All Lab Programs

Shreyansh Ranpariya -24010101679

Lab 1

1 NoIsPostiveOrNot.c

```c
#include <stdio.h>
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (num > 0)
        printf("Positive number.\n");
    else if (num < 0)
        printf("Negative number.\n");
    else
        printf("Zero.\n");

    return 0;
}
```

2 MaxOfThree.c

```c
#include <stdio.h>
int main() {
    int a, b, c;
    printf("Enter three numbers: ");
    scanf("%d %d %d", &a, &b, &c);

    if (a >= b && a >= c)
        printf("Maximum is: %d\n", a);
    else if (b >= a && b >= c)
        printf("Maximum is: %d\n", b);
    else
        printf("Maximum is: %d\n", c);

    return 0;
}
```

3 GetDay.c

```c
#include <stdio.h>
int main() {
    int day;
    printf("Enter day number (1 to 7): ");
    scanf("%d", &day);
    switch(day) {
        case 1: printf("Sunday\n"); break;
        case 2: printf("Monday\n"); break;
        case 3: printf("Tuesday\n"); break;
        case 4: printf("Wednesday\n"); break;
        case 5: printf("Thursday\n"); break;
        case 6: printf("Friday\n"); break;
        case 7: printf("Saturday\n"); break;
        default: printf("Invalid input\n");
    }

    return 0;
}
```

4 Print1ToN.c

```
#include <stdio.h>
int main() {
    int i, n;
    printf("Enter N: ");
    scanf("%d", &n);

    printf("Using for loop:\n");
    for(i = 1; i <= n; i++)
        printf("%d ", i);
    printf("\n");

    printf("Using while loop:\n");
    i = 1;
    while(i <= n) {
        printf("%d ", i);
        i++;
    }
    printf("\n");

    printf("Using do-while loop:\n");
    i = 1;
    do {
        printf("%d ", i);
        i++;
    } while(i <= n);
    printf("\n");

    return 0;
}
```

5 PrimeOrNot.c

```
#include <stdio.h>
int main() {
    int num, i, isPrime = 1;
    printf("Enter a number: ");
    scanf("%d", &num);
```

```c
    if (num <= 1)
        isPrime = 0;
    else {
        for(i = 2; i <= num/2; i++) {
            if (num % i == 0) {
                isPrime = 0;
                break;
            }
        }
    }

    if (isPrime)
        printf("Prime number.\n");
    else
        printf("Not a prime number.\n");

    return 0;
}
```

6 CheckPalindrome.c

```c
#include <stdio.h>
int main() {
    int num, reversed = 0, original, remainder;
    printf("Enter a number: ");
    scanf("%d", &num);
    original = num;

    while(num != 0) {
        remainder = num % 10;
        reversed = reversed * 10 + remainder;
        num /= 10;
    }

    if (original == reversed)
        printf("Palindrome number.\n");
    else
        printf("Not a palindrome.\n");

    return 0;
}
```

7 CheckArmstrong.c

```c
#include <stdio.h>
#include <math.h>

int main() {
    int num, original, sum = 0, digit, n = 0, temp;
    printf("Enter a number: ");
    scanf("%d", &num);
    original = num;
    temp = num;

    while (temp != 0) {
        temp /= 10;
        n++;
    }

    temp = num;
    while (temp != 0) {
        digit = temp % 10;

        // Manual   power = 1;
        // for (i = 0; i < n; i++) {
        //     power *= digit;
        // }
        // sum += power;

        // Using pow function
        sum += pow(digit, n);
        temp /= 10;
    }

    if (sum == original)
        printf("Armstrong number.\n");
    else
        printf("Not an Armstrong number.\n");

    return 0;
}
```

Lab 2

1 ReadnNoAndPrint.c

```c
#include <stdio.h>

int main() {
    int n, i;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d numbers:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Normal Order:\n");
    for(i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\nReverse Order:\n");
    for(i = n - 1; i >= 0; i--) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

2 Sum&Avg.c

```c
#include <stdio.h>

int main() {
    int n, i, sum = 0;
    float avg;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d numbers:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }

    avg =sum / n;
    printf("Sum = %d\n", sum);
    printf("Average = %.2f\n", avg);

    return 0;
}
```

3 Multiply2Matrics.c

```c
#include <stdio.h>

int main() {
    int a[10][10], b[10][10], axb[10][10];
    int r1, c1, r2, c2, i, j, k;

    printf("Enter rows and columns of matrix A: ");
    scanf("%d %d", &r1, &c1);
    printf("Enter rows and columns of matrix B: ");
    scanf("%d %d", &r2, &c2);

    if(c1 != r2) {
        printf("Matrix multiplication not possible.\n");
        return 0;
    }

    printf("Enter elements of matrix A:\n");
    for(i = 0; i < r1; i++)
```

```
        for(j = 0; j < c1; j++)
            scanf("%d", &a[i][j]);

    printf("Enter elements of matrix B:\n");
    for(i = 0; i < r2; i++)
        for(j = 0; j < c2; j++)
            scanf("%d", &b[i][j]);

    // Multiplication
    for(i = 0; i < r1; i++) {
        for(j = 0; j < c2; j++) {
            axb[i][j] = 0;
            for(k = 0; k < c1; k++) {
                axb[i][j] += a[i][k] * b[k][j];
            }
        }
    }

    printf("Resultant Matrix:\n");
    for(i = 0; i < r1; i++) {
        for(j = 0; j < c2; j++) {
            printf("%d ", axb[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

4 LengthOfString.c

```
#include <stdio.h>

int main() {
    char str[100];  // Array to store user input string
    int i = 0;

    printf("Enter a string: ");
    scanf("%[^\n]s", str);  // Read string until newline

    while(str[i] != '\0') {
        i++;
    }

    printf("Length of string = %d\n", i);
    return 0;
}
```

5 Max&Min.c

```
#include <stdio.h>

int main() {
    int n, i, max, min;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d numbers:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    max = min = arr[0];

    for(i = 1; i < n; i++) {
        if(arr[i] > max)
            max = arr[i];
        if(arr[i] < min)
            min = arr[i];
    }

    printf("Maximum = %d\n", max);
    printf("Minimum = %d\n", min);

    return 0;
}
```

6 CopyString.c

```
#include <stdio.h>

int main() {
    char str1[100], str2[100];
    int i = 0;

    printf("Enter a string: ");
    fgets(str1, sizeof(str1), stdin);
    while(str1[i] != '\0') {
        str2[i] = str1[i];
        i++;
    }

    str2[i] = '\0';

    printf("Copied string: %s\n", str2);
    return 0;
}
```

Lab 3

Address.java

```
public class Address {
    public static void main(String[] args) {
```

```java
        System.out.print("123 str, Jaipur, Rajasthan, India");

        System.out.println();

        System.out.println("123 Main Street");
        System.out.println("Jaipur");
        System.out.println("Rajasthan");
        System.out.println("India");
    }
}
```

CircleArea.java

```java
public class CircleArea{
    public static void main(String[] args) {
        double r = 5.0;
        double area = 3.14 * r * r;
        System.out.println("Area of Circle: " + area);
    }
}
```

SimpleInterest.java

```java
public class SimpleInterest {
    public static void main(String[] args) {
        double principal = 10000;
        double roi = 5;
        double time = 2;
        double interest = (principal * roi * time) / 100;
        System.out.println("Simple Interest: " + interest);
    }
}
```

Welcome.java

```java
public class Welcome {
    public static void main(String[] args) {
        System.out.println("Welcome to Java");
    }
}
```

Lab 4

Addition.java

```java
import java.util.Scanner;

public class Addition {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int sum = a + b;
        System.out.println(sum);
        sc.close();
    }
}
```

BMI.java

```java
import java.util.Scanner;

public class BMI {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double pounds = sc.nextDouble();
        double inches = sc.nextDouble();
        double weightKg = pounds * 0.45359237;
        double heightM = inches * 0.0254;
        double bmi = weightKg / (heightM * heightM);
        System.out.println(bmi);
        sc.close();
    }
}
```

FahrenheitToCelsius.java

```java
import java.util.Scanner;

public class FahrenheitToCelsius {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double f = sc.nextDouble();
        double c = (f - 32) * 5 / 9;
        System.out.println(c);
        sc.close();
    }
}
```

MetersToFeet.java

```java
import java.util.Scanner;

public class MetersToFeet {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double meters = sc.nextDouble();
        double feet = meters * 3.28084;
        System.out.println(feet);
        sc.close();
    }
}
```

Lab 5

BitwiseOperator.java

```java
import java.util.Scanner;

public class BitwiseOperator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        System.out.println(a & b);
        System.out.println(a | b);
        System.out.println(a ^ b);
        System.out.println(~a);
        System.out.println(a << 1);
        System.out.println(a >> 1);
    }
}
```

MaxOfThree.java

```java
import java.util.Scanner;

public class MaxOfThree {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();
        int max = (a > b) ? (a > c ? a : c) : (b > c ? b : c);
        System.out.println(max);
    }
}
```

OddOrEven.java

```java
import java.util.Scanner;

public class OddOrEven {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        String result = (num % 2 == 0) ? "Even" : "Odd";
        System.out.println(result);
    }
}
```

PositiveOrNegative.java

```java
import java.util.Scanner;
```

```java
public class PositiveOrNegative {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        String result = (num >= 0) ? "Positive" : "Negative";
        System.out.println(result);
    }
}
```

TypeCasting.java

```java
import java.util.Scanner;

public class TypeCasting {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int i = sc.nextInt();
        double d = i;
        double pi = sc.nextDouble();
        int x = (int) pi;
        System.out.println(d);
        System.out.println(x);
    }
}
```

Lab 6

GradeCalculator.java

```java
import java.util.Scanner;

public class GradeCalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int total = 0;
        for (int i = 0; i < 5; i++) {
            total += sc.nextInt();
        }
        double percentage = total / 5.0;
        if(percentage<0 || percentage > 100) {
            System.out.println("Invalid percentage");
            return;
        }
        else if (percentage < 35) {
            System.out.println("Fail");
        } else if (percentage < 45) {
            System.out.println("Pass Class");
        } else if (percentage < 60) {
            System.out.println("Second Class");
        } else if (percentage < 70) {
            System.out.println("First Class");
        } else {
            System.out.println("Distinction");
        }
    }
}
```

