**C-DAC Mumbai Date 25/09/2024**

**Subject: Algorithm and Data Structure**

**Assignment 1**

**Solve the assignment with following thing to be added in each question.**

-Program

-Flow chart

-Explanation

-Output

-Time and Space complexity

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

**Program**

**import** java.util.Scanner;

**public** **class** Armstrong {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

**int** temp=n;

**int** sum=0;

**while**(n>0){

**int** rem=n%10;

sum+=Math.*pow*(rem,3);

n=n/10;

}

**if**(sum==temp)

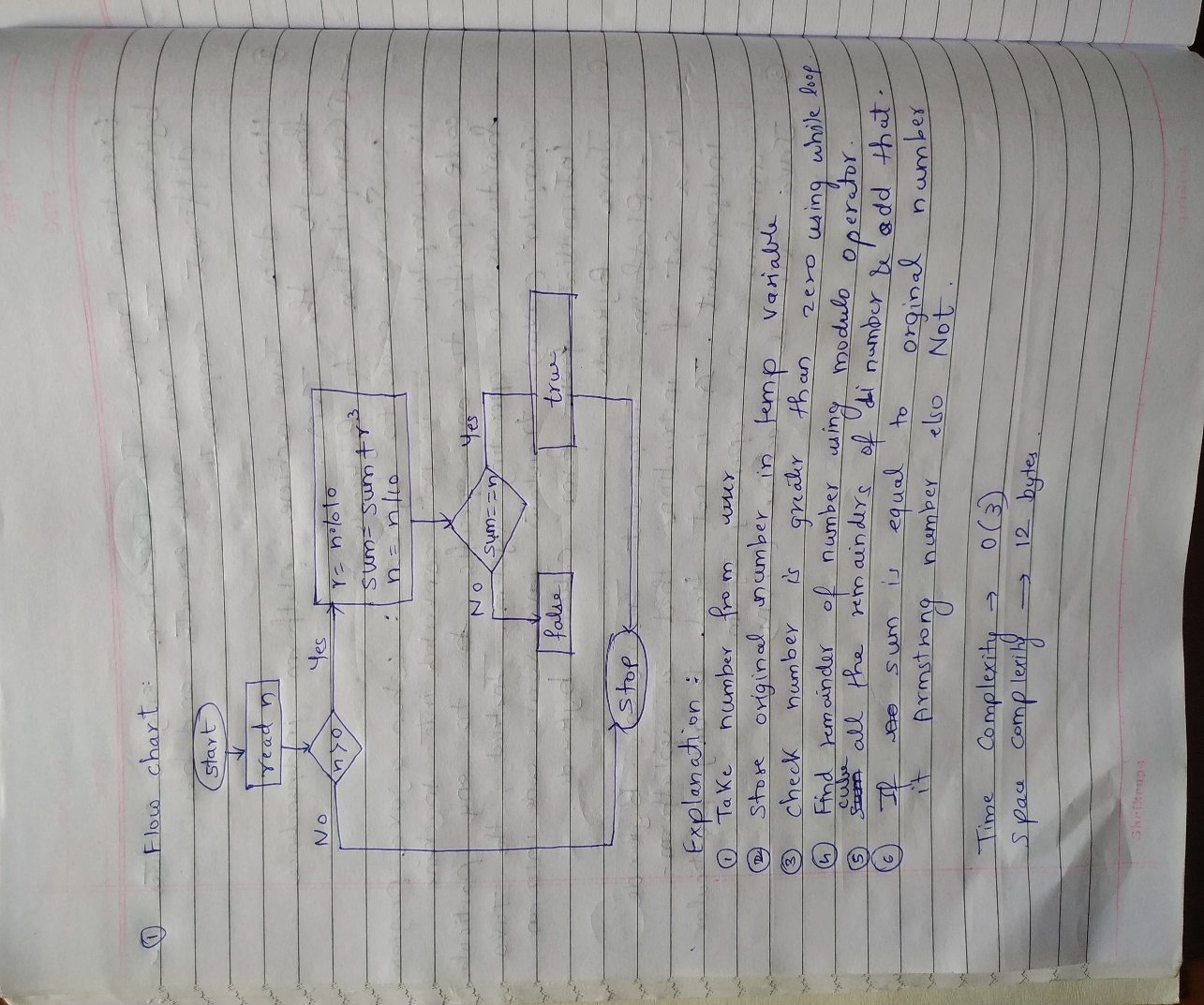
System.***out***.println("true");

**else**

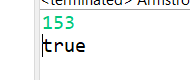
System.***out***.println("false");

}

}



**Output**



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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** PrimeNo {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

**int** cnt=0;

**for**(**int** i=1;i<=n;i++) {

**if**(n%i==0)

cnt++;

