**INDEX**

|  |  |  |
| --- | --- | --- |
| S.No. | **Title** | **Remarks** |
| **1.** | Program to find Sum of n natural number. |  |
| **2.** | Program for Narrowing Typecasting. |  |
| **3.** | Program for Widening Typecasting. |  |
| **4.** | Program for typeconversion from int to string. |  |
| **5.** | Program for typeconversion from string to int. |  |
| **6.** | Program to check if the given number is prime or not. |  |
| **7.** | Program to calculate Area of the circle. |  |
| **8.** | Sorting Program |  |
| **9.** | Searching Program |  |
| **10.** | Program to find HCF of two numbers |  |
| **11.** | Program to find the factorial of the number. |  |
| **12.** | Program for Command Line Arguments. |  |
| **13.** | Program to split string to words: |  |
| **14.** | Program to overload methods by parameters. |  |
| **15.** | Program to overload methods on basis of type of parameters. |  |
| **16.** | Default constructors Program. |  |
| **17.** | Parameterized Constructor Program. |  |
| **18.** | Overriding in Polymorphism Program. |  |
| **19.** | Overloading in Polymorphism Program. |  |
| **20.** | Single Inheritance |  |
| **21.** | Multilevel Inheritance |  |
| **22.** | Hierarchial Inheritance |  |
| **23.** | Super Keyword in Inheritance. |  |
| **24.** | Program of Abstract Class. |  |
| **25.** | Program of Abstract Method. |  |
| **26.** | Program for this keyword. |  |
| **27.** | Program for interface in java. |  |
| **28.** | Program of Inheriting Multiple Interface. |  |
| **29.** | Exception Handling using try-catch |  |
| **30.** | Arithmetic Exception using try-catch and finally block. |  |
| **31.** | Multithreading using Thread |  |
| **32.** | Multithreading using runnable |  |
| **33.** | Program for awt button. |  |
| **34.** | Program for AWT GUI extending Frame class. |  |
| **35.** | Swing JButton Program |  |
| **36.** | Swing JTextField Program |  |
| **37.** | Swing JCheckBox Program |  |

1. **Program to find Sum of n natural number.**

// Sum of n natural number

class A{

public int sum(int n){

int sum = 0;

for(int i = 0; i<=n; i++){

sum += i;

}

return sum;

}

}

class \_01\_sum\_of\_n\_nat\_number{

public static void main(String[] args) {

A obj = new A();

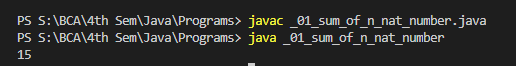
int sum = obj.sum(5);

System.out.println(sum);

}

}

**Output:**



**Typecasting:**

1. **Program for Narrowing Typecasting.**

// narrowing TypeCasting

public class \_02\_typecasting {

public static void main(String[] args) {

// double type variable

double num = 10.99;

System.out.println("Double value: " + num);

// double to int typecasting

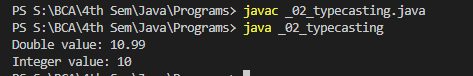
int data = (int)num;

System.out.println("Integer value: " + data);

}

}

**Output:**



1. **Program for Widening Typecasting.**

// widening TypeCasting

public class \_03\_typecasting {

public static void main(String[] args) {

// Int variable

int num = 10;

System.out.println("The integer value: " + num);

// int to double conversion

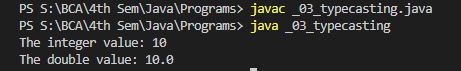
double data = num;

System.out.println("The double value: " + data);

}

}

**Output:**



1. **Program for typeconversion from int to string.**

// int to string typeconversion

public class \_04\_typeconversion {

public static void main(String[] args) {

// int type variable

int num = 10;

System.out.println("Integer value: " + num);

// int to string type

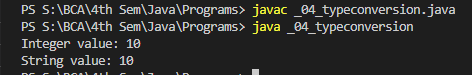
String data = String.valueOf(num);

System.out.println("String value: " + data);

}

}

**Output:**



1. **Program for typeconversion from string to int.**

// string to int typeconversion

public class \_05\_typeconversion {

public static void main(String[] args) {

// String type

String data = "10";

System.out.println("The string value is: " + data);

// string var to int

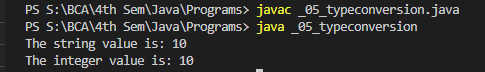
int num = Integer.parseInt(data);

System.out.println("The integer value is: " + num);

}

}

**Output:**



1. **Program to check if the given number is prime or not.**

// Program to check if number is prime or not

public class \_06\_prime {

static boolean isPrime(int num) {

boolean flag = false;

for (int i = 2; i <= num / 2; ++i) {

// condition for nonprime number

if (num % i == 0) {

flag = true;

break;

}

}

if (!flag)

return false;

else

return true;

}

public static void main(String[] args) {

int num = 29;

if (!isPrime(num))

System.out.println(num + " is a prime number.");

else

System.out.println(num + " is not a prime number.");

}

}

**Output:**



1. **Program to calculate Area of the circle.**

// Program to calculatre area of radius

public class \_07\_radius {

public static void main(String args[]){

final double pi = 3.14;

int r = 9;

double area = pi \* r \* r;

System.out.println("Area of Circle of radius " + r + " is: " + area);

