# PRACTICAL 2

**Aim**: To understand the Looping and Selection constructs of Java programming Language

**Prerequisite:**

* Basic knowledge of computer usage.

**Outcome:** After successful completion of this experiment students will be able to,

* Use conditional and multiple *If* statements
* Demonstrate the use *switch* statements
* Demonstrate the use of selection constructs such as *while* statement, *do-while* statements and the *for* statements.
* Demonstrate the use the *break* and *continue* statements

**Theory:**

* 1. If-else Statement

The Java *if statement* is used to test the condition. It checks Boolean condition: *true* or *false*. There are various types of if statement in java.

* if statement
* if-else statement
* if-else-if ladder
* nested if statement

The syntax of the IF statements is as shown below.

**if statement syntax syntax**

**if**(condition){

//code to be executed

}

**if-else statement syntax**

**if**(condition){

//code if condition is true

}**else**{

//code if condition is false

}

**if-else-if ladder syntax**

**if**(condition1){

//code to be executed if condition1 is true

}**else** **if**(condition2){

//code to be executed if condition2 is true

}

**else** **if**(condition3){

//code to be executed if condition3 is true

}

...

**else**{

//code to be executed if all the conditions are false

}

**nested if statement syntax**

**if**(condition){

     //code to be executed

**if**(condition){

             //code to be executed

    }

}

* 1. Switch Statements

The switch statement is a multi-way branch statement. It provides an easy way to dispatch execution to different parts of code based on the value of the expression. Basically, the expression can be byte, short, char, and int primitive data types.

Switch (expression) {

case value :

// Statements

break; // optional

case value :

// Statements

break; // optional

// You can have any number of case statements.

default : // Optional

// Statements

}

* 1. Do-while statements

A **while** loop statement in Java programming language repeatedly executes a target statement as long as a given condition is true.

while(Boolean\_expression) {

// Statements

}

A **do...while** loop is similar to a while loop, except that a do...while loop is guaranteed to execute at least one time.

do {

// Statements

}while(Boolean\_expression);

* 1. For statements
* A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to be executed a specific number of times.
* A **for** loop is useful when you know how many times a task is to be repeated.

for(initialization; Boolean\_expression; update) {

// Statements

}

1. Break and continue statements

The break statement can be used to jump out of a **loop**. The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

(TO BE COMPLETED BY STUDENTS)

|  |  |
| --- | --- |
| Roll No. K005 | Name: Jal Bafana |
| Class: Btech. Cyber Security | Batch: K1 |
| Date of Practical: 20.01.2025 | Date of Submission: 20.01.2025 |
| Grade: |  |

1. Program to reverse a number and check if it’s a palindrome.

package Practical2;

import java.util.Scanner;

public class Pgm1 {

public static void main(String[]args){

Scanner sc = new Scanner(System.in);

// Input

System.out.print("Enter a number: ");

int num = sc.nextInt();

int originalNum = num;

int reversedNum = 0;

// Reversing

while (num != 0) {

int digit = num % 10;

reversedNum = reversedNum \* 10 + digit;

num /= 10;

}

// Checking palindrome

if (originalNum == reversedNum) {

System.out.println(originalNum + " is a palindrome.");

} else {

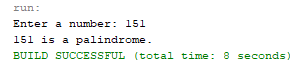
System.out.println(originalNum + " is not a palindrome.");

}

sc.close();

}

}



2. Program to display ‘n’ odd numbers and their sum.

import java.util.Scanner;

package Practical2;

public class Pgm2 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input

System.out.print("Enter the value of n: ");

int n = sc.nextInt();

int sum = 0;

System.out.println("The first " + n + " odd numbers are:");

for (int i = 1, count = 0; count < n; i += 2) {

System.out.print(i + " ");

sum += i;

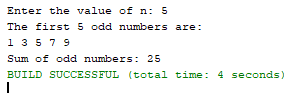
count++;

}

System.out.println("\nSum of odd numbers: " + sum);

sc.close();

}



3. Program to count number of vowels and consonants in a given line of text.

import java.util.Scanner;

package Practical2;

public class Pgm3 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input

System.out.print("Enter a line of text: ");

String text = sc.nextLine();

int vowels = 0, consonants = 0;

text = text.toLowerCase(); // lowercase

// Count vowels and consonants

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

char ch = text.charAt(i);

if (ch >= 'a' && ch <= 'z') {

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {

vowels++;

} else {

consonants++;

}

}

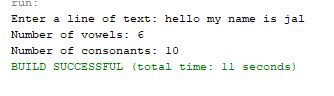
}

System.out.println("Number of vowels: " + vowels);

System.out.println("Number of consonants: " + consonants);

sc.close();

}



4. Program to accept 2 numbers from user and display "increasing" if the numbers are in

increasing order, "decreasing" if the numbers are in decreasing order, and "Neither

increasing or decreasing order" otherwise.

import java.util.Scanner;

package Practical2;

public class Pgm4 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Inputs

System.out.print("Enter first number: ");

int num1 = sc.nextInt();

System.out.print("Enter second number: ");

int num2 = sc.nextInt();

// Check if increasing, decreasing, or neither

if (num1 < num2) {

System.out.println("Increasing");

} else if (num1 > num2) {

System.out.println("Decreasing");

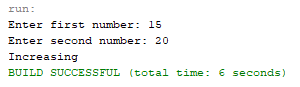
} else {

System.out.println("Neither increasing nor decreasing order");

}

sc.close();

}



5. Program to find the power of a number given its base and exponent value.

import java.util.Scanner;

package Practical2;

public class Pgm5 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input

System.out.print("Enter the base: ");

int base = sc.nextInt();

System.out.print("Enter the exponent: ");

int exponent = sc.nextInt();

int result = 1;

// power

for (int i = 1; i <= exponent; i++) {

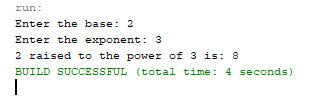
result \*= base;

}

System.out.println(base + " raised to the power of " + exponent + " is: " + result);

sc.close();

}



6. Program to check whether 2 strings are equal, using the string functions.

import java.util.Scanner;

package Practical2;

public class Pgm6 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Inputs

System.out.print("Enter first string: ");

String str1 = sc.nextLine();

System.out.print("Enter second string: ");

String str2 = sc.nextLine();

// Check if strings are equal

if (str1.equals(str2)) {

System.out.println("The strings are equal.");

} else {

System.out.println("The strings are not equal.");

}

sc.close();

}