}

**if**(cnt>2)

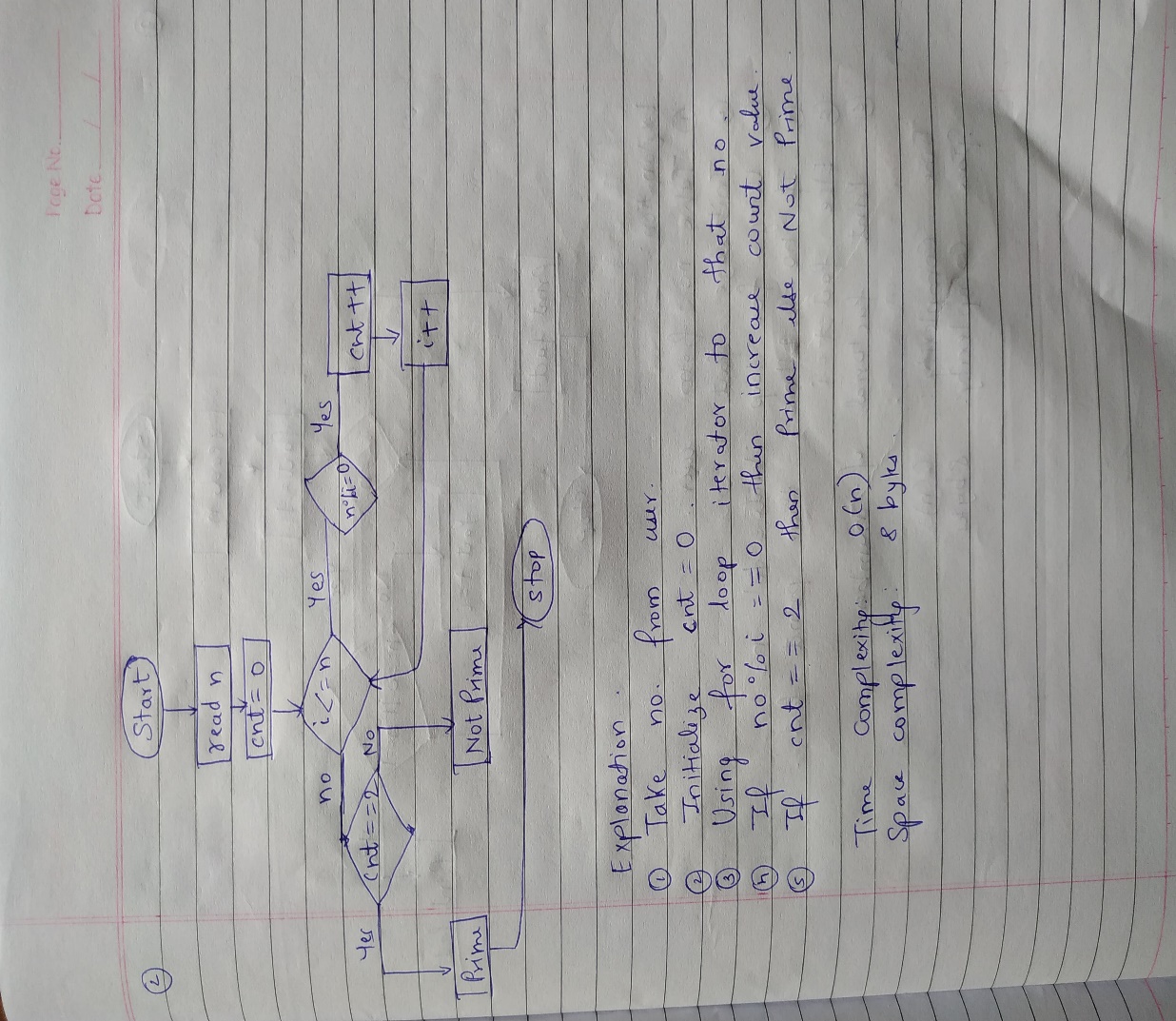
System.***out***.println(**false**);

**else**

System.***out***.println(**true**);

}

}



**Output**



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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Factorial {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

**int** fact=1;

**for**(**int** i=1;i<=n;i++) {

**if**(n==0 || n==1)

System.***out***.println(fact);

