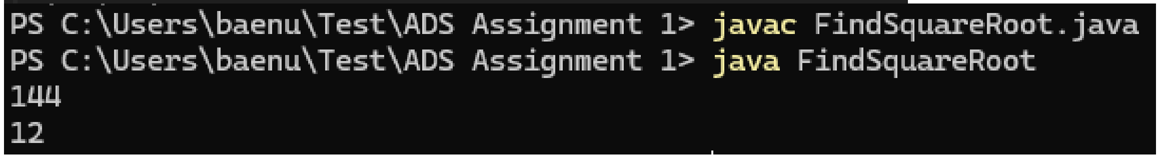
```
PS C:\Users\baenu\Test\ADS Assignment 1> javac FibonacciSeries.java
PS C:\Users\baenu\Test\ADS Assignment 1> java FibonacciSeries
12
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

5. Find GCD
Problem: Write a Java program to find the Greatest Common Divisor (GCD) of two numbers.
Test Cases:
Input: a = 54, b = 24
Output: 6
Input: a = 17, b = 13
Output: 1

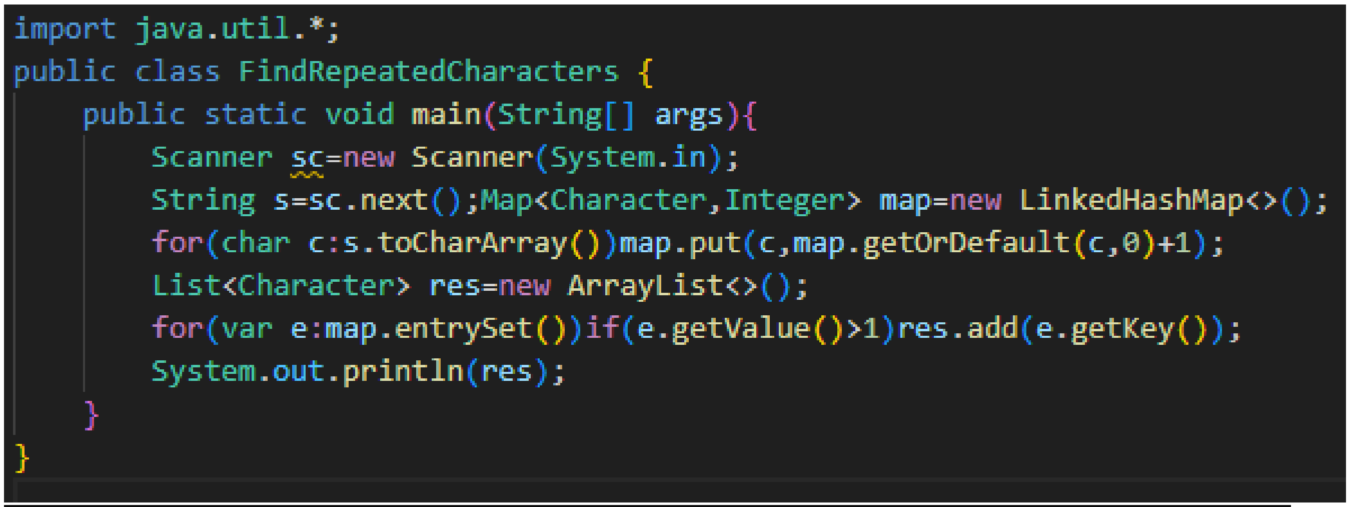
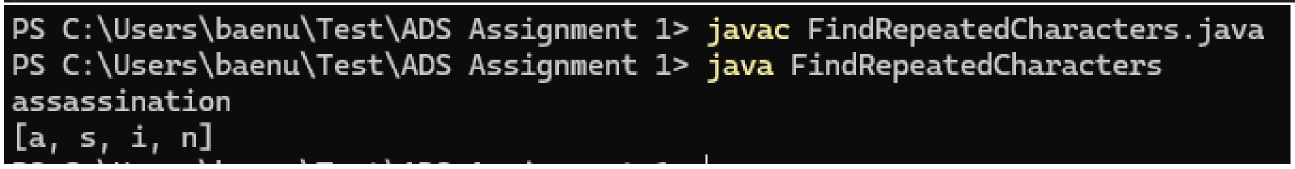

```java
import java.util.Scanner;
public class FindGCD {
    static int gcd(int a,int b){return b==0?a:gcd(b,a%b);}
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int a=sc.nextInt(),b=sc.nextInt();
        System.out.println(gcd(a,b));
    }
}
```

```
PS C:\Users\baenu\Test\ADS Assignment 1> javac FindGCD.java
PS C:\Users\baenu\Test\ADS Assignment 1> java FindGCD
36
90
18
```

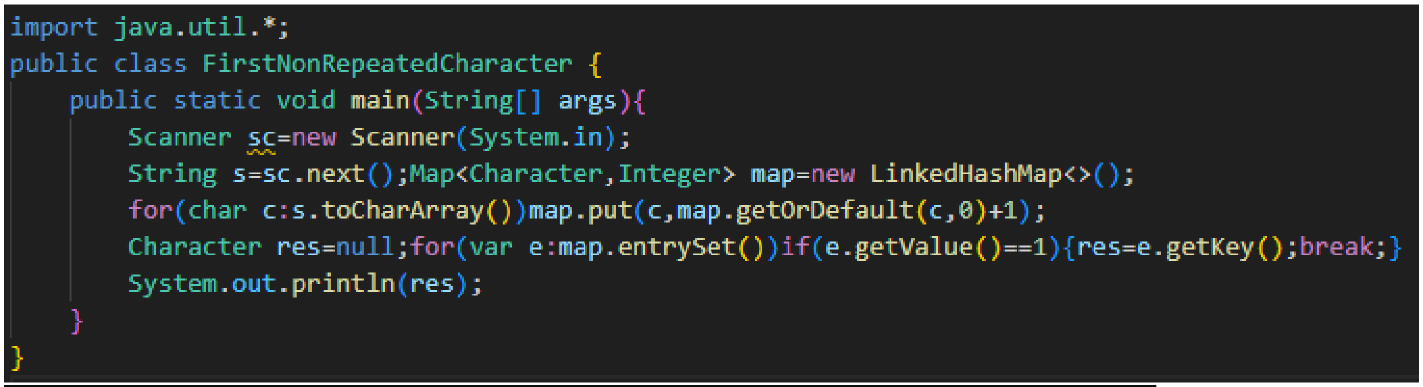
6. Find Square Root
Problem: Write a Java program to find the square root of a given number (using integer approximation).
Test Cases:
Input: x = 16
Output: 4
Input: x = 27
Output: 5