LargestNumber.java

```
import java.util.Scanner;

public class LargestNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();
        if (a > b) {
            if (a > c) {
                System.out.println(a);
            } else {
                System.out.println(c);
            }
        } else {
            if (b > c) {
                System.out.println(b);
            } else {
                System.out.println(c);
            }
        }
    }
}
```

LeapYear.java

```
import java.util.Scanner;

public class LeapYear {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int year = sc.nextInt();
        if ((year % 400 == 0) || (year % 4 == 0 && year % 100 != 0)) {
            System.out.println("Leap Year");
        } else {
            System.out.println("Not Leap Year");
        }
    }
}
```

SimpleCalculator.java

```
import java.util.Scanner;

public class SimpleCalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        char op = sc.next().charAt(0);
        switch (op) {
            case '+':
                System.out.println(a + b);
                break;
            case '-':
                System.out.println(a - b);
                break;
            case '*':
                System.out.println(a * b);
                break;
            case '/':
                if (b != 0) System.out.println(a / b);
                else System.out.println("Cannot divide by zero");
                break;
            default:
                System.out.println("Invalid operator");
        }
    }
}
```

VowelOrConsonant.java

```
import java.util.Scanner;

public class VowelOrConsonant {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        char ch = sc.next().charAt(0);
        ch = Character.toLowerCase(ch);
        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
            System.out.println("Vowel");
        } else {
            System.out.println("Consonant");
        }
    }
}
```

Lab 7

Factorial.java

```
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;
```

```java
        for (int i = 1; i <= n; i++) {
            fact *= i;
        }
        System.out.println(fact);
    }
}
```

Pattern1.java

```java
public class Pattern1 {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

Pattern2.java

```java
public class Pattern2 {
    public static void main(String[] args) {
        for (int i = 5; i >= 1; i--) {
            for (int j = 1; j <= i; j++) {
                System.out.print(j);
            }
            System.out.println();
        }
    }
}
```

Pattern3.java

```java
public class Pattern3 {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            for (int j = 1; j <= i; j++) {
                if ((i + j) % 2 == 0) System.out.print("1 ");
                else System.out.print("0 ");
            }
            System.out.println();
        }
    }
}
```

Pattern4.java

```java
public class Pattern4 {
    public static void main(String[] args) {
        int num = 1;
        for (int i = 1; i <= 5; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(num + " ");
                num++;
            }
            System.out.println();
        }
    }
}
```

PrimeCheck.java

```
import java.util.Scanner;

public class PrimeCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        boolean isPrime = true;
        if (n <= 1) {
            isPrime = false;
        } else {
            for (int i = 2; i <= n / 2; i++) {
                if (n % i == 0) {
                    isPrime = false;
                    break;
                }
            }
        }
        System.out.println(isPrime ? "Prime" : "Not Prime");
    }
}
```

PrintNumbers1to10.java

```
public class PrintNumbers1to10 {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            System.out.print(i + " ");
        }
        System.out.println();

        int j = 1;
        while (j <= 10) {
            System.out.print(j + " ");
            j++;
        }
        System.out.println();

        int k = 1;
        do {
            System.out.print(k + " ");
            k++;
        } while (k <= 10);
    }
}
```

ReverseNumber.java

```java
import java.util.Scanner;

public class ReverseNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int rev = 0;
        while (n != 0) {
            rev = rev * 10 + n % 10;
            n /= 10;
        }
        System.out.println(rev);
    }
}
```

SumOfNNumbers.java

```java
import java.util.Scanner;

public class SumOfNNumbers {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();

        int sumFor = 0;
        for (int i = 1; i <= n; i++) {
            sumFor += i;
        }
        System.out.println(sumFor);

        int sumWhile = 0;
        int i = 1;
        while (i <= n) {
            sumWhile += i;
            i++;
        }
        System.out.println(sumWhile);
    }
}
```

Lab 8

BreakAtFive.java

```java
public class BreakAtFive {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5) break;
            System.out.println(i);
        }
    }
}
```

BreakOnDivisibleBySeven.java

```java
import java.util.Scanner;

public class BreakOnDivisibleBySeven {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            int n = sc.nextInt();
            if (n % 7 == 0) break;
            System.out.println(n);
        }
    }
}
```

ContinueAtFive.java

```java
public class ContinueAtFive {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5) continue;
            System.out.println(i);
        }
    }
}
```

ContinueAtFiveAndEight.java

```java
public class ContinueAtFiveAndEight {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5 || i == 8) continue;
            System.out.println(i);
        }
    }
}
```

Lab 9

FactorialRecursion.java

```java
import java.util.Scanner;

public class FactorialRecursion {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int fact = factorial(n);
        System.out.println(fact);
    }

    static int factorial(int n) {
        if (n == 0 || n == 1) return 1;
        return n * factorial(n - 1);
    }
}
```

FibonacciSeries.java

```java
import java.util.Scanner;

public class FibonacciSeries {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        generateFibonacci(n);
    }

    static void generateFibonacci(int n) {
        int a = 0, b = 1;
        for (int i = 1; i <= n; i++) {
            System.out.print(a + " ");
            int next = a + b;
            a = b;
            b = next;
        }
    }
}
```

MaxOfThreeMethod.java

```java
import java.util.Scanner;

public class MaxOfThreeMethod {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();
        int max = findMax(a, b, c);
        System.out.println(max);
    }

    static int findMax(int a, int b, int c) {
        if (a > b && a > c) return a;
        else if (b > c) return b;
        else return c;
    }
}
```

PrimeCheckMethod.java

```java
import java.util.Scanner;

public class PrimeCheckMethod {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int result = check(n);
        if (result == 1) System.out.println("Prime");
        else System.out.println("Not Prime");
    }

    static int check(int n) {
        if (n <= 1) return 0;
        for (int i = 2; i <= n / 2; i++) {
            if (n % i == 0) return 0;
        }
        return 1;
    }
}
```

SimpleInterestMethod.java

```java
import java.util.Scanner;

public class SimpleInterestMethod {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double p = sc.nextDouble();
        double r = sc.nextDouble();
        double t = sc.nextDouble();
        double si = calculateSI(p, r, t);
        System.out.println(si);
    }

    static double calculateSI(double p, double r, double t) {
        return (p * r * t) / 100;
    }
}
```

Lab 10

ArrayInputOutput.java

```java
import java.util.Scanner;

public class ArrayInputOutput {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

CountEvenOdd.java

```
import java.util.Scanner;

public class CountEvenOdd {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        int even = 0, odd = 0;
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
            if (arr[i] % 2 == 0) even++;
            else odd++;
        }
        System.out.println("Even: " + even);
        System.out.println("Odd: " + odd);
    }
}
```

MatrixAddition.java

```
import java.util.Scanner;

public class MatrixAddition {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[][] a = new int[3][3];
        int[][] b = new int[3][3];
        int[][] c = new int[3][3];
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                a[i][j] = sc.nextInt();
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                b[i][j] = sc.nextInt();
                c[i][j] = a[i][j] + b[i][j];
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
```

```java
                System.out.print(c[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

MatrixInputOutput.java

```java
import java.util.Scanner;

public class MatrixInputOutput {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[][] mat = new int[3][3];
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                mat[i][j] = sc.nextInt();
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

MatrixMultiplication.java

```java
import java.util.Scanner;

public class MatrixMultiplication {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[][] a = new int[3][3];
        int[][] b = new int[3][3];
        int[][] c = new int[3][3];
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                a[i][j] = sc.nextInt();
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                b[i][j] = sc.nextInt();
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                c[i][j] = 0;
                for (int k = 0; k < 3; k++) {
                    c[i][j] += a[i][k] * b[k][j];
                }
            }
        }
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.print(c[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

RemoveDuplicates.java

```java
import java.util.Scanner;

public class RemoveDuplicates {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        int[] temp = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        int j = 0;
        for (int i = 0; i < n; i++) {
            boolean found = false;
            for (int k = 0; k < j; k++) {
                if (arr[i] == temp[k]) {
                    found = true;
                    break;
                }
            }
            if (!found) {
                temp[j++] = arr[i];
            }
        }
        for (int i = 0; i < j; i++) {
            System.out.print(temp[i] + " ");
        }
    }
}
```

SecondMinMax.java

```java
import java.util.Scanner;

public class SecondMinMax {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of elements in the array:");
        int n = sc.nextInt();
        if (n < 2) {
            System.out.println("Array must contain at least two elements.");
            return;
        }
        System.out.println("Enter the elements of the array:");
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

         for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }

        if (arr.length==2) {
            System.out.println("Array has only two elements.");
            System.out.println(" Smallest: " + arr[0]);
            System.out.println(" Largest: " + arr[1]);
            return;
        }
        System.out.println("Second Smallest: " + arr[1]);
        System.out.println("Second Largest: " + arr[n - 2]);
    }
}
```

Lab 11

BinarySearch.java

```
import java.util.Scanner;
import java.util.Arrays;

public class BinarySearch {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        int key = sc.nextInt();
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        Arrays.sort(arr);
        int low = 0, high = n - 1, mid, index = -1;
        while (low <= high) {
            mid = (low + high) / 2;
            if (arr[mid] == key) {
                index = mid;
                break;
            } else if (arr[mid] < key) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }
        if (index == -1) {
            System.out.println("Not found");
        } else {
            System.out.println("Found at index " + index);
        }
    }
}
```

LinearSearch.java

```
import java.util.Scanner;

public class LinearSearch {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        int key = sc.nextInt();
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        boolean found = false;
        for (int i = 0; i < n; i++) {
            if (arr[i] == key) {
                System.out.println("Found at index " + i);
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println("Not found");
        }
    }
}
```

SortArray.java