**else**

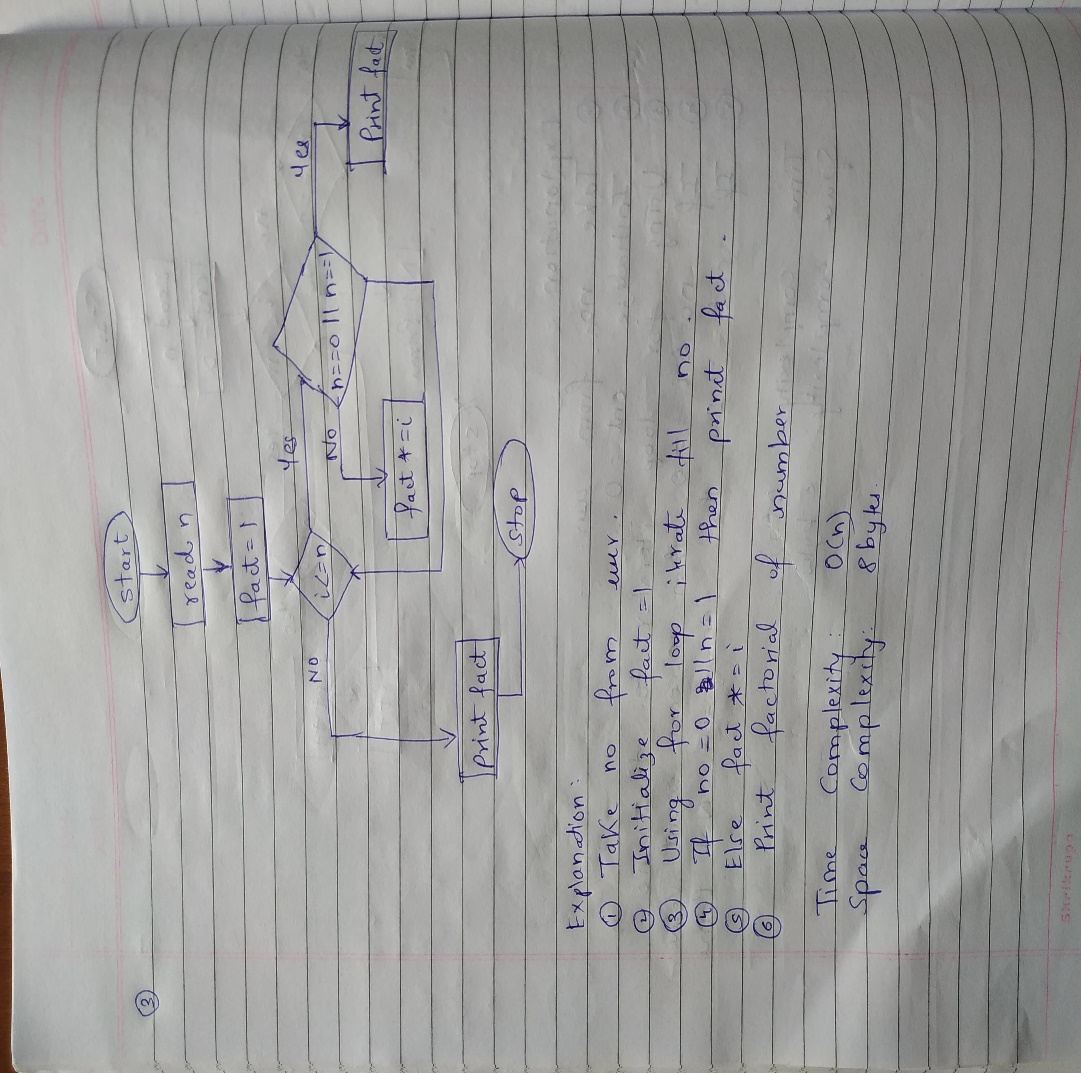
fact\*=i;

}

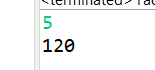
System.***out***.println(fact);

}

}



**Output**

****

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]

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Fibonacci {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

**int** f1=0,f2=1;

System.***out***.print("["+f1);

**int** res=0;

**for**(**int** i=1;i<n;i++) {

f1=f2;

f2=res;

res=f1+f2;

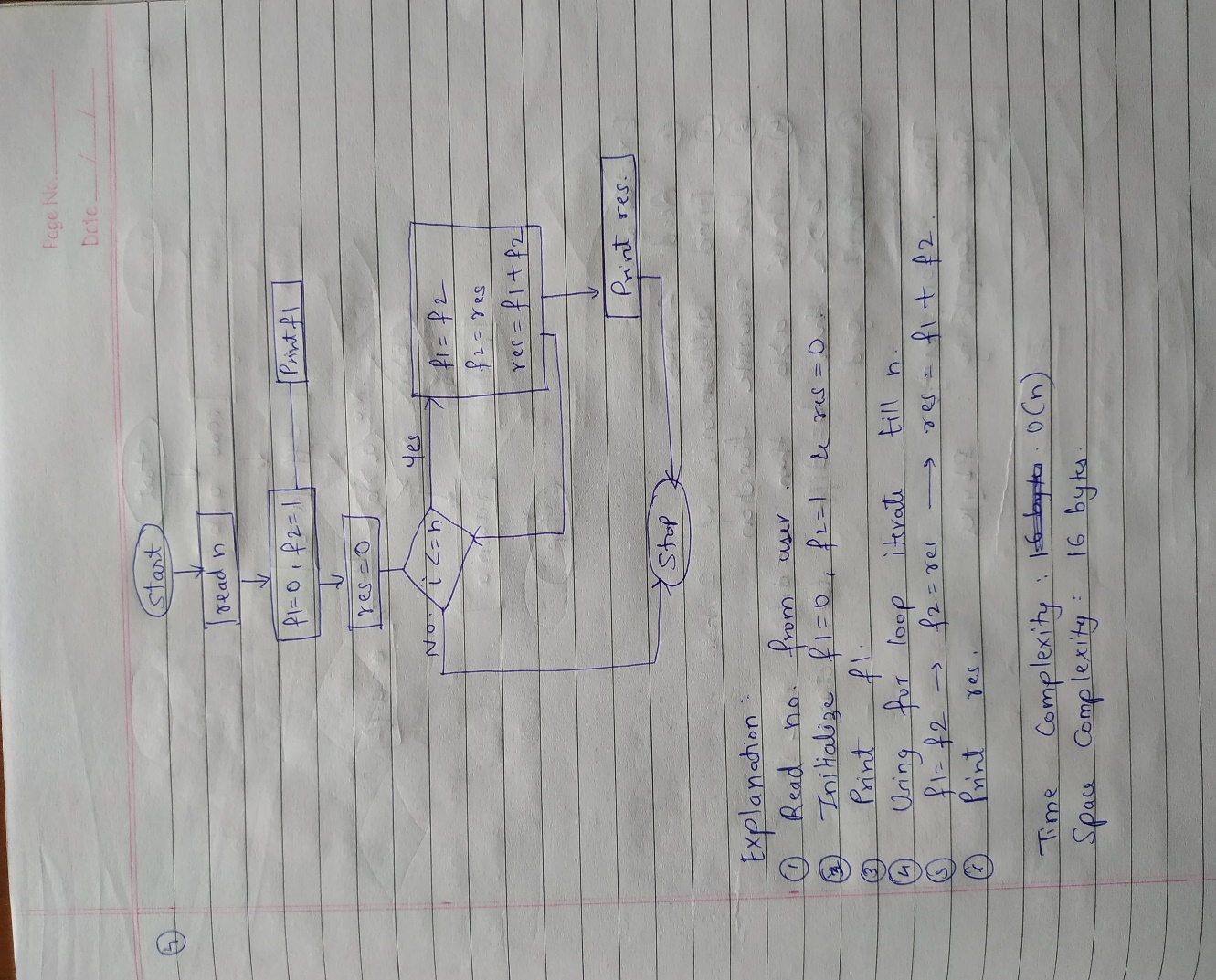
System.***out***.print(","+res);