```java
import java.util.Scanner;
public class FindSquareRoot {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int x=sc.nextInt();int r=(int)Math.sqrt(x);
        System.out.println(r);
    }
}
```

```
PS C:\Users\baenu\Test\ADS Assignment 1> javac FindSquareRoot.java
PS C:\Users\baenu\Test\ADS Assignment 1> java FindSquareRoot
144
12
```

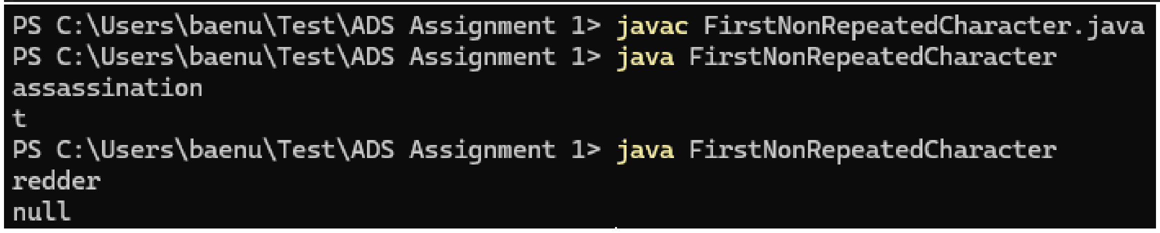
7. Find Repeated Characters in a String
Problem: Write a Java program to find all repeated characters in a string.
Test Cases:
Input: "programming"
Output: ['r', 'g', 'm']
Input: "hello"
Output: ['l']


```java
import java.util.*;
public class FindRepeatedCharacters {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        String s=sc.next();Map<Character,Integer> map=new LinkedHashMap<>();
        for(char c:s.toCharArray())map.put(c,map.getOrDefault(c,0)+1);
        List<Character> res=new ArrayList<>();
        for(var e:map.entrySet())if(e.getValue()>1)res.add(e.getKey());
        System.out.println(res);
    }
}
```

```
PS C:\Users\baenu\Test\ADS Assignment 1> javac FindRepeatedCharacters.java
PS C:\Users\baenu\Test\ADS Assignment 1> java FindRepeatedCharacters
assassination
[a, s, i, n]
```

8. First Non-Repeated Character
Problem: Write a Java program to find the first non-repeated character in a string.
Test Cases:
Input: "stress"
Output: 't'
Input: "aabbcc"
Output: null

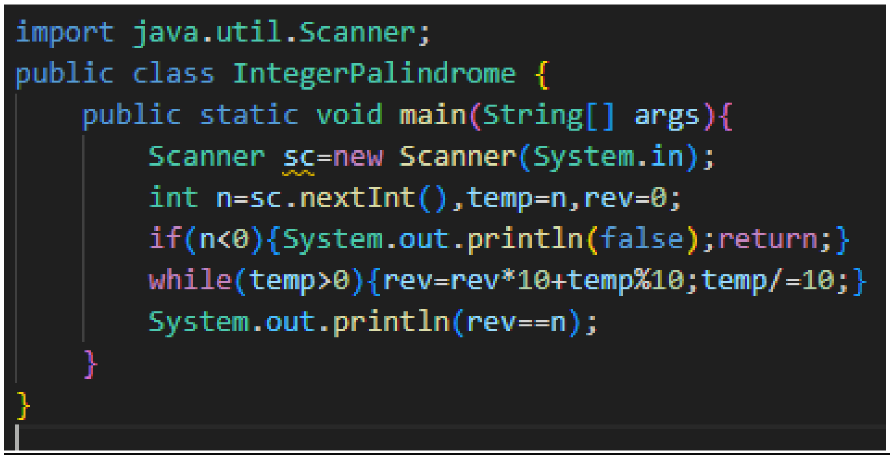


```java
import java.util.*;
public class FirstNonRepeatedCharacter {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        String s=sc.next();Map<Character,Integer> map=new LinkedHashMap<>();
        for(char c:s.toCharArray())map.put(c,map.getOrDefault(c,0)+1);
        Character res=null;for(var e:map.entrySet())if(e.getValue()==1){res=e.getKey();break;}
        System.out.println(res);
    }
}
```

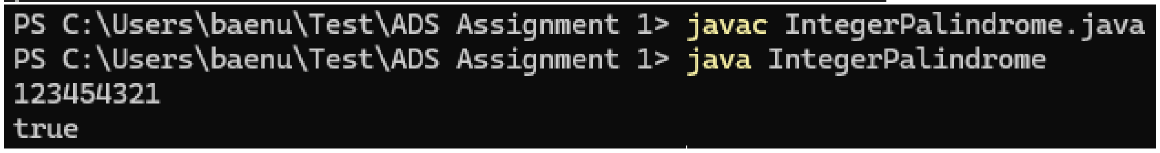


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac FirstNonRepeatedCharacter.java
PS C:\Users\baenu\Test\ADS Assignment 1> java FirstNonRepeatedCharacter
assassination
t
PS C:\Users\baenu\Test\ADS Assignment 1> java FirstNonRepeatedCharacter
redder
null
```