```
import java.util.Scanner;

public class SortArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
```

```java
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }

        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Lab 12

BankAccountDetails.java

```java
import java.util.Scanner;

class Bank_Account {
    int Account_No;
    String User_Name;
    String Email;
    String Account_Type;
    double Account_Balance;

    void GetAccountDetails() {
        Scanner sc = new Scanner(System.in);
        Account_No = sc.nextInt();
        sc.nextLine();
        User_Name = sc.nextLine();
        Email = sc.nextLine();
        Account_Type = sc.nextLine();
        Account_Balance = sc.nextDouble();
    }

    void DisplayAccountDetails() {
        System.out.println(Account_No);
        System.out.println(User_Name);
        System.out.println(Email);
        System.out.println(Account_Type);
        System.out.println(Account_Balance);
    }
}
```

```java
public class BankAccountDetails {
    public static void main(String[] args) {
        Bank_Account b = new Bank_Account();
        b.GetAccountDetails();
        b.DisplayAccountDetails();
    }
}
```

CandidatDetails.java

```java
import java.util.Scanner;

class Candidate {
    int Candidate_ID;
    String Candidate_Name;
    int Candidate_Age;
    double Candidate_Weight;
    double Candidate_Height;

    void GetCandidateDetails() {
        Scanner sc = new Scanner(System.in);
        Candidate_ID = sc.nextInt();
        sc.nextLine();
        Candidate_Name = sc.nextLine();
        Candidate_Age = sc.nextInt();
        Candidate_Weight = sc.nextDouble();
        Candidate_Height = sc.nextDouble();
    }

    void DisplayCandidateDetails() {
        System.out.println(Candidate_ID);
        System.out.println(Candidate_Name);
        System.out.println(Candidate_Age);
        System.out.println(Candidate_Weight);
        System.out.println(Candidate_Height);
    }
}

public class CandidatDetails {
    public static void main(String[] args) {
        Candidate c = new Candidate();
        c.GetCandidateDetails();
        c.DisplayCandidateDetails();
    }
}
```

StudentDetails.java

```java
import java.util.Scanner;

class Student {
    String name;
    int roll;

    void getDetails() {
        Scanner sc = new Scanner(System.in);
        name = sc.nextLine();
        roll = sc.nextInt();
    }

    void showDetails() {
        System.out.println(name);
        System.out.println(roll);
    }
}

public class StudentDetails {
    public static void main(String[] args) {
        Student s = new Student();
        s.getDetails();
        s.showDetails();
    }
}
```

Lab 13

CircleTest.java

```java
import java.util.Scanner;

class Circle {
    double radius;

    void getRadius() {
        Scanner sc = new Scanner(System.in);
        radius = sc.nextDouble();
    }

    void area() {
        double a = Math.PI * radius * radius;
        System.out.println(a);
    }

    void perimeter() {
        double p = 2 * Math.PI * radius;
        System.out.println(p);
    }
}

public class CircleTest {
    public static void main(String[] args) {
        Circle c = new Circle();
        c.getRadius();
        c.area();
        c.perimeter();
    }
}
```

EmployeeTest.java

```java
import java.util.Scanner;

class Employee {
    int Employee_ID;
    String Employee_Name;
    String Designation;
    int Age;
    double Salary;

    void GetEmployeeDetails() {
        Scanner sc = new Scanner(System.in);
        Employee_ID = sc.nextInt();
        sc.nextLine();
        Employee_Name = sc.nextLine();
        Designation = sc.nextLine();
        Age = sc.nextInt();
        Salary = sc.nextDouble();
    }

    void DisplayEmployeeDetails() {
        System.out.println(Employee_ID);
        System.out.println(Employee_Name);
        System.out.println(Designation);
        System.out.println(Age);
        System.out.println(Salary);
    }
}

public class EmployeeTest {
    public static void main(String[] args) {
        Employee e = new Employee();
        e.GetEmployeeDetails();
        e.DisplayEmployeeDetails();
    }
}
```

StudentTest.java

```java
import java.util.Scanner;

class Student {
    int Enrollment_No;
    String Student_Name;
    int Semester;
    double CPI;
    double SPI;
    void GetStudentDetails() {
        Scanner sc = new Scanner(System.in);
        Enrollment_No = sc.nextInt();
        sc.nextLine();
        Student_Name = sc.nextLine();
        Semester = sc.nextInt();
        CPI = sc.nextDouble();
        SPI = sc.nextDouble();
    }

    void DisplayStudentDetails() {
        System.out.println(Enrollment_No);
        System.out.println(Student_Name);
        System.out.println(Semester);
        System.out.println(CPI);
        System.out.println(SPI);
    }
}

public class StudentTest {
    public static void main(String[] args) {
        Student s = new Student();
        s.GetStudentDetails();
        s.DisplayStudentDetails();
    }
}
```

Lab 14

EvenLengthWords.java

```
public class EvenLengthWords {
    public static void main(String[] args) {
        String sentence = "Java is an awesome language";
        String[] words = sentence.split(" ");

        for (String word : words) {
            if (word.length() % 2 == 0) {
                System.out.println(word);
            }
        }
    }
}
```

PalindromeCheck.java

```
import java.util.Scanner;

public class PalindromeCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String str = sc.nextLine();
```

```java
        String reversed = "";
        for (int i = str.length() - 1; i >= 0; i--) {
            reversed += str.charAt(i);
        }

        if (str.equals(reversed)) {
            System.out.println("Palindrome");
        } else {
            System.out.println("Not Palindrome");
        }
    }
}
```

StringBufferTest.java

```java
public class StringBufferTest {
    public static void main(String[] args) {
        StringBuffer sb = new StringBuffer("Hello");

        sb.append(" World");
        System.out.println(sb);

        sb.insert(5, " Java");
        System.out.println(sb);

        sb.delete(5, 10);
        System.out.println(sb);

        sb.reverse();
        System.out.println(sb);

        System.out.println(sb.charAt(3));
        System.out.println(sb.capacity());
        System.out.println(sb.toString());

        sb.replace(0, 5, "Test");
        System.out.println(sb);
    }
}
```

StringFunctionsTest.java

```java
public class StringFunctionsTest {
    public static void main(String[] args) {
        String str = " Hello Java ";
        String str2 = "World";

        System.out.println(str.length());
        System.out.println(str.charAt(2));
        System.out.println(str.concat(str2));
        System.out.println(str.indexOf('J'));
        System.out.println(str.equals(str2));
        System.out.println(String.valueOf(123));
        System.out.println(str.toString());
        System.out.println(str.trim());
        System.out.println(str.substring(1, 6));
    }
}
```

Lab 15

ArmstrongNumber.java

```java
import java.util.Scanner;

public class ArmstrongNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int number = sc.nextInt();
        int original = number;
        int digits = String.valueOf(number).length();
        int sum = 0;

        while (number > 0) {
            int digit = number % 10;
            sum += Math.pow(digit, digits);
            number /= 10;
        }

        if (sum == original) {
            System.out.println("Armstrong");
        } else {
            System.out.println("Not Armstrong");
        }
    }
}
```

LargestOfThree.java

```java
import java.util.Scanner;

public class LargestOfThree {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();

        int maxAB = Math.max(a, b);
        int maxABC = Math.max(maxAB, c);

        System.out.println(maxABC);
    }
}
```

MathFunctionsTest.java

```java
public class MathFunctionsTest {
    public static void main(String[] args) {
        System.out.println(Math.min(5, 10));
        System.out.println(Math.max(5, 10));
        System.out.println(Math.random());
        System.out.println(Math.pow(2, 3));
        System.out.println(Math.sqrt(16));
        System.out.println(Math.round(4.6));
        System.out.println(Math.ceil(4.2));
        System.out.println(Math.floor(4.8));
        System.out.println(Math.abs(-9));
    }
}
```

Lab 16

AnonymousInnerClass.java

```java
class Animal {
   void sound(){System.out.println("Hi I am Normal");};
}

public class AnonymousInnerClass {
    public static void main(String[] args) {
        Animal obj = new Animal() {
            void sound() {
                System.out.println("Hi I am Anonymous");
            }
        };
        obj.sound();
    }
}
```

OuterClass.java

```java
public class OuterClass {
    int x = 5;

    static class NestedClass {
        void display() {
            System.out.println("This is a static nested class.");
        }
    }

    public static void main(String[] args) {
        NestedClass obj = new NestedClass();
        obj.display();
    }
}
```

WrapperExample.java

```java
public class WrapperExample {
    public static void main(String[] args) {
        int a = 10;
        Integer obj = a;
        int b = obj;
        System.out.println(obj);
        System.out.println(b);
    }
}
```

Lab 17

AreaCalculator.java

```java
//  Area of Circle, Triangle, and Square using Method Overloading (A)
public class AreaCalculator {
    void area(double r) {
        System.out.println(3.14 * r * r);
    }

    void area(int b, int h) {
        System.out.println(0.5 * b * h);
    }

    void area(int a) {
        System.out.println(a * a);
    }

    public static void main(String[] args) {
        AreaCalculator obj = new AreaCalculator();
        obj.area(5.0);
        obj.area(4, 6);
        obj.area(4);
    }
}
```

Book.java

```java
// Copy Constructor (B)
public class Book {
    String title;

    Book(String t) {
        title = t;
    }

    Book(Book b) {
        title = b.title;
    }

    void display() {
        System.out.println(title);
    }

    public static void main(String[] args) {
        Book b1 = new Book("Java");
        Book b2 = new Book(b1);
        b1.display();
        b2.display();
    }
}
```

Employee.java

```java
//  Constructor Overloading (C)
public class Employee {
    String name;
    int age;

    Employee() {
        name = "None";
        age = 0;
    }

    Employee(String n) {
        name = n;
        age = 0;
    }

    Employee(String n, int a) {
        name = n;
        age = a;
    }

    void display() {
        System.out.println(name + " " + age);
    }

    public static void main(String[] args) {
        Employee e1 = new Employee();
        Employee e2 = new Employee("Alice");
        Employee e3 = new Employee("Bob", 30);
        e1.display();
        e2.display();
        e3.display();
    }
}
```