}

System.***out***.println("]");

}

}



**Output**

****

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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Gcd {

**public** **static** **int** gcd(**int** a,**int** b) {

**int** diff=Math.*abs*(a-b);

**if**(a>b)

a=diff;

**else** **if**(b>a)

b=diff;

**else**

**return** a;

**return** *gcd*(a,b);

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

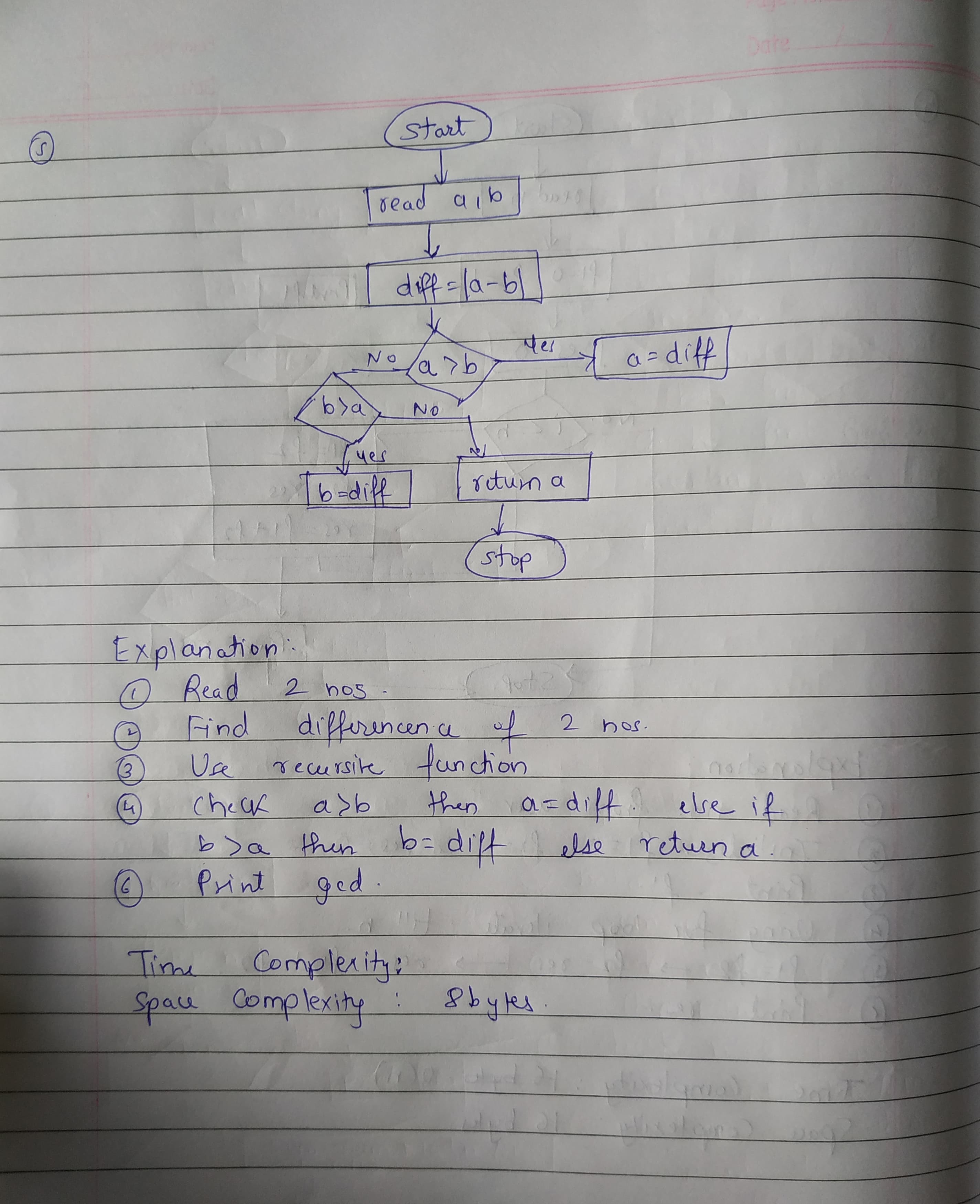
**int** n1=sc.nextInt();

**int** n2=sc.nextInt();

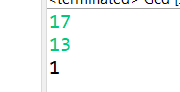
System.***out***.println(*gcd*(n1,n2));

}

}



**Output**

****

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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Squareroot {