9. Integer Palindrome
Problem: Write a Java program to check if a given integer is a palindrome.
Test Cases:
Input: 121
Output: true
Input: -121
Output: false

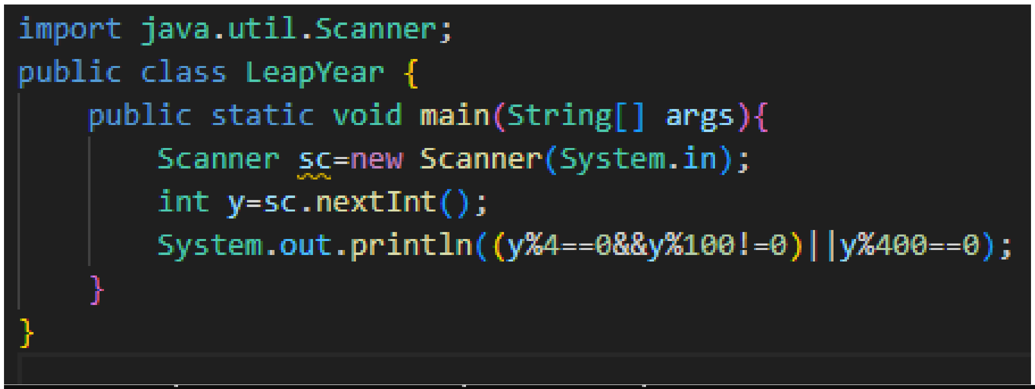


```java
import java.util.Scanner;
public class IntegerPalindrome {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt(),temp=n,rev=0;
        if(n<0){System.out.println(false);return;}
        while(temp>0){rev=rev*10+temp%10;temp/=10;}
        System.out.println(rev==n);
    }
}
```

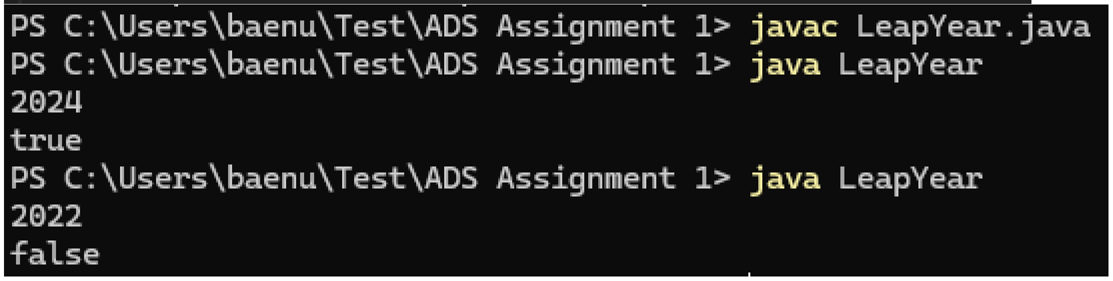


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac IntegerPalindrome.java
PS C:\Users\baenu\Test\ADS Assignment 1> java IntegerPalindrome
123454321
true
```

10. Leap Year
Problem: Write a Java program to check if a given year is a leap year.
Test Cases:
Input: 2020
Output: true
Input: 1900
Output: false

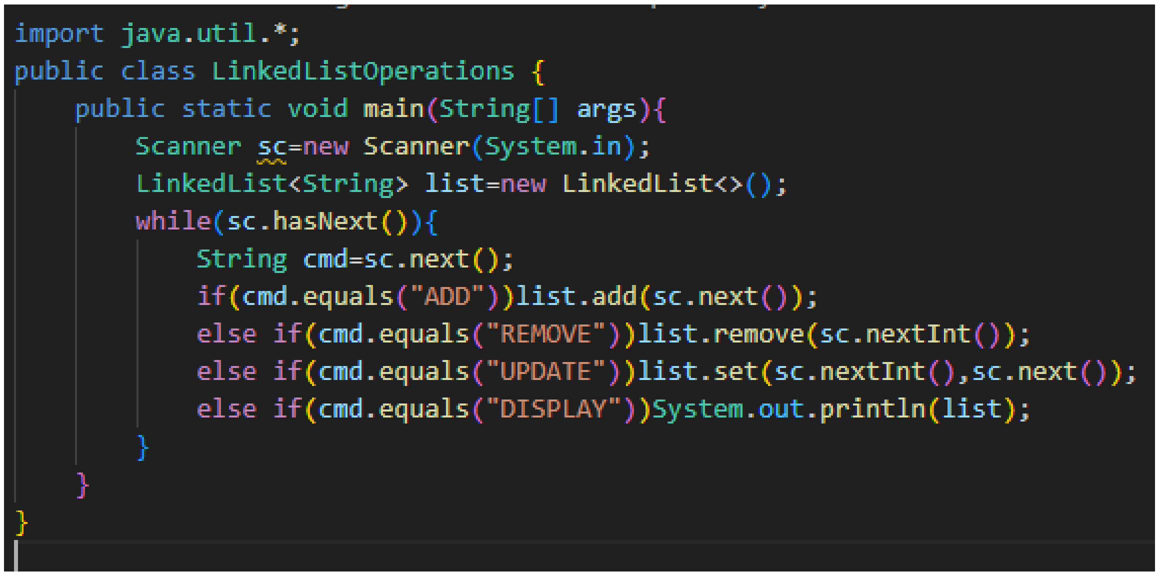