Student.java

```java
// Default and Parameterized Constructor (A)
public class Student {
    String name;
    int age;

    Student() {
        name = "Default";
        age = 0;
    }

    Student(String n, int a) {
        name = n;
        age = a;
    }

    void display() {
        System.out.println(name + " " + age);
    }

    public static void main(String[] args) {
        Student s1 = new Student();
        Student s2 = new Student("John", 22);
        s1.display();
        s2.display();
    }
}
```

Lab 18

StaticBlockExample.java

```
public class StaticBlockExample {
    static {
        System.out.println("Static block executed");
    }

    static void show() {
        System.out.println("Static method called");
    }

    public static void main(String[] args) {
        show();
    }
}
```

StaticExample.java

```
public class StaticExample {
    static int count = 0;

    StaticExample() {
        count++;
    }

    void display() {
        System.out.println(count);
    }

    public static void main(String[] args) {
        StaticExample a = new StaticExample();
        StaticExample b = new StaticExample();
        StaticExample c = new StaticExample();
        c.display();
    }
}
```

ThisExample.java

```
public class ThisExample {
    int a;

    ThisExample(int a) {
        this.a = a;
    }

    void display() {
        System.out.println(a);
    }

    public static void main(String[] args) {
        ThisExample obj = new ThisExample(10);
        obj.display();
    }
}
```

ThisStaticCheck.java

```
public class ThisStaticCheck {
    static int x = 100;

    void display() {
        System.out.println(this.x);
    }

    public static void main(String[] args) {
        ThisStaticCheck obj = new ThisStaticCheck();
        ThisStaticCheck obj2 = new ThisStaticCheck();
        obj.display();
        obj2.display();
    }
}
```

Lab 19

InheritanceDemo.java

```java
// Single Inheritance
class Animal {
    void eat() {
        System.out.println("This animal eats food.");
    }
}

class Dog extends Animal {
    void bark() {
        System.out.println("Dog barks.");
    }
}

// Multilevel Inheritance
class Vehicle {
    void start() {
        System.out.println("Vehicle started.");
    }
}

class Car extends Vehicle {
    void drive() {
        System.out.println("Car is driving.");
    }
}

class SportsCar extends Car {
    void turbo() {
        System.out.println("Sports car turbo mode ON.");
    }
}

// Hierarchical Inheritance
class Person {
    void walk() {
        System.out.println("Person is walking.");
    }
}

class Teacher extends Person {
    void teach() {
        System.out.println("Teacher is teaching.");
    }
}

class Student extends Person {
    void study() {
        System.out.println("Student is studying.");
    }
}

public class InheritanceDemo {
    public static void main(String[] args) {
        // Single Inheritance
        Dog dog = new Dog();
        dog.eat();
        dog.bark();

        System.out.println();

        // Multilevel Inheritance
        SportsCar sportsCar = new SportsCar();
        sportsCar.start();
        sportsCar.drive();
        sportsCar.turbo();

        System.out.println();

        // Hierarchical Inheritance
        Teacher teacher = new Teacher();
        teacher.walk();
        teacher.teach();

        Student student = new Student();
        student.walk();
        student.study();
    }
}
```

InterestDemo.java

```
import java.util.Scanner;

class AccountDetails {
    String name;
    double principal;
    double rate;
    int time;

    void getDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account holder name: ");
        name = sc.nextLine();
        System.out.print("Enter principal amount: ");
        principal = sc.nextDouble();
        System.out.print("Enter rate of interest (%): ");
        rate = sc.nextDouble();
        System.out.print("Enter time (years): ");
        time = sc.nextInt();
    }
}

class Interest extends AccountDetails {
    double calculateInterest() {
```

```java
        return (principal * rate * time) / 100;
    }

    void displayInterest() {
        double interest = calculateInterest();
        System.out.println("Account Holder: " + name);
        System.out.println("Total Interest: " + interest);
    }
}

public class InterestDemo {
    public static void main(String[] args) {
        Interest interest = new Interest();
        interest.getDetails();
        interest.displayInterest();
    }
}
```

ShapeAreaDemo.java

```java
class Shape {
    void area() {
        System.out.println("Area of shape");
    }
}

class Circle extends Shape {
    double radius;

    Circle(double radius) {
        this.radius = radius;
    }

    void area() {
        double area = Math.PI * radius * radius;
        System.out.println("Area of Circle: " + area);
    }
}

class Triangle extends Shape {
    double base, height;

    Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }

    void area() {
        double area = 0.5 * base * height;
        System.out.println("Area of Triangle: " + area);
    }
}
```

```java
class Square extends Shape {
    double side;

    Square(double side) {
        this.side = side;
    }

    void area() {
        double area = side * side;
        System.out.println("Area of Square: " + area);
    }
}

public class ShapeAreaDemo {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        circle.area();

        Triangle triangle = new Triangle(4, 6);
        triangle.area();

        Square square = new Square(3);
        square.area();
    }
}
```

Lab 20

FinalKeywordDemo.java

```java
final class FinalClass {
    final int finalVar = 50;

    final void finalMethod() {
        System.out.println("This is a final method.");
    }
}


class TestFinal {
    final int CONSTANT = 10;

}

public class FinalKeywordDemo {
    public static void main(String[] args) {
        FinalClass obj = new FinalClass();
        obj.finalMethod();

        TestFinal test = new TestFinal();
        System.out.println("Final variable: " + test.CONSTANT);
    }
}
```

MemberDemo.java

```
class Member {
    String name;
    int age;
    String phoneNumber;
    String address;
    double salary;

    void printSalary() {
        System.out.println("Salary: " + salary);
    }
}

class Employee extends Member {
    String specialization;

    void printEmployeeDetails() {
        System.out.println("Employee Details:");
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " + phoneNumber);
        System.out.println("Address: " + address);
        printSalary();
        System.out.println("Specialization: " + specialization);
        System.out.println();
    }
}

class Manager extends Member {
    String department;

    void printManagerDetails() {
        System.out.println("Manager Details:");
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " + phoneNumber);
        System.out.println("Address: " + address);
        printSalary();
        System.out.println("Department: " + department);
        System.out.println();
    }
}

public class MemberDemo {
    public static void main(String[] args) {
        Employee emp = new Employee();
        emp.name = "Alice";
        emp.age = 28;
        emp.phoneNumber = "1234567890";
        emp.address = "123 Main St";
        emp.salary = 50000;
        emp.specialization = "Software Development";

        Manager mgr = new Manager();
        mgr.name = "Bob";
        mgr.age = 35;
        mgr.phoneNumber = "0987654321";
        mgr.address = "456 Park Ave";
        mgr.salary = 80000;
        mgr.department = "IT";

        emp.printEmployeeDetails();
        mgr.printManagerDetails();
    }
}
```

MethodOverridingDemo.java

```
class Parent {
    void show() {
        System.out.println("Parent's show method");
    }
}

class Child extends Parent {
    @Override
    void show() {
        System.out.println("Child's overridden show method");
    }
}

public class MethodOverridingDemo {
    public static void main(String[] args) {
        Parent p = new Parent();
        p.show();

        Child c = new Child();
        c.show();

        Parent p2 = new Child();
        p2.show();  // runtime polymorphism
    }
}
```

SuperKeywordDemo.java

```
class Parent {
    int num = 100;

    Parent() {
        System.out.println("Parent constructor called");
    }

    void display() {
        System.out.println("Parent display method");
    }
}

class Child extends Parent {
    int num = 200;

    Child() {
        super(); // calling parent constructor
        System.out.println("Child constructor called");
    }

    void display() {
        super.display(); // calling parent method
        System.out.println("Child display method");
    }

    void printNum() {
        System.out.println("Child num: " + num);
        System.out.println("Parent num: " + super.num); // accessing parent variable
    }
}

public class SuperKeywordDemo {
    public static void main(String[] args) {
        Child c = new Child();
        c.display();
        c.printNum();
    }
}
```

Lab 21

AbstractShapeDemo.java

```
abstract class Shape {
    abstract double area();  // abstract method
}

class Rectangle extends Shape {
    double length, width;

    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    double area() {
```

```java
        return length * width;
    }
}

class Circle extends Shape {
    double radius;

    Circle(double radius) {
        this.radius = radius;
    }

    @Override
    double area() {
        return Math.PI * radius * radius;
    }
}

public class AbstractShapeDemo {
    public static void main(String[] args) {
        Rectangle rect = new Rectangle(5, 3);
        Circle circle = new Circle(4);

        System.out.println("Area of Rectangle: " + rect.area());
        System.out.println("Area of Circle: " + circle.area());
    }
}
```

VegetableDemo.java

```java
abstract class Vegetable {
    String color;

    Vegetable(String color) {
        this.color = color;
    }

    @Override
    public String toString() {
        return this.getClass().getSimpleName() + " color: " + color;
    }
}

class Potato extends Vegetable {
    Potato(String color) {
        super(color);
    }
}

class Brinjal extends Vegetable {
    Brinjal(String color) {
        super(color);
    }
}

class Tomato extends Vegetable {
    Tomato(String color) {
        super(color);
    }
}

public class VegetableDemo {
    public static void main(String[] args) {
        Potato potato = new Potato("Brown");
        Brinjal brinjal = new Brinjal("Purple");
        Tomato tomato = new Tomato("Red");

        System.out.println(potato);
        System.out.println(brinjal);
        System.out.println(tomato);
    }
}
```

Lab 22

MultipleInheritanceDemo.java

```java
interface Printable {
    void print();
}

interface Showable {
    void show();
}

class Demo implements Printable, Showable {
    @Override
    public void print() {
        System.out.println("Printing...");
    }

    @Override
    public void show() {
        System.out.println("Showing...");
    }
}

public class MultipleInheritanceDemo {
    public static void main(String[] args) {
        Demo obj = new Demo();
        obj.print();
        obj.show();
    }
}
```