**public** **static** **void** main(String[] args) {

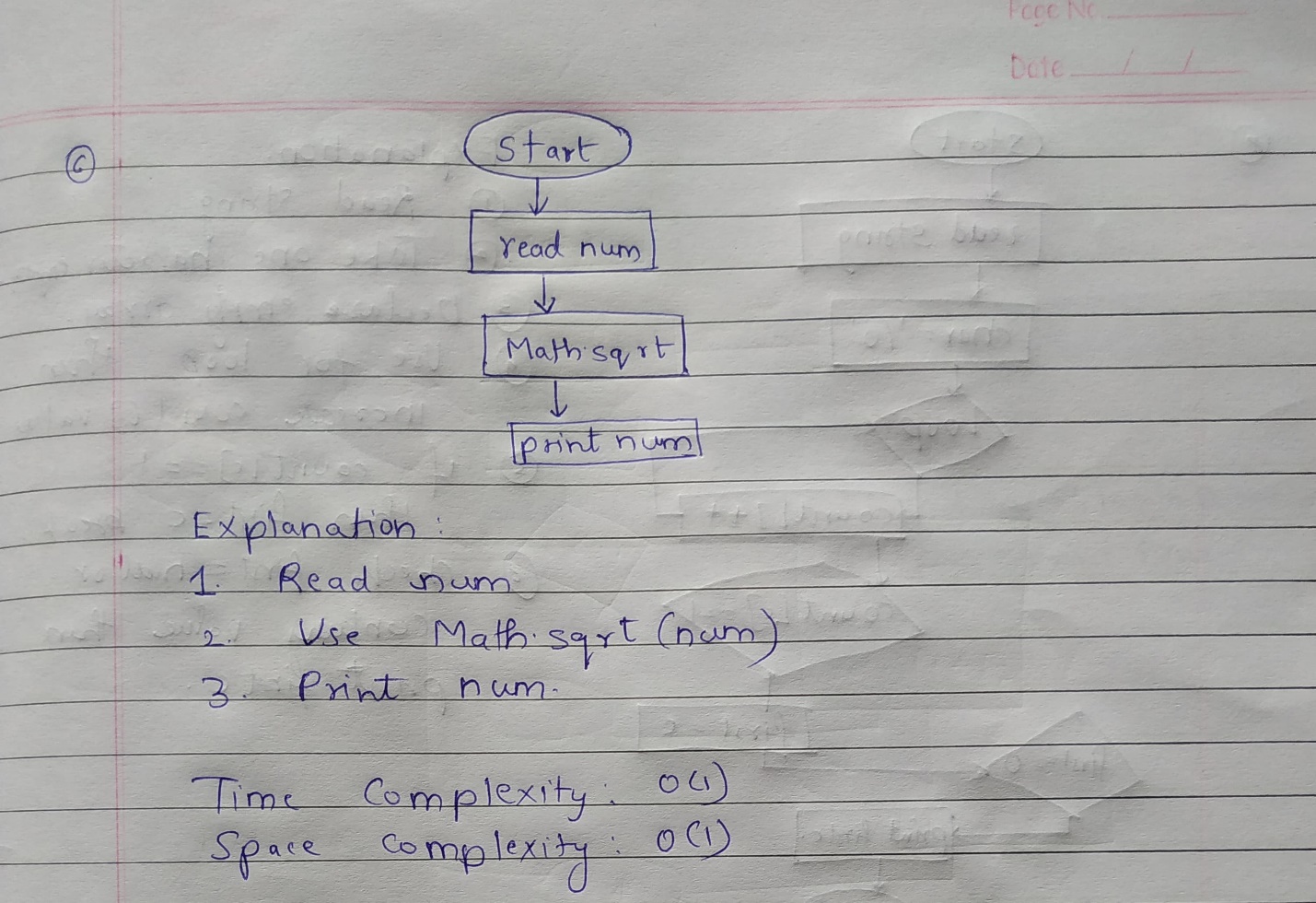
Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

System.***out***.println((**int**)Math.*sqrt*(n));

}

}



**Output**



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']

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** RepCharString {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

String s=sc.nextLine();

**int** count[]=**new** **int**[256];

**for**(**char** c :s.toCharArray()) {

count[c]++;