```java
import java.util.Scanner;
public class LeapYear {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int y=sc.nextInt();
        System.out.println((y%4==0&&y%100!=0)||y%400==0);
    }
}
```

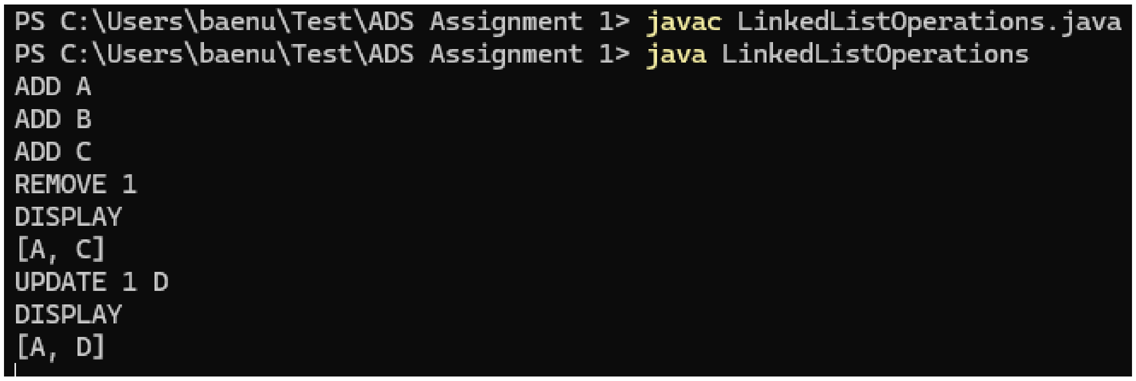


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac LeapYear.java
PS C:\Users\baenu\Test\ADS Assignment 1> java LeapYear
2024
true
PS C:\Users\baenu\Test\ADS Assignment 1> java LeapYear
2022
false
```

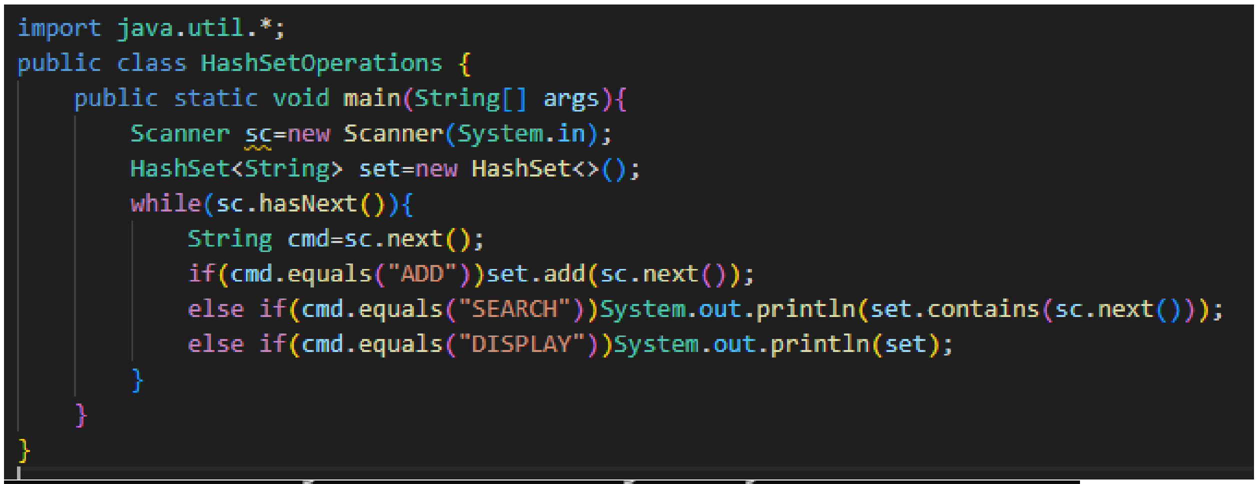
11. Write a Java program to add, update, remove, and display elements using LinkedList.
Testcase:
Input: ADD A
ADD B
ADD C
REMOVE 0
DISPLAY
Output: [B, C]
Input: ADD A
ADD B
ADD C
UPDATE 1 X
DISPLAY
Output: [A, X, C]

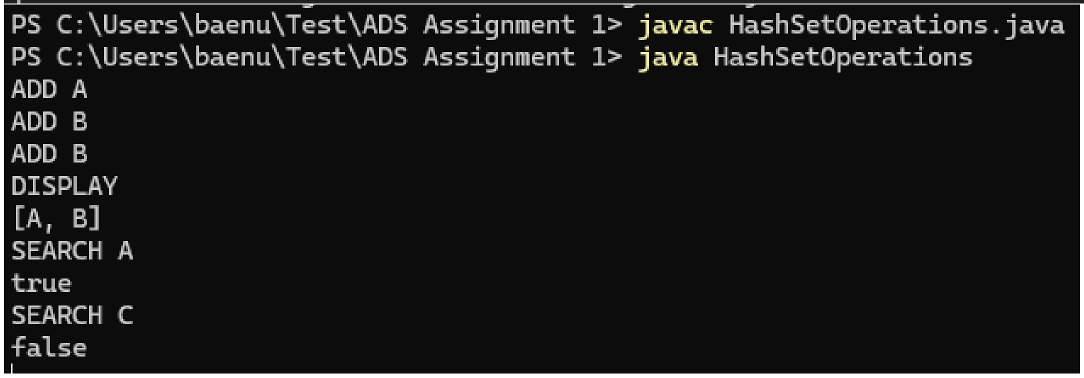


```java
import java.util.*;
public class LinkedListOperations {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        LinkedList<String> list=new LinkedList<>();
        while(sc.hasNext()){
            String cmd=sc.next();
            if(cmd.equals("ADD"))list.add(sc.next());
            else if(cmd.equals("REMOVE"))list.remove(sc.nextInt());
            else if(cmd.equals("UPDATE"))list.set(sc.nextInt(),sc.next());
            else if(cmd.equals("DISPLAY"))System.out.println(list);
        }
    }
}
```