PublicationDemo.java

```java
class Book {
    private String author_name;

    Book(String author_name) {
        this.author_name = author_name;
    }

    public void display() {
        System.out.println("Author: " + author_name);
    }
}

class BookPublication extends Book {
    private String title;

    BookPublication(String author_name, String title) {
        super(author_name);
        this.title = title;
    }

    @Override
    public void display() {
        super.display();
        System.out.println("Book Publication Title: " + title);
    }
}

class PaperPublication extends Book {
    private String title;

    PaperPublication(String author_name, String title) {
        super(author_name);
        this.title = title;
    }

    @Override
    public void display() {
        super.display();
        System.out.println("Paper Publication Title: " + title);
    }
}

public class PublicationDemo {
    public static void main(String[] args) {
        Book b1 = new BookPublication("J.K. Rowling", "Harry Potter");
        Book b2 = new PaperPublication("Albert Einstein", "Theory of Relativity");

        b1.display();
        System.out.println();
        b2.display();
    }
}
```

TransportDemo.java

```
interface Transport {
    void deliver();
}

abstract class Animal {
    abstract void sound();
}

class Tiger extends Animal {
    @Override
    void sound() {
        System.out.println("Tiger roars");
    }
}

class Camel extends Animal implements Transport {
    @Override
    void sound() {
        System.out.println("Camel grunts");
    }

    @Override
    public void deliver() {
        System.out.println("Camel is delivering goods");
    }
}

class Deer extends Animal {
    @Override
    void sound() {
        System.out.println("Deer makes a sound");
    }
}

class Donkey extends Animal implements Transport {
    @Override
    void sound() {
        System.out.println("Donkey brays");
    }

    @Override
    public void deliver() {
        System.out.println("Donkey is delivering goods");
    }
}

public class TransportDemo {
```

```java
    public static void main(String[] args) {
        Animal[] animals = {new Tiger(), new Camel(), new Deer(), new Donkey()};

        for (Animal a : animals) {
            a.sound();
            if (a instanceof Transport) {
                ((Transport) a).deliver();
            }
            System.out.println();
        }
    }
}
```

Lab_23

Q1

AccessDemo.java

```java
package Lab_23.Q1;
import Lab_23.Q1.pkg2.*;

public class AccessDemo {
    public static void main(String[] args) {
        Derived d = new Derived();
        d.show();
    }
}
```

pkg1

Base.java

```java
package Lab_23.Q1.pkg1;

public class Base {
    private int privateVar = 1;
    int defaultVar = 2;
    protected int protectedVar = 3;
    public int publicVar = 4;

    public void display() {
        System.out.println("privateVar: " + privateVar);
        System.out.println("defaultVar: " + defaultVar);
        System.out.println("protectedVar: " + protectedVar);
        System.out.println("publicVar: " + publicVar);
    }
}
```

pkg2

Derived.java

```java
package Lab_23.Q1.pkg2;
import Lab_23.Q1.pkg1.Base;

public class Derived extends Base {
    public void show() {
        // System.out.println(privateVar); // Not accessible
        // System.out.println(defaultVar); // Not accessible
        System.out.println(protectedVar);
        System.out.println(publicVar);
    }
}
```

Q2

Exam

Result.java

```java
package Lab_23.Q2.Exam;

import Lab_23.Q2.Student.Student;

public class Result extends Student {
    int marks1, marks2;

    public Result(String name, int roll, int marks1, int marks2) {
        super(name, roll);
        this.marks1 = marks1;
        this.marks2 = marks2;
    }

    public void display() {
        System.out.println("Name: " + name);
        System.out.println("Roll No: " + roll);
        System.out.println("Total Marks: " + (marks1 + marks2));
    }
}
```

MarksheetDemo.java

```java
package Lab_23.Q2;
import Lab_23.Q2.Exam.Result;

public class MarksheetDemo {
    public static void main(String[] args) {
        Result r = new Result("Ankit", 101, 85, 90);
        r.display();
    }
}
```

Student

Student.java

```
package Lab_23.Q2.Student;

public class Student {
    public String name;
    public int roll;

    public Student(String name, int roll) {
        this.name = name;
        this.roll = roll;
    }
}
```

Q3

Calculater

MathOperation.java

```
package Lab_23.Q3.Calculater;

public class MathOperation {
    public int add(int a, int b) { return a + b; }
    public int sub(int a, int b) { return a - b; }
    public int mul(int a, int b) { return a * b; }
    public int div(int a, int b) { return b != 0 ? a / b : 0; }
}
```

DemoOperation.java

```
package Lab_23.Q3;

import Lab_23.Q3.Calculater.MathOperation;

public class DemoOperation {
    public static void main(String[] args) {
        MathOperation mo = new MathOperation();
        System.out.println("Add: " + mo.add(10, 5));
        System.out.println("Sub: " + mo.sub(10, 5));
        System.out.println("Mul: " + mo.mul(10, 5));
        System.out.println("Div: " + mo.div(10, 5));
    }
}
```

Q4

Pkg1

A.java

```
package Lab_23.Q4.Pkg1 ;

```

```java
public class A {
    protected int protectedVar = 10;
    private int privateVar = 20;
    public int publicVar = 30;
}
```

Pkg2

B.java

```java
package Lab_23.Q4.Pkg2;

import Lab_23.Q4.Pkg1.A;

public class B extends A {
    public void display() {
        System.out.println(protectedVar);
        System.out.println(publicVar);
        // System.out.println(privateVar);
    }
}
```

Pkg3

C.java

```java
package Lab_23.Q4.Pkg3;

import Lab_23.Q4.Pkg1.A;

public class C {
    public void display() {
        A a = new A();
        System.out.println(a.publicVar);
        // System.out.println(a.protectedVar);
        // System.out.println(a.privateVar);
    }
}
```

Pkg4

ProtectedDemo.java

```java
package Lab_23.Q4.Pkg4;

import Lab_23.Q4.Pkg2.B;
import Lab_23.Q4.Pkg3.C;

public class ProtectedDemo {
    public static void main(String[] args) {
        B b = new B();
        C c = new C();
        b.display();
        c.display();
    }
}
```

Lab 24

BankAccountDemo.java

```
class InsufficientBalanceException extends Exception {
    InsufficientBalanceException(String msg) {
        super(msg);
    }
}

class Account {
    double balance;

    void deposit(double amount) {
        balance += amount;
    }

    void withdraw(double amount) throws InsufficientBalanceException {
        if (amount > balance) {
            throw new InsufficientBalanceException("Insufficient Balance.");
        }
        balance -= amount;
    }
}

public class BankAccountDemo {
    public static void main(String[] args) {
        Account acc = new Account();
        acc.deposit(1000);
        try {
            acc.withdraw(1500);
        } catch (InsufficientBalanceException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

CustomExceptionDemo.java

```
class MyException extends Exception {
    MyException(String message) {
        super(message);
    }
}

public class CustomExceptionDemo {
    public static void main(String[] args) {
        try {
```

```java
            throw new MyException("This is a custom exception.");
        } catch (MyException e) {
            System.out.println("Caught: " + e.getMessage());
        }
    }
}
```

CustomValidation.java

```java
public class CustomValidation {
    public static void main(String[] args) {
        int sum = 0;

        for (String arg : args) {
            try {
                int num = Integer.parseInt(arg);

                if (num < 0) throw new Exception("Negative number");
                if (num % 10 == 0) throw new Exception("Divisible by 10");
                if (num > 1000 && num < 2000) throw new Exception("Between 1000 and 
2000");
                if (num > 7000) throw new Exception("Greater than 7000");

                sum += num;
            } catch (Exception e) {
                System.out.println("Skipped: " + arg + " (" + e.getMessage() + ")");
            }
        }

        System.out.println("Sum = " + sum);
    }
}
```

MultipleExceptionDemo.java

```java
public class MultipleExceptionDemo {
    public static void main(String[] args) {
        try {
            int[] arr = new int[3];
            arr[5] = 10;
            int result = 10 / 0;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array exception: " + e.getMessage());
        } catch (ArithmeticException e) {
            System.out.println("Arithmetic exception: " + e.getMessage());
        }
    }
}
```

ThrowThrowsDemo.java

```java
public class ThrowThrowsDemo {
    static void checkAge(int age) throws Exception {
        if (age < 18) {
            throw new Exception("Age must be 18 or above.");
        }
        System.out.println("Eligible.");
    }

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

TryCatchFinallyDemo.java

```java
public class TryCatchFinallyDemo {
    public static void main(String[] args) {
        try {
            int result = 10 / 0;
        } catch (ArithmeticException e) {
            System.out.println("Exception: " + e.getMessage());
        } finally {
            System.out.println("Finally block executed.");
        }
    }
}
```

Lab 25

CopyFileDemo.java

```java
import java.io.*;

public class CopyFileDemo {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("source.txt");
            FileWriter fw = new FileWriter("copy.txt");
            int ch;
            while ((ch = fr.read()) != -1) {
                fw.write(ch);
                System.out.print((char) ch);
            }
            fr.close();
            fw.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

CopyToMultiple.java

```
import java.io.*;

public class CopyToMultiple {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("original.txt");
            FileWriter fw1 = new FileWriter("copy1.txt");
            FileWriter fw2 = new FileWriter("copy2.txt");

            int ch;
            while ((ch = fr.read()) != -1) {
                fw1.write(ch);
                fw2.write(ch);
            }

            fr.close();
            fw1.close();
            fw2.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

CreateFileDemo.java

```
import java.io.*;

public class CreateFileDemo {
    public static void main(String[] args) {
        File file = new File("created_file.txt");
        try {
            if (file.createNewFile()) {
                System.out.println("File created: " + file.getAbsolutePath());
            } else {
                System.out.println("File already exists.");
            }
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

MergeFiles.java

```
import java.io.*;

public class MergeFiles {
    public static void main(String[] args) {
        try {
            FileReader fr1 = new FileReader("file1.txt");
            FileReader fr2 = new FileReader("file2.txt");
            FileWriter fw = new FileWriter("merged.txt");

            int ch;
            while ((ch = fr1.read()) != -1) fw.write(ch);
            while ((ch = fr2.read()) != -1) fw.write(ch);

            fr1.close();
            fr2.close();
            fw.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Lab 26