}

**for**(**int** i=0;i<count.length;i++) {

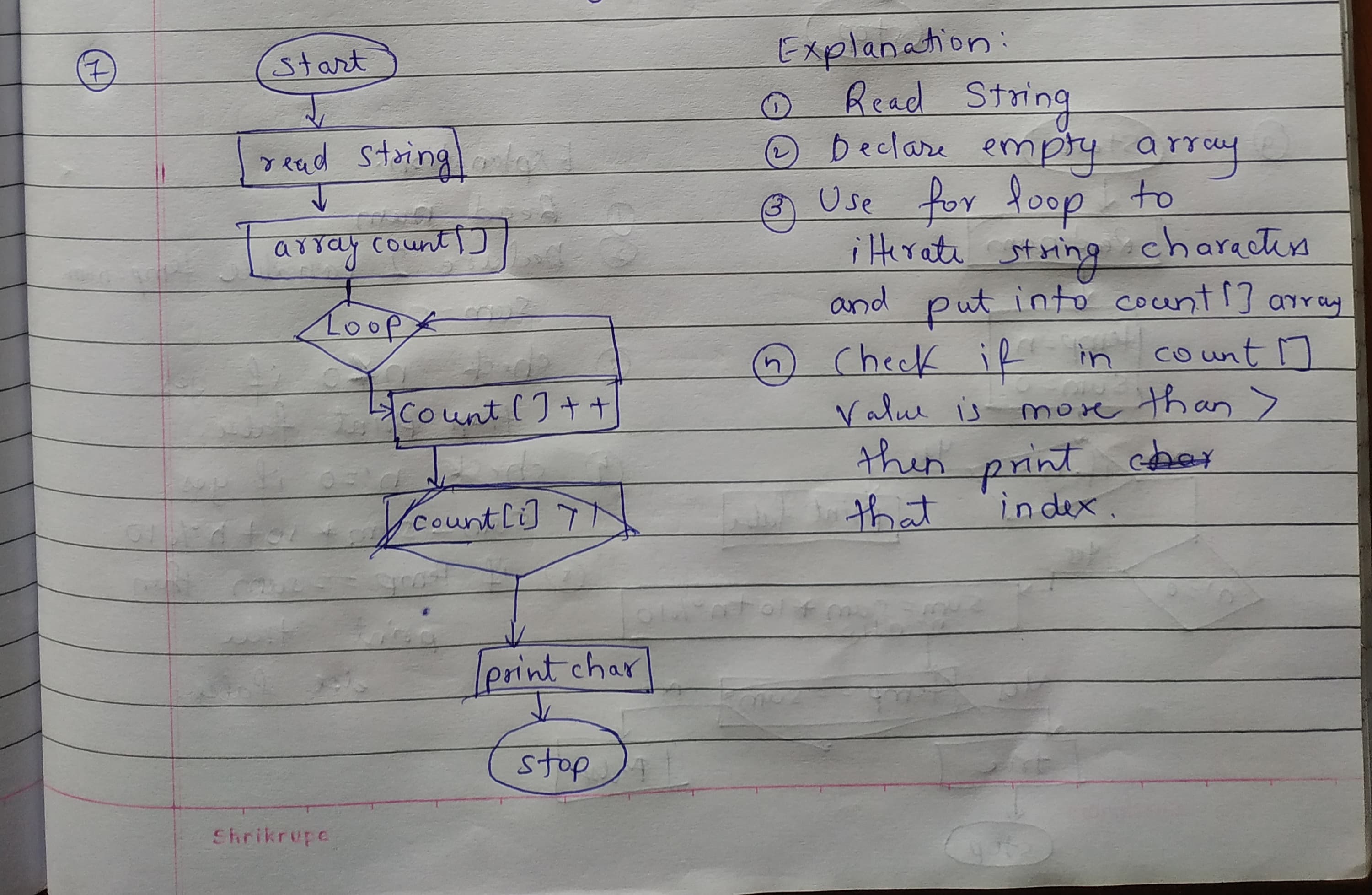
**if**(count[i]>1)

System.***out***.println((**char**)i+" ");

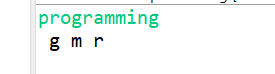
}

}

}



**Output**



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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** NonRepChar {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

String s=sc.nextLine();

**char** firstc='\0';

**int** count[]=**new** **int**[256];

**for**(**char** c :s.toCharArray()) {

count[c]++;

}

**for**(**char** c :s.toCharArray()) {

**if**(count[c]==1) {

firstc=c;

**break**;

}

}

**if** (firstc == '\0') {

System.***out***.println("Output: null");