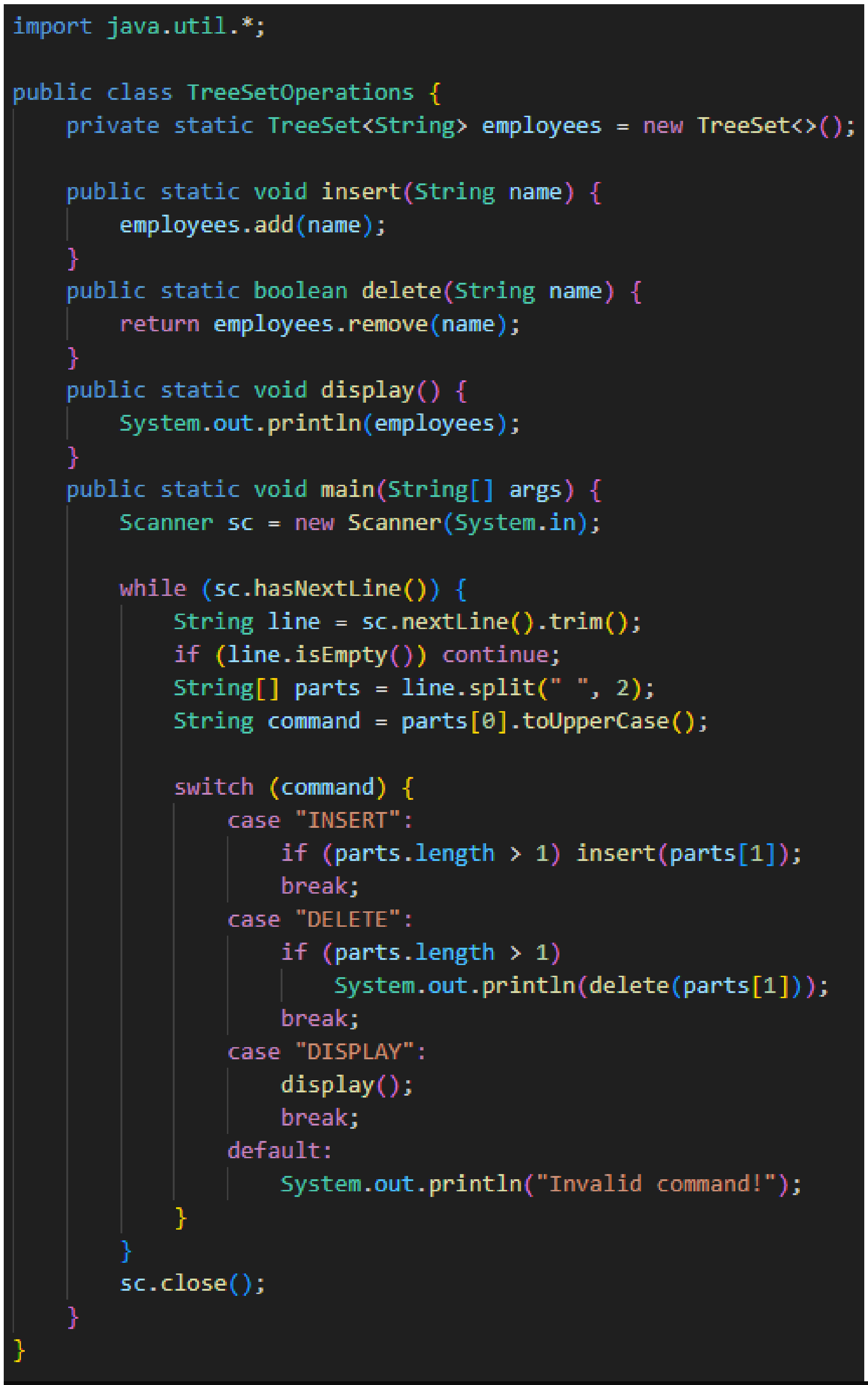

```
PS C:\Users\baenu\Test\ADS Assignment 1> javac LinkedListOperations.java
PS C:\Users\baenu\Test\ADS Assignment 1> java LinkedListOperations
ADD A
ADD B
ADD C
REMOVE 1
DISPLAY
[A, C]
UPDATE 1 D
DISPLAY
[A, D]
```

12. Write a Java program to add, search, remove, and display elements using HashSet.
Testcase:
Input: Add duplicates ignored
ADD A
ADD A
ADD B
DISPLAY
Output: [A, B]
Input: Search present vs absent
ADD A
ADD B
SEARCH A
SEARCH C
Output: true
False