BufferedReaderWriterDemo.java

```
import java.io.*;

public class BufferedReaderWriterDemo {
    public static void main(String[] args) {
        String text = "BufferedReader and BufferedWriter demo.";

        try {
            BufferedWriter writer = new BufferedWriter(new FileWriter("demo2.txt"));
            writer.write(text);
            writer.newLine();
            writer.write("Second line of text.");
            writer.close();
            System.out.println("Data written using BufferedWriter.");

            BufferedReader reader = new BufferedReader(new FileReader("demo2.txt"));
            String line;
            System.out.println("Data read using BufferedReader:");
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

BufferedStreamDemo.java

```
import java.io.*;
```

```java
public class BufferedStreamDemo {
    public static void main(String[] args) {
        String data = "BufferedInputStream and BufferedOutputStream demo.";

        try {
            // Writing using BufferedOutputStream
            FileOutputStream fos = new FileOutputStream("demo3.txt");
            BufferedOutputStream bos = new BufferedOutputStream(fos);
            bos.write(data.getBytes());
            bos.close();
            System.out.println("Data written using BufferedOutputStream.");

            // Reading using BufferedInputStream
            FileInputStream fis = new FileInputStream("demo3.txt");
            BufferedInputStream bis = new BufferedInputStream(fis);
            int ch;
            System.out.print("Data read using BufferedInputStream: ");
            while ((ch = bis.read()) != -1) {
                System.out.print((char) ch);
            }
            bis.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

FileIOStreamDemo.java

```java
import java.io.*;

public class FileIOStreamDemo {
    public static void main(String[] args) {
        String data = "This is a demo of FileOutputStream and FileInputStream.";

        try {
            FileOutputStream fos = new FileOutputStream("demo1.txt");
            fos.write(data.getBytes());
            fos.close();
            System.out.println("Data written to file using FileOutputStream.");

            FileInputStream fis = new FileInputStream("demo1.txt");
            int ch;
            System.out.print("Data read from file: ");
            while ((ch = fis.read()) != -1) {
                System.out.print((char) ch);
            }
            fis.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

.git

COMMIT_EDITMSG

```
Lab 23-24-25-26 Added
```

HEAD

```
ref: refs/heads/main
```

config

```
[core]
    repositoryformatversion = 0
    filemode = false
    bare = false
    logallrefupdates = true
    symlinks = false
    ignorecase = true
[remote "origin"]
    url = https://github.com/Shreyansh284/TA-OOPJ-Dip-3-Lab
    fetch = +refs/heads/*:refs/remotes/origin/*
[branch "main"]
    remote = origin
    merge = refs/heads/main
```

description

```
Unnamed repository; edit this file 'description' to name the repository.
```

hooks

applypatch-msg.sample

```
#!/bin/sh
#
# An example hook script to check the commit log message taken by
# applypatch from an e-mail message.
#
# The hook should exit with non-zero status after issuing an
# appropriate message if it wants to stop the commit.  The hook is
# allowed to edit the commit message file.
#
# To enable this hook, rename this file to "applypatch-msg".

. git-sh-setup
commitmsg="$(git rev-parse --git-path hooks/commit-msg)"
test -x "$commitmsg" && exec "$commitmsg" ${1+"$@"}
:
```

commit-msg.sample

```
#!/bin/sh
#
# An example hook script to check the commit log message.
# Called by "git commit" with one argument, the name of the file
# that has the commit message.  The hook should exit with non-zero
# status after issuing an appropriate message if it wants to stop the
# commit.  The hook is allowed to edit the commit message file.
#
# To enable this hook, rename this file to "commit-msg".

# Uncomment the below to add a Signed-off-by line to the message.
# Doing this in a hook is a bad idea in general, but the prepare-commit-msg
# hook is more suited to it.
#
# SOB=$(git var GIT_AUTHOR_IDENT | sed -n 's/^\(.*>\).*$/Signed-off-by: \1/p')
# grep -qs "^$SOB" "$1" || echo "$SOB" >> "$1"

# This example catches duplicate Signed-off-by lines.

test "" = "$(grep '^Signed-off-by: ' "$1" |
     sort | uniq -c | sed -e '/^[     ]*1[     ]/d')" || {
    echo >&2 Duplicate Signed-off-by lines.
    exit 1
}
```

fsmonitor-watchman.sample

```
#!/usr/bin/perl

use strict;
use warnings;
use IPC::Open2;

# An example hook script to integrate Watchman
# (https://facebook.github.io/watchman/) with git to speed up detecting
# new and modified files.
#
# The hook is passed a version (currently 2) and last update token
# formatted as a string and outputs to stdout a new update token and
# all files that have been modified since the update token. Paths must
# be relative to the root of the working tree and separated by a single NUL.
#
# To enable this hook, rename this file to "query-watchman" and set
# 'git config core.fsmonitor .git/hooks/query-watchman'
#
my ($version, $last_update_token) = @ARGV;

# Uncomment for debugging
# print STDERR "$0 $version $last_update_token\n";
# Check the hook interface version
if ($version ne 2) {
    die "Unsupported query-fsmonitor hook version '$version'.\n" .
        "Falling back to scanning...\n";
}

my $git_work_tree = get_working_dir();

my $retry = 1;

my $json_pkg;
eval {
    require JSON::XS;
    $json_pkg = "JSON::XS";
    1;
} or do {
    require JSON::PP;
    $json_pkg = "JSON::PP";
};

launch_watchman();

sub launch_watchman {
    my $o = watchman_query();
    if (is_work_tree_watched($o)) {
        output_result($o->{clock}, @{$o->{files}});
    }
}

sub output_result {
    my ($clockid, @files) = @_;

    # Uncomment for debugging watchman output
    # open (my $fh, ">", ".git/watchman-output.out");
    # binmode $fh, ":utf8";
    # print $fh "$clockid\n@files\n";
    # close $fh;

    binmode STDOUT, ":utf8";
    print $clockid;
    print "\0";
    local $, = "\0";
    print @files;
}

sub watchman_clock {
    my $response = qx/watchman clock "$git_work_tree"/;
    die "Failed to get clock id on '$git_work_tree'.\n" .
        "Falling back to scanning...\n" if $? != 0;

    return $json_pkg->new->utf8->decode($response);
}
```

```
sub watchman_query {
    my $pid = open2(\*CHLD_OUT, \*CHLD_IN, 'watchman -j --no-pretty')
    or die "open2() failed: $!\n" .
    "Falling back to scanning...\n";

    # In the query expression below we're asking for names of files that
    # changed since $last_update_token but not from the .git folder.
    #
    # To accomplish this, we're using the "since" generator to use the
    # recency index to select candidate nodes and "fields" to limit the
    # output to file names only. Then we're using the "expression" term to
    # further constrain the results.
    my $last_update_line = "";
    if (substr($last_update_token, 0, 1) eq "c") {
        $last_update_token = "\"$last_update_token\"";
        $last_update_line = qq[\n"since": $last_update_token,];
    }
    my $query = <<"    END";
        ["query", "$git_work_tree", {$last_update_line
            "fields": ["name"],
            "expression": ["not", ["dirname", ".git"]]
        }]
    END

    # Uncomment for debugging the watchman query
    # open (my $fh, ">", ".git/watchman-query.json");
    # print $fh $query;
    # close $fh;

    print CHLD_IN $query;
    close CHLD_IN;
    my $response = do {local $/; <CHLD_OUT>};

    # Uncomment for debugging the watch response
    # open ($fh, ">", ".git/watchman-response.json");
    # print $fh $response;
    # close $fh;

    die "Watchman: command returned no output.\n" .
    "Falling back to scanning...\n" if $response eq "";
    die "Watchman: command returned invalid output: $response\n" .
    "Falling back to scanning...\n" unless $response =~ /^\{/;

    return $json_pkg->new->utf8->decode($response);
}

sub is_work_tree_watched {
    my ($output) = @_;
    my $error = $output->{error};
    if ($retry > 0 and $error and $error =~ m/unable to resolve root .* directory (.*)
is not watched/) {
        $retry--;
        my $response = qx/watchman watch "$git_work_tree"/;
        die "Failed to make watchman watch '$git_work_tree'.\n" .
            "Falling back to scanning...\n" if $? != 0;
        $output = $json_pkg->new->utf8->decode($response);
        $error = $output->{error};
        die "Watchman: $error.\n" .
        "Falling back to scanning...\n" if $error;

        # Uncomment for debugging watchman output
        # open (my $fh, ">", ".git/watchman-output.out");
        # close $fh;

        # Watchman will always return all files on the first query so
        # return the fast "everything is dirty" flag to git and do the
        # Watchman query just to get it over with now so we won't pay
        # the cost in git to look up each individual file.
        my $o = watchman_clock();
        $error = $output->{error};

        die "Watchman: $error.\n" .
        "Falling back to scanning...\n" if $error;

        output_result($o->{clock}, ("/"));
        $last_update_token = $o->{clock};

        eval { launch_watchman() };
        return 0;
    }

    die "Watchman: $error.\n" .
    "Falling back to scanning...\n" if $error;

    return 1;
}

sub get_working_dir {
    my $working_dir;
    if ($^O =~ 'msys' || $^O =~ 'cygwin') {
        $working_dir = Win32::GetCwd();
        $working_dir =~ tr/\\/\//;
    } else {
        require Cwd;
        $working_dir = Cwd::cwd();
    }

    return $working_dir;
}
```

post-update.sample

```
#!/bin/sh
#
# An example hook script to prepare a packed repository for use over
# dumb transports.
#
# To enable this hook, rename this file to "post-update".

exec git update-server-info
```

pre-applypatch.sample

```
#!/bin/sh
#
# An example hook script to verify what is about to be committed
# by applypatch from an e-mail message.
#
# The hook should exit with non-zero status after issuing an
# appropriate message if it wants to stop the commit.
#
# To enable this hook, rename this file to "pre-applypatch".