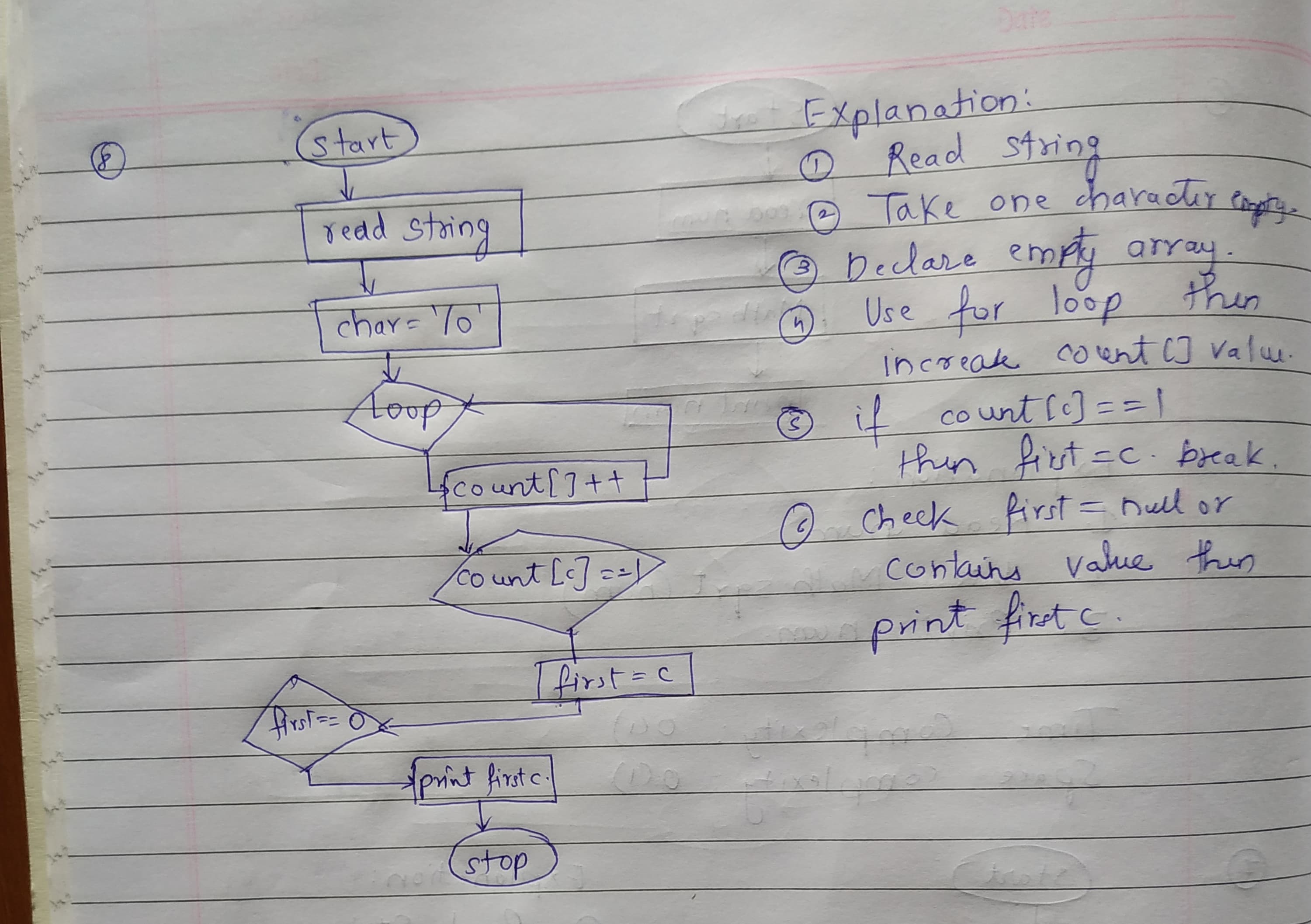
} **else** {

System.***out***.println("Output: " + firstc);

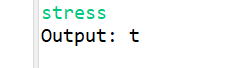
}

}

}



**Output**



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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Palindrome {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

**int** n=sc.nextInt();

**int** temp=n;

**int** sum=0;

**if**(n<0) {

System.***out***.println(**false**);

**return**;

}

**while**(n!=0) {

sum=sum\*10+n%10;

n=n/10;

}

**if**(temp==sum)

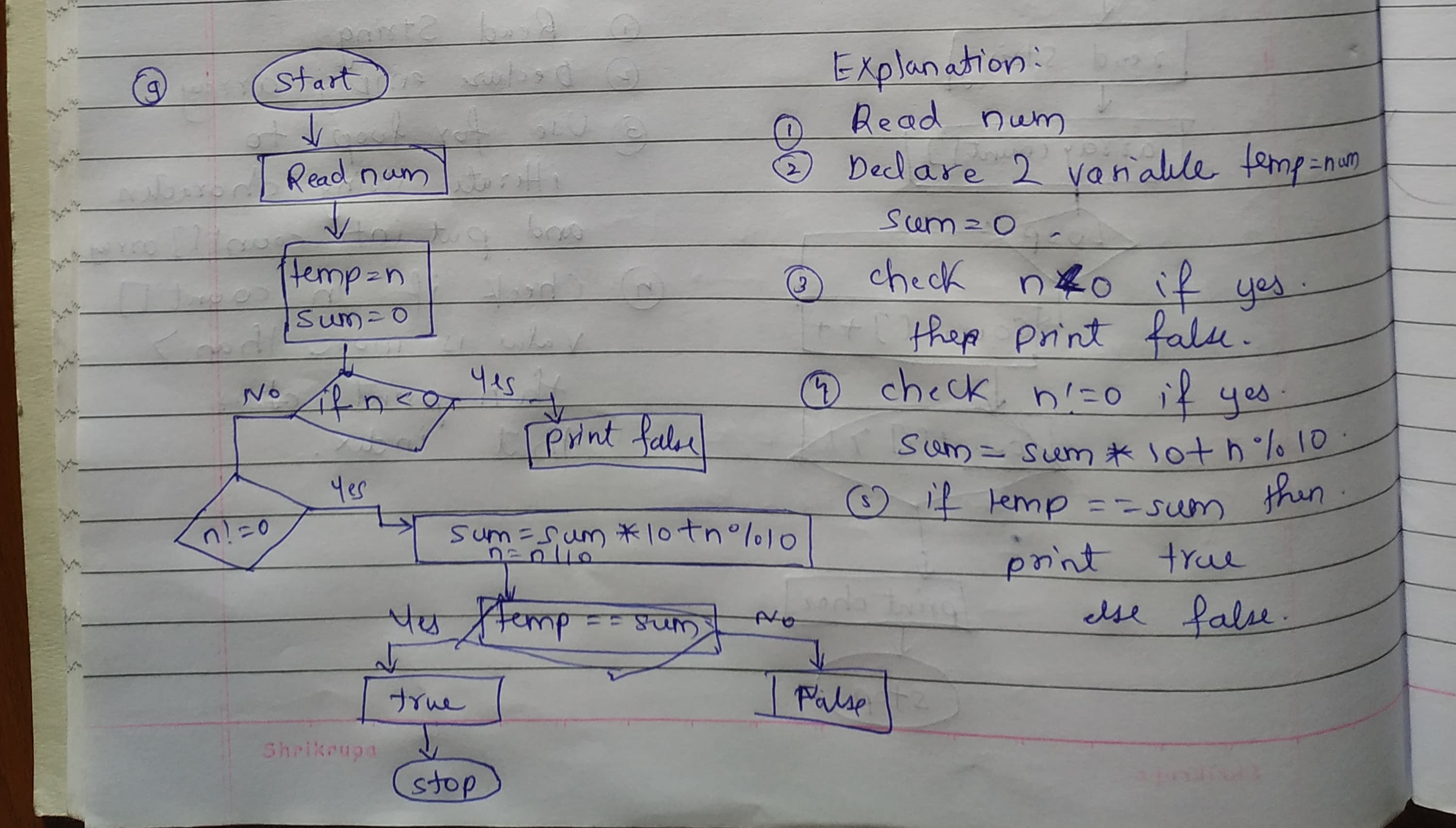
System.***out***.println(**true**);