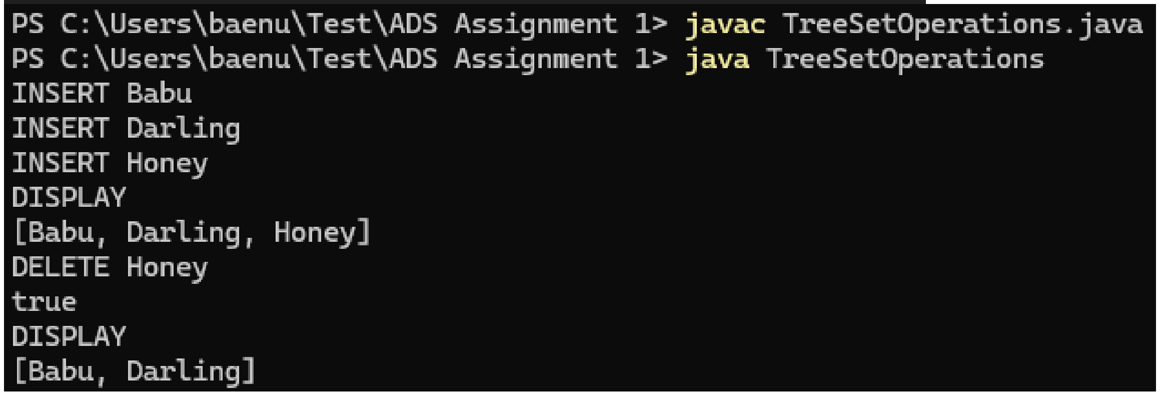

```java
import java.util.*;
public class HashSetOperations {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        HashSet<String> set=new HashSet<>();
        while(sc.hasNext()){
            String cmd=sc.next();
            if(cmd.equals("ADD"))set.add(sc.next());
            else if(cmd.equals("SEARCH"))System.out.println(set.contains(sc.next()));
            else if(cmd.equals("DISPLAY"))System.out.println(set);
        }
    }
}
```


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac HashSetOperations.java
PS C:\Users\baenu\Test\ADS Assignment 1> java HashSetOperations
ADD A
ADD B
ADD B
DISPLAY
[A, B]
SEARCH A
true
SEARCH C
false
```

13. Write a Java program to insert, delete, and display employee names in sorted order using TreeSet.
TestCases:
Input: Basic insert, sorted display, and delete
INSERT Zara
INSERT Aman
INSERT Neha
DISPLAY
DELETE Neha
DISPLAY
Output: [Aman, Neha, Zara]
true
[Aman, Zara]
Input: Duplicates ignored & case sensitivity
INSERT Meera
INSERT meera
INSERT Arjun
INSERT Arjun
DISPLAY
DELETE Rahul
DELETE Meera
DISPLAY
Output: [Arjun, Meera, meera]
false
true
[Arjun, meera]

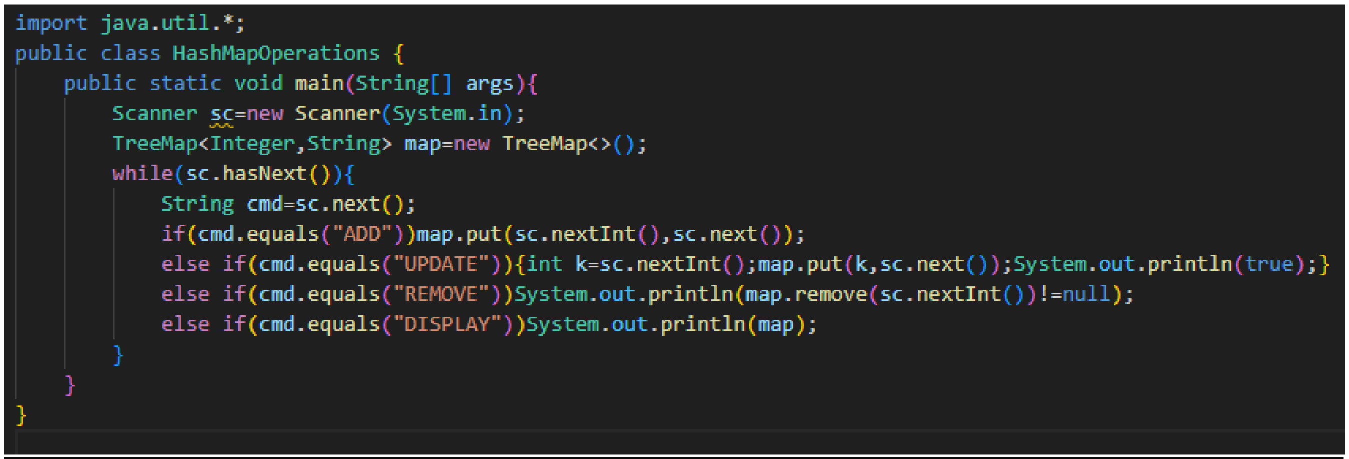


```java
import java.util.*;

public class TreeSetOperations {
    private static TreeSet<String> employees = new TreeSet<>();

    public static void insert(String name) {
        employees.add(name);
    }
    public static boolean delete(String name) {
        return employees.remove(name);
    }
    public static void display() {
        System.out.println(employees);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        while (sc.hasNextLine()) {
            String line = sc.nextLine().trim();
            if (line.isEmpty()) continue;
            String[] parts = line.split(" ", 2);
            String command = parts[0].toUpperCase();