. git-sh-setup
precommit="$(git rev-parse --git-path hooks/pre-commit)"
test -x "$precommit" && exec "$precommit" ${1+"$@"}
:
```

pre-commit.sample

```
#!/bin/sh
#
# An example hook script to verify what is about to be committed.
# Called by "git commit" with no arguments.  The hook should
# exit with non-zero status after issuing an appropriate message if
# it wants to stop the commit.
#
# To enable this hook, rename this file to "pre-commit".

if git rev-parse --verify HEAD >/dev/null 2>&1
then
    against=HEAD
else
    # Initial commit: diff against an empty tree object
    against=$(git hash-object -t tree /dev/null)
fi

# If you want to allow non-ASCII filenames set this variable to true.
allownonascii=$(git config --type=bool hooks.allownonascii)

# Redirect output to stderr.
exec 1>&2

# Cross platform projects tend to avoid non-ASCII filenames; prevent
# them from being added to the repository. We exploit the fact that the
# printable range starts at the space character and ends with tilde.
if [ "$allownonascii" != "true" ] &&
    # Note that the use of brackets around a tr range is ok here, (it's
    # even required, for portability to Solaris 10's /usr/bin/tr), since
    # the square bracket bytes happen to fall in the designated range.
    test $(git diff-index --cached --name-only --diff-filter=A -z $against |
      LC_ALL=C tr -d '[ -~]\0' | wc -c) != 0
then
    cat <<\EOF
Error: Attempt to add a non-ASCII file name.

This can cause problems if you want to work with people on other platforms.

To be portable it is advisable to rename the file.

If you know what you are doing you can disable this check using:

  git config hooks.allownonascii true
EOF
    exit 1
fi

# If there are whitespace errors, print the offending file names and fail.
exec git diff-index --check --cached $against --
```

pre-merge-commit.sample

```
#!/bin/sh
#
# An example hook script to verify what is about to be committed.
# Called by "git merge" with no arguments.  The hook should
# exit with non-zero status after issuing an appropriate message to
# stderr if it wants to stop the merge commit.
#
# To enable this hook, rename this file to "pre-merge-commit".

. git-sh-setup
test -x "$GIT_DIR/hooks/pre-commit" &&
        exec "$GIT_DIR/hooks/pre-commit"
:
```

pre-push.sample

```
#!/bin/sh

# An example hook script to verify what is about to be pushed.  Called by "git
# push" after it has checked the remote status, but before anything has been
# pushed.  If this script exits with a non-zero status nothing will be pushed.
#
# This hook is called with the following parameters:
#
# $1 -- Name of the remote to which the push is being done
# $2 -- URL to which the push is being done
#
# If pushing without using a named remote those arguments will be equal.
#
# Information about the commits which are being pushed is supplied as lines to
# the standard input in the form:
#
#   <local ref> <local oid> <remote ref> <remote oid>
#
# This sample shows how to prevent push of commits where the log message starts
# with "WIP" (work in progress).

remote="$1"
url="$2"

zero=$(git hash-object --stdin </dev/null | tr '[0-9a-f]' '0')

while read local_ref local_oid remote_ref remote_oid
do
    if test "$local_oid" = "$zero"
    then
        # Handle delete
        :
    else
        if test "$remote_oid" = "$zero"
        then
            # New branch, examine all commits
            range="$local_oid"
        else
            # Update to existing branch, examine new commits
            range="$remote_oid..$local_oid"
        fi

        # Check for WIP commit
        commit=$(git rev-list -n 1 --grep '^WIP' "$range")
        if test -n "$commit"
        then
            echo >&2 "Found WIP commit in $local_ref, not pushing"
            exit 1
        fi
    fi
done

exit 0
```

pre-rebase.sample

```
#!/bin/sh
#
# Copyright (c) 2006, 2008 Junio C Hamano
#
# The "pre-rebase" hook is run just before "git rebase" starts doing
# its job, and can prevent the command from running by exiting with
# non-zero status.
#
# The hook is called with the following parameters:
#
# $1 -- the upstream the series was forked from.
# $2 -- the branch being rebased (or empty when rebasing the current branch).
#
# This sample shows how to prevent topic branches that are already
# merged to 'next' branch from getting rebased, because allowing it
# would result in rebasing already published history.

publish=next
basebranch="$1"
if test "$#" = 2
then
    topic="refs/heads/$2"
else
    topic=`git symbolic-ref HEAD` ||
    exit 0 ;# we do not interrupt rebasing detached HEAD
fi

case "$topic" in
refs/heads/??/*)
    ;;
*)
    exit 0 ;# we do not interrupt others.
    ;;
esac

# Now we are dealing with a topic branch being rebased
# on top of master.  Is it OK to rebase it?

# Does the topic really exist?
git show-ref -q "$topic" || {
    echo >&2 "No such branch $topic"
    exit 1
}

# Is topic fully merged to master?
not_in_master=`git rev-list --pretty=oneline ^master "$topic"`
if test -z "$not_in_master"
then
    echo >&2 "$topic is fully merged to master; better remove it."
    exit 1 ;# we could allow it, but there is no point.
fi

# Is topic ever merged to next?  If so you should not be rebasing it.
only_next_1=`git rev-list ^master "^$topic" ${publish} | sort`
only_next_2=`git rev-list ^master           ${publish} | sort`
if test "$only_next_1" = "$only_next_2"
then
```

```
    not_in_topic=`git rev-list "^$topic" master`
    if test -z "$not_in_topic"
    then
        echo >&2 "$topic is already up to date with master"
        exit 1 ;# we could allow it, but there is no point.
    else
        exit 0
    fi
else
    not_in_next=`git rev-list --pretty=oneline ^${publish} "$topic"`
    /usr/bin/perl -e '
        my $topic = $ARGV[0];
        my $msg = "* $topic has commits already merged to public branch:\n";
        my (%not_in_next) = map {
            /^([0-9a-f]+) /;
            ($1 => 1);
        } split(/\n/, $ARGV[1]);
        for my $elem (map {
                /^([0-9a-f]+) (.*)$/;
                [$1 => $2];
            } split(/\n/, $ARGV[2])) {
            if (!exists $not_in_next{$elem->[0]}) {
                if ($msg) {
                    print STDERR $msg;
                    undef $msg;
                }
                print STDERR " $elem->[1]\n";
            }
        }
    ' "$topic" "$not_in_next" "$not_in_master"
    exit 1
fi

<<\DOC_END

This sample hook safeguards topic branches that have been
published from being rewound.

The workflow assumed here is:

 * Once a topic branch forks from "master", "master" is never
   merged into it again (either directly or indirectly).

 * Once a topic branch is fully cooked and merged into "master",
   it is deleted.  If you need to build on top of it to correct
   earlier mistakes, a new topic branch is created by forking at
   the tip of the "master".  This is not strictly necessary, but
   it makes it easier to keep your history simple.

 * Whenever you need to test or publish your changes to topic
   branches, merge them into "next" branch.
```

```
The script, being an example, hardcodes the publish branch name
to be "next", but it is trivial to make it configurable via
$GIT_DIR/config mechanism.

With this workflow, you would want to know:

(1) ... if a topic branch has ever been merged to "next".  Young
    topic branches can have stupid mistakes you would rather
    clean up before publishing, and things that have not been
    merged into other branches can be easily rebased without
    affecting other people.  But once it is published, you would
    not want to rewind it.

(2) ... if a topic branch has been fully merged to "master".
    Then you can delete it.  More importantly, you should not
    build on top of it -- other people may already want to
    change things related to the topic as patches against your
    "master", so if you need further changes, it is better to
    fork the topic (perhaps with the same name) afresh from the
    tip of "master".

Let's look at this example:

           o---o---o---o---o---o---o---o---o---o "next"
          /       /           /           /
         /   a---a---b A     /           /
        /   /               /           /
           /   /   c---c---c---c B         /
          /   /   /             \         /
         /   /   /   b---b C     \       /
        /   /   /   /             \     /
    ---o---o---o---o---o---o---o---o---o---o---o "master"


A, B and C are topic branches.

 * A has one fix since it was merged up to "next".

 * B has finished.  It has been fully merged up to "master" and "next",
   and is ready to be deleted.

 * C has not merged to "next" at all.

We would want to allow C to be rebased, refuse A, and encourage
B to be deleted.

To compute (1):

    git rev-list ^master ^topic next
    git rev-list ^master        next

    if these match, topic has not merged in next at all.
```

```
To compute (2):

    git rev-list master..topic

    if this is empty, it is fully merged to "master".

DOC_END
```

pre-receive.sample

```
#!/bin/sh
#
# An example hook script to make use of push options.
# The example simply echoes all push options that start with 'echoback='
# and rejects all pushes when the "reject" push option is used.
#
# To enable this hook, rename this file to "pre-receive".

if test -n "$GIT_PUSH_OPTION_COUNT"
then
    i=0
    while test "$i" -lt "$GIT_PUSH_OPTION_COUNT"
    do
        eval "value=\$GIT_PUSH_OPTION_$i"
        case "$value" in
        echoback=*)
            echo "echo from the pre-receive-hook: ${value#*=}" >&2
            ;;
        reject)
            exit 1
        esac
        i=$((i + 1))
    done
fi
```

prepare-commit-msg.sample

```
#!/bin/sh
#
# An example hook script to prepare the commit log message.
# Called by "git commit" with the name of the file that has the
# commit message, followed by the description of the commit
# message's source.  The hook's purpose is to edit the commit
# message file.  If the hook fails with a non-zero status,
# the commit is aborted.
#
# To enable this hook, rename this file to "prepare-commit-msg".

# This hook includes three examples. The first one removes the
# "# Please enter the commit message..." help message.
#
# The second includes the output of "git diff --name-status -r"
# into the message, just before the "git status" output.  It is
# commented because it doesn't cope with --amend or with squashed
# commits.
#
# The third example adds a Signed-off-by line to the message, that can
# still be edited.  This is rarely a good idea.