**else**

System.***out***.println(**false**);

}

}



**Output**



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

**Program**

**package** Assignment1;

**import** java.util.Scanner;

**public** **class** Leapyear {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=**new** Scanner(System.***in***);

**int** y=sc.nextInt();

**if**((y%4==0 && y%100!=0) || y%400==0) {

System.***out***.println("true");

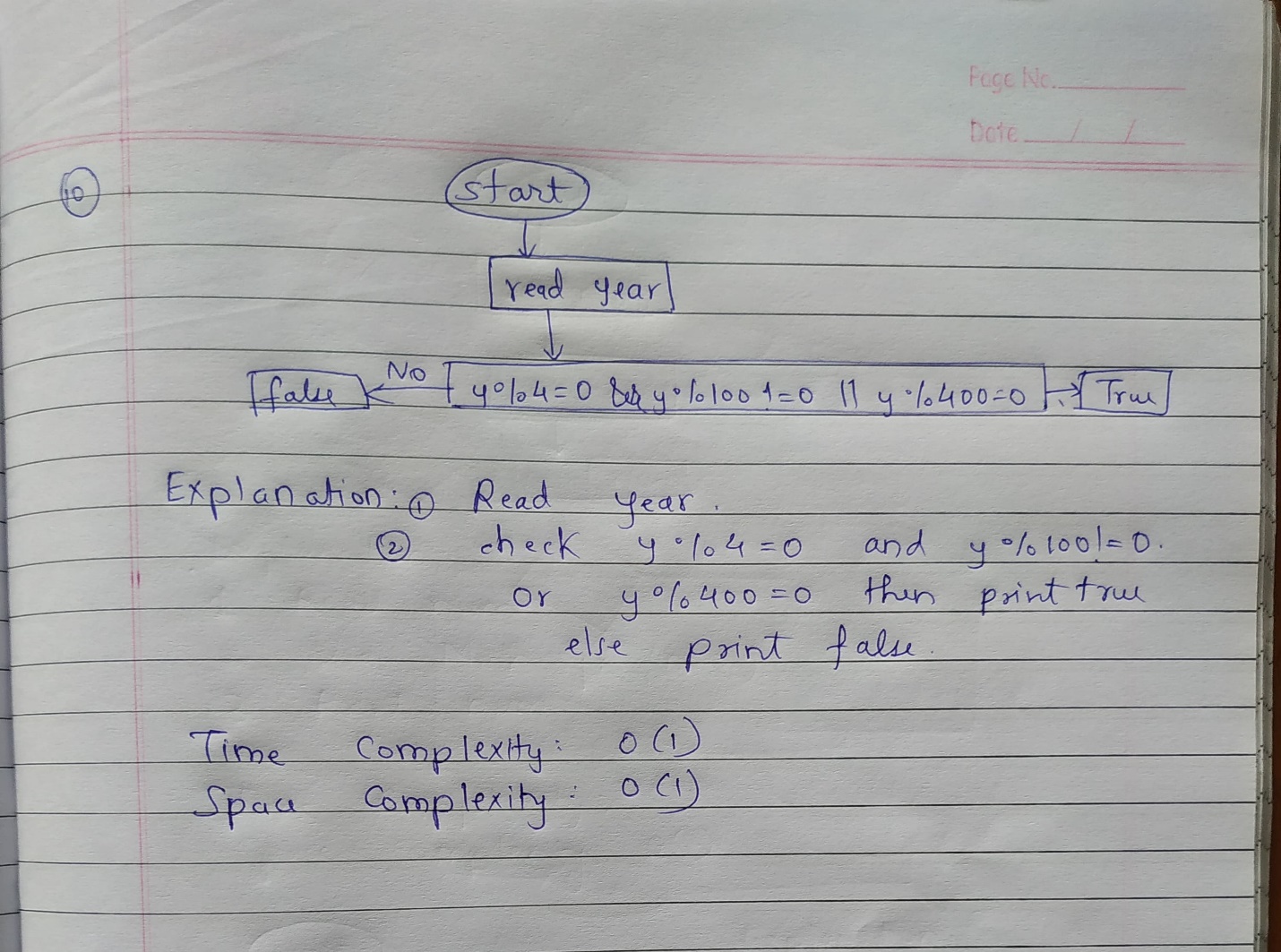
}

**else**

System.***out***.println("false");

}

}



**Output**