            switch (command) {
                case "INSERT":
                    if (parts.length > 1) insert(parts[1]);
                    break;
                case "DELETE":
                    if (parts.length > 1)
                        System.out.println(delete(parts[1]));
                    break;
                case "DISPLAY":
                    display();
                    break;
                default:
                    System.out.println("Invalid command!");
            }
        }
        sc.close();
    }
}
```

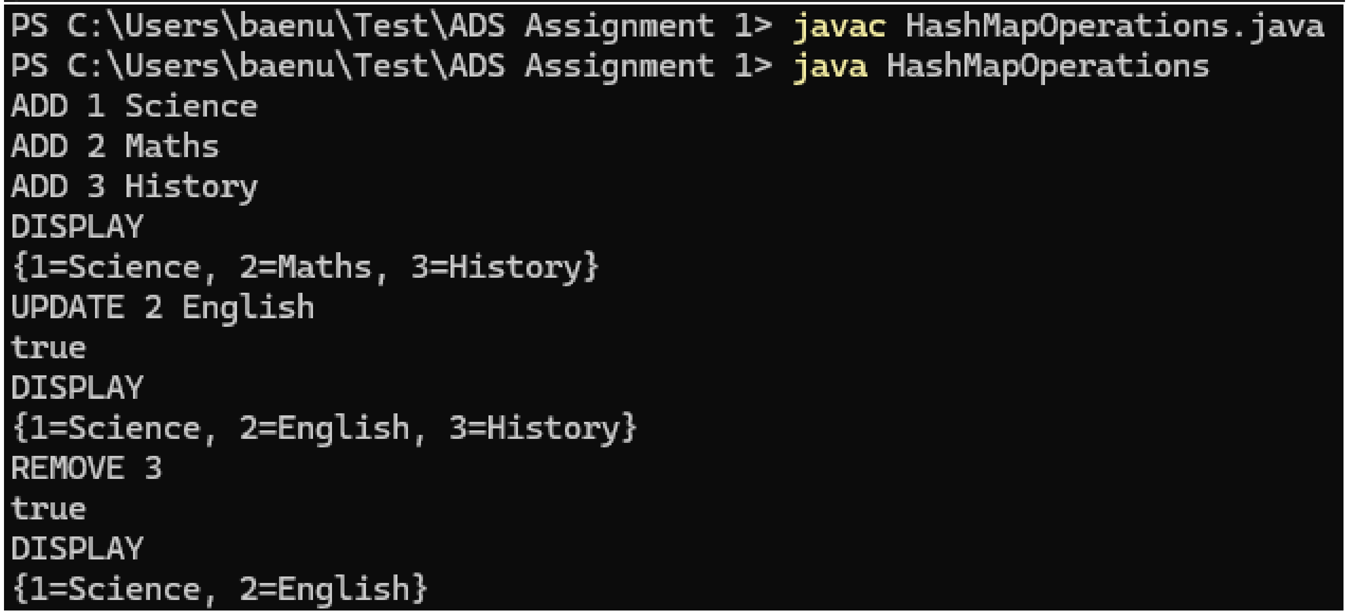


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac TreeSetOperations.java
PS C:\Users\baenu\Test\ADS Assignment 1> java TreeSetOperations
INSERT Babu
INSERT Darling
INSERT Honey
DISPLAY
[Babu, Darling, Honey]
DELETE Honey
true
DISPLAY
[Babu, Darling]
```

14. Write a Java program to add, update, remove, and display books using HashMap.
TestCases:
Input: Basic add & sorted display
ADD 205 Refactoring
ADD 101 Clean_Code
ADD 150 Effective_Java
DISPLAY
Output: {101=Clean_Code, 150=Effective_Java, 205=Refactoring}
Input: Update, remove, and verify
ADD 1 Alpha
ADD 2 Beta
UPDATE 2 Beta_2nd_Ed
REMOVE 1
DISPLAY
Output: true
true
{2=Beta_2nd_Ed}

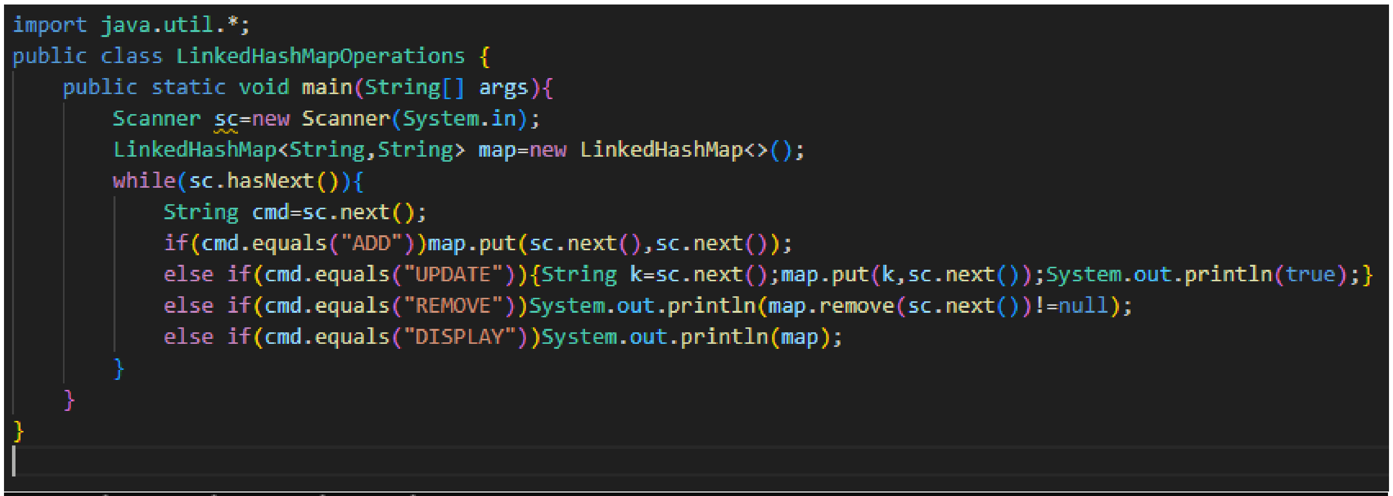


```java
import java.util.*;
public class HashMapOperations {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        TreeMap<Integer,String> map=new TreeMap<>();
        while(sc.hasNext()){
            String cmd=sc.next();
            if(cmd.equals("ADD"))map.put(sc.nextInt(),sc.next());
            else if(cmd.equals("UPDATE")){int k=sc.nextInt();map.put(k,sc.next());System.out.println(true);}
            else if(cmd.equals("REMOVE"))System.out.println(map.remove(sc.nextInt())!=null);
            else if(cmd.equals("DISPLAY"))System.out.println(map);
        }
    }
}
```