COMMIT_MSG_FILE=$1
COMMIT_SOURCE=$2
SHA1=$3

/usr/bin/perl -i.bak -ne 'print unless(m/^. Please enter the commit message/..m/^#$/)'
"$COMMIT_MSG_FILE"

# case "$COMMIT_SOURCE,$SHA1" in
#  ,|template,)
#    /usr/bin/perl -i.bak -pe '
#       print "\n" . `git diff --cached --name-status -r`
#      if /^#/ && $first++ == 0' "$COMMIT_MSG_FILE" ;;
#  *) ;;
# esac

# SOB=$(git var GIT_COMMITTER_IDENT | sed -n 's/^\(.*>\).*$/Signed-off-by: \1/p')
# git interpret-trailers --in-place --trailer "$SOB" "$COMMIT_MSG_FILE"
# if test -z "$COMMIT_SOURCE"
# then
#   /usr/bin/perl -i.bak -pe 'print "\n" if !$first_line++' "$COMMIT_MSG_FILE"
# fi
```

push-to-checkout.sample

```
#!/bin/sh

# An example hook script to update a checked-out tree on a git push.
#
# This hook is invoked by git-receive-pack(1) when it reacts to git
# push and updates reference(s) in its repository, and when the push
# tries to update the branch that is currently checked out and the
# receive.denyCurrentBranch configuration variable is set to
# updateInstead.
#
# By default, such a push is refused if the working tree and the index
# of the remote repository has any difference from the currently
# checked out commit; when both the working tree and the index match
# the current commit, they are updated to match the newly pushed tip
# of the branch. This hook is to be used to override the default
# behaviour; however the code below reimplements the default behaviour
# as a starting point for convenient modification.
#
# The hook receives the commit with which the tip of the current
# branch is going to be updated:
commit=$1
```

```
# It can exit with a non-zero status to refuse the push (when it does
# so, it must not modify the index or the working tree).
die () {
    echo >&2 "$*"
    exit 1
}

# Or it can make any necessary changes to the working tree and to the
# index to bring them to the desired state when the tip of the current
# branch is updated to the new commit, and exit with a zero status.
#
# For example, the hook can simply run git read-tree -u -m HEAD "$1"
# in order to emulate git fetch that is run in the reverse direction
# with git push, as the two-tree form of git read-tree -u -m is
# essentially the same as git switch or git checkout that switches
# branches while keeping the local changes in the working tree that do
# not interfere with the difference between the branches.

# The below is a more-or-less exact translation to shell of the C code
# for the default behaviour for git's push-to-checkout hook defined in
# the push_to_deploy() function in builtin/receive-pack.c.
#
# Note that the hook will be executed from the repository directory,
# not from the working tree, so if you want to perform operations on
# the working tree, you will have to adapt your code accordingly, e.g.
# by adding "cd .." or using relative paths.

if ! git update-index -q --ignore-submodules --refresh
then
    die "Up-to-date check failed"
fi

if ! git diff-files --quiet --ignore-submodules --
then
    die "Working directory has unstaged changes"
fi

# This is a rough translation of:
#
#   head_has_history() ? "HEAD" : EMPTY_TREE_SHA1_HEX
if git cat-file -e HEAD 2>/dev/null
then
    head=HEAD
else
    head=$(git hash-object -t tree --stdin </dev/null)
fi

if ! git diff-index --quiet --cached --ignore-submodules $head --
then
    die "Working directory has staged changes"
fi
```

```
if ! git read-tree -u -m "$commit"
then
    die "Could not update working tree to new HEAD"
fi
```

sendemail-validate.sample

```
#!/bin/sh

# An example hook script to validate a patch (and/or patch series) before
# sending it via email.
#
# The hook should exit with non-zero status after issuing an appropriate
# message if it wants to prevent the email(s) from being sent.
#
# To enable this hook, rename this file to "sendemail-validate".
#
# By default, it will only check that the patch(es) can be applied on top of
# the default upstream branch without conflicts in a secondary worktree. After
# validation (successful or not) of the last patch of a series, the worktree
# will be deleted.
#
# The following config variables can be set to change the default remote and
# remote ref that are used to apply the patches against:
#
#   sendemail.validateRemote (default: origin)
#   sendemail.validateRemoteRef (default: HEAD)
#
# Replace the TODO placeholders with appropriate checks according to your
# needs.

validate_cover_letter () {
    file="$1"
    # TODO: Replace with appropriate checks (e.g. spell checking).
    true
}

validate_patch () {
    file="$1"
    # Ensure that the patch applies without conflicts.
    git am -3 "$file" || return
    # TODO: Replace with appropriate checks for this patch
    # (e.g. checkpatch.pl).
    true
}

validate_series () {
    # TODO: Replace with appropriate checks for the whole series
    # (e.g. quick build, coding style checks, etc.).
    true
}
```

```
# main -------------------------------------------------------------------------

if test "$GIT_SENDEMAIL_FILE_COUNTER" = 1
then
    remote=$(git config --default origin --get sendemail.validateRemote) &&
    ref=$(git config --default HEAD --get sendemail.validateRemoteRef) &&
    worktree=$(mktemp --tmpdir -d sendemail-validate.XXXXXXX) &&
    git worktree add -fd --checkout "$worktree" "refs/remotes/$remote/$ref" &&
    git config --replace-all sendemail.validateWorktree "$worktree"
else
    worktree=$(git config --get sendemail.validateWorktree)
fi || {
    echo "sendemail-validate: error: failed to prepare worktree" >&2
    exit 1
}

unset GIT_DIR GIT_WORK_TREE
cd "$worktree" &&

if grep -q "^diff --git " "$1"
then
    validate_patch "$1"
else
    validate_cover_letter "$1"
fi &&

if test "$GIT_SENDEMAIL_FILE_COUNTER" = "$GIT_SENDEMAIL_FILE_TOTAL"
then
    git config --unset-all sendemail.validateWorktree &&
    trap 'git worktree remove -ff "$worktree"' EXIT &&
    validate_series
fi
```

update.sample

```
#!/bin/sh
#
# An example hook script to block unannotated tags from entering.
# Called by "git receive-pack" with arguments: refname sha1-old sha1-new
#
# To enable this hook, rename this file to "update".
#
# Config
# ------
# hooks.allowunannotated
#   This boolean sets whether unannotated tags will be allowed into the
#   repository.  By default they won't be.
# hooks.allowdeletetag
#   This boolean sets whether deleting tags will be allowed in the
#   repository.  By default they won't be.
# hooks.allowmodifytag
#   This boolean sets whether a tag may be modified after creation. By default
#   it won't be.
# hooks.allowdeletebranch
#   This boolean sets whether deleting branches will be allowed in the
#   repository.  By default they won't be.
# hooks.denycreatebranch
#   This boolean sets whether remotely creating branches will be denied
#   in the repository.  By default this is allowed.
#

# --- Command line
refname="$1"
oldrev="$2"
newrev="$3"

# --- Safety check
if [ -z "$GIT_DIR" ]; then
    echo "Don't run this script from the command line." >&2
    echo " (if you want, you could supply GIT_DIR then run" >&2
    echo "  $0 <ref> <oldrev> <newrev>)" >&2
    exit 1
fi

if [ -z "$refname" -o -z "$oldrev" -o -z "$newrev" ]; then
    echo "usage: $0 <ref> <oldrev> <newrev>" >&2
    exit 1
fi

# --- Config
allowunannotated=$(git config --type=bool hooks.allowunannotated)
allowdeletebranch=$(git config --type=bool hooks.allowdeletebranch)
denycreatebranch=$(git config --type=bool hooks.denycreatebranch)
allowdeletetag=$(git config --type=bool hooks.allowdeletetag)
allowmodifytag=$(git config --type=bool hooks.allowmodifytag)

# check for no description
projectdesc=$(sed -e '1q' "$GIT_DIR/description")
case "$projectdesc" in
"Unnamed repository"* | "")
    echo "*** Project description file hasn't been set" >&2
    exit 1
    ;;
esac

# --- Check types
# if $newrev is 0000...0000, it's a commit to delete a ref.
zero=$(git hash-object --stdin </dev/null | tr '[0-9a-f]' '0')
if [ "$newrev" = "$zero" ]; then
    newrev_type=delete
else
    newrev_type=$(git cat-file -t $newrev)
fi
```

```
case "$refname","$newrev_type" in
    refs/tags/*,commit)
        # un-annotated tag
        short_refname=${refname##refs/tags/}
        if [ "$allowunannotated" != "true" ]; then
            echo "*** The un-annotated tag, $short_refname, is not allowed in this
repository" >&2
            echo "*** Use 'git tag [ -a | -s ]' for tags you want to propagate." >&2
            exit 1
        fi
        ;;
    refs/tags/*,delete)
        # delete tag
        if [ "$allowdeletetag" != "true" ]; then
            echo "*** Deleting a tag is not allowed in this repository" >&2
            exit 1
        fi
        ;;
    refs/tags/*,tag)
        # annotated tag
        if [ "$allowmodifytag" != "true" ] && git rev-parse $refname > /dev/null 2>&1
        then
            echo "*** Tag '$refname' already exists." >&2
            echo "*** Modifying a tag is not allowed in this repository." >&2
            exit 1
        fi
        ;;
    refs/heads/*,commit)
        # branch
        if [ "$oldrev" = "$zero" -a "$denycreatebranch" = "true" ]; then
            echo "*** Creating a branch is not allowed in this repository" >&2
            exit 1
        fi
        ;;
    refs/heads/*,delete)
        # delete branch
        if [ "$allowdeletebranch" != "true" ]; then
            echo "*** Deleting a branch is not allowed in this repository" >&2
            exit 1
        fi
        ;;
    refs/remotes/*,commit)
        # tracking branch
        ;;
    refs/remotes/*,delete)
        # delete tracking branch
        if [ "$allowdeletebranch" != "true" ]; then
            echo "*** Deleting a tracking branch is not allowed in this repository"
>&2
            exit 1
        fi
        ;;
    *)
        # Anything else (is there anything else?)
        echo "*** Update hook: unknown type of update to ref $refname of type
$newrev_type" >&2
        exit 1
        ;;
esac

# --- Finished
exit 0
```

index

``` DIRC
```