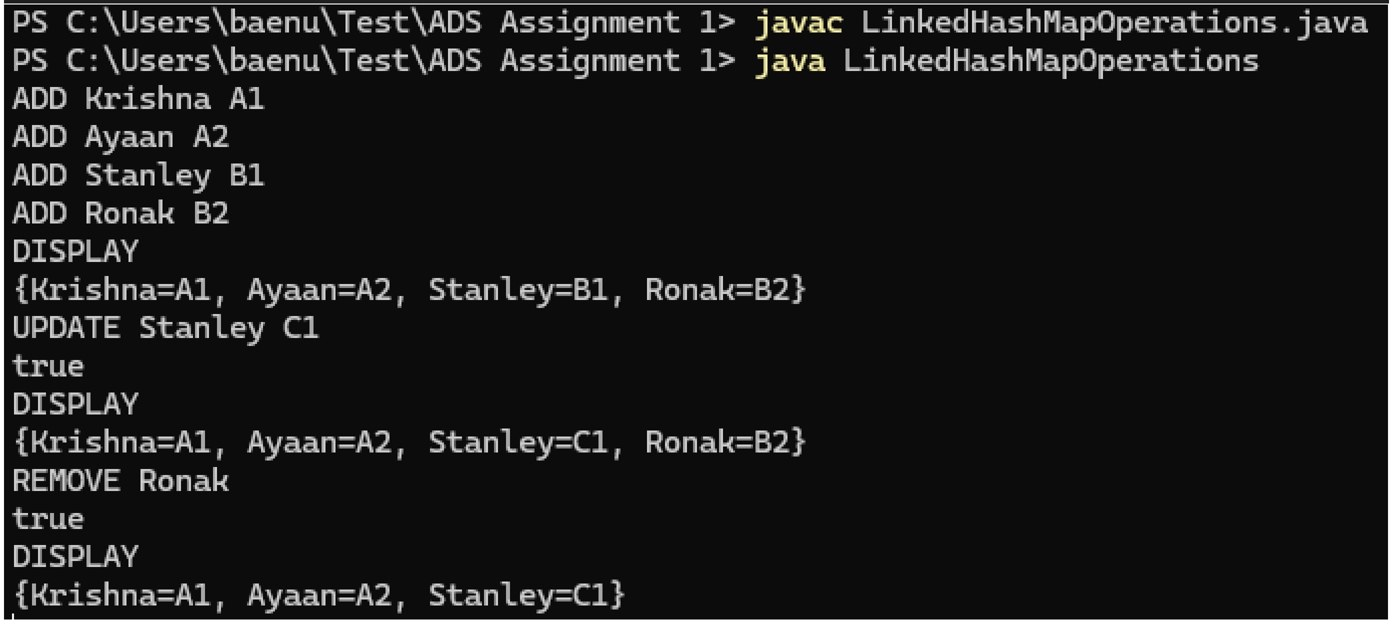


```
PS C:\Users\baenu\Test\ADS Assignment 1> javac HashMapOperations.java
PS C:\Users\baenu\Test\ADS Assignment 1> java HashMapOperations
ADD 1 Science
ADD 2 Maths
ADD 3 History
DISPLAY
{1=Science, 2=Maths, 3=History}
UPDATE 2 English
true
DISPLAY
{1=Science, 2=English, 3=History}
REMOVE 3
true
DISPLAY
{1=Science, 2=English}
```

15. Write a Java program to add, update, remove, and display login details using LinkedHashMap.
TestCases:
Input: Add, update, display (insertion order preserved)
ADD alice a1
ADD bob b1
UPDATE alice a2
DISPLAY
Output: true
{alice=a2, bob=b1}
Input: Remove, re-add (reinserted at end)
ADD alice a1
ADD bob b1
ADD carol c1
REMOVE bob
ADD bob b2
DISPLAY
Output: true
{alice=a1, carol=c1, bob=b2}



```java
import java.util.*;
public class LinkedHashMapOperations {
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        LinkedHashMap<String,String> map=new LinkedHashMap<>();
        while(sc.hasNext()){
            String cmd=sc.next();
            if(cmd.equals("ADD"))map.put(sc.next(),sc.next());
            else if(cmd.equals("UPDATE")){String k=sc.next();map.put(k,sc.next());System.out.println(true);}
            else if(cmd.equals("REMOVE"))System.out.println(map.remove(sc.next())!=null);
            else if(cmd.equals("DISPLAY"))System.out.println(map);
        }
    }
}
```



```
PS C:\Users\baenu\Test\ADS Assignment 1> javac LinkedHashMapOperations.java
PS C:\Users\baenu\Test\ADS Assignment 1> java LinkedHashMapOperations
ADD Krishna A1
ADD Ayaan A2
ADD Stanley B1
ADD Ronak B2
DISPLAY
{Krishna=A1, Ayaan=A2, Stanley=B1, Ronak=B2}
UPDATE Stanley C1
true
DISPLAY
{Krishna=A1, Ayaan=A2, Stanley=C1, Ronak=B2}
REMOVE Ronak
true
DISPLAY
{Krishna=A1, Ayaan=A2, Stanley=C1}
```