}

}

**Output:**



1. **Sorting Program**

import java.util.\*;

public class \_08\_bubble\_sort {

static void bubbleSort(int arr[]) {

int size = arr.length;

for (int i = 0; i < size - 1; i++)

for (int j = 0; j < size - i - 1; j++)

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

public static void main(String args[]) {

int[] arr = { 25, 35, 34, 1, 0 };

// call method using class name

bubbleSort(arr);

System.out.println("Sorted Array in Ascending Order:");

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

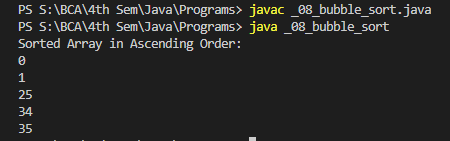
System.out.println(arr[i] );

}

}

}

**Output:**



1. **Searching Program**

public class \_10\_linearSearch {

public static int linearSearch(int[] arr, int element){

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

if(arr[i] == element){

return i;

}

}

return -1;

}

public static void main(String a[]){

int[] a1= {100, 110,120, 130, 140, 150, 180};

int element = 130;

System.out.println(element +" is at index: " + linearSearch(a1, element));

}

}

**Output:**



1. **Program to find HCF of two numbers**

public class \_09\_HCF {

static void HCF(int n1, int n2){

int hcf = 0;

for(int i = 1; i <= Math.min(n1, n2); i++){

if( n1%i == 0 && n2%i == 0){

hcf = i;

}

}

System.out.println(hcf);

}

public static void main(String[] args) {

HCF(12, 18);

}

}

**Output:**



1. **Program to find the factorial of the number.**

public class \_11\_factorail {

public static int factorial(int n){

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

return 1;

}

return n \* factorial(n-1);

}

public static void main(String[] args) {

int n = 5;

System.out.println("factorial of " + n + " is: " + factorial(n));

}

}

**Output:**



1. **Program for Command Line Arguments.**

public class \_12\_command\_line\_args {

public static void main(String args[]){

System.out.println(args.length);

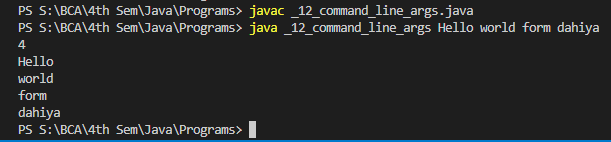
for(int i=0;i<args.length;i++)

System.out.println(args[i]);

}

}

**OUTPUT:**



1. **Program to split string to words:**

public class \_13\_string\_in\_words{

static void splitString(String str){

int i =0;

for( ; i<str.length(); i++){

if(str.charAt(i) == ' '){

System.out.println();

continue;

}

System.out.print(str.charAt(i));

}

}

public static void main(String[] args){

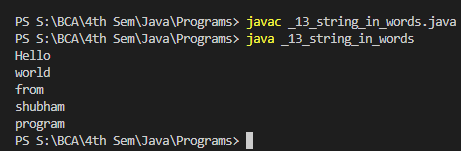
String a = "Hello world from shubham program";

splitString(a);

}

}

**OUTPUT:**



1. **Program to overload methods by parameters.**

//method overloading by parameters

public class \_01\_overloading\_by\_parameters {

private static void display(int a){

System.out.println("Arguments: " + a);

}

private static void display(int a, int b){

System.out.println("Arguments: " + a + " and " + b);

}

public static void main(String[] args) {

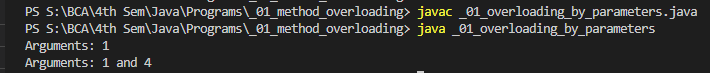
display(1);

display(1, 4);

}

}

**Output:**



1. **Program to overload methods on basis of type of parameters.**

public class \_02\_overloading\_by\_typeOf\_parameter {

// this method accepts int

private static void display(int a){

System.out.println("Integer data.");

}

// this method accepts String object

private static void display(String a){

System.out.println("String object.");

}

public static void main(String[] args) {

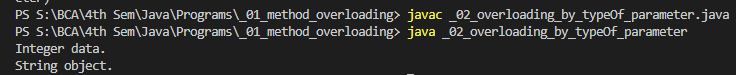
display(1);

display("Hello");

}

}

**Output:**



1. **Default constructors Program.**

class Main{

int a;

double b;

boolean c;

}

public class \_01\_default\_constructor {

public static void main(String[] args) {

// A default constructor is called

Main obj = new Main();

System.out.println("Default Value:");

System.out.println("a = " + obj.a);

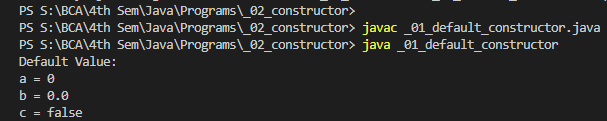
System.out.println("b = " + obj.b);

System.out.println("c = " + obj.c);

}

}

**Output:**



1. **Parameterized Constructor Program.**

class Main {

String name;

// constructor accepting single value

Main(String n) {

name = n;

System.out.println("Hello " + name);

}

}

public class \_02\_parameterized\_constructor {

public static void main(String[] args) {

// call constructor by passing a single value

Main obj1 = new Main("Shubham ");

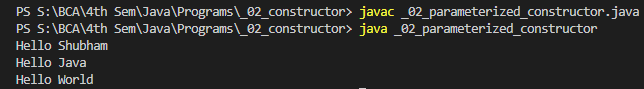
Main obj2 = new Main("Java");

Main obj3 = new Main("World");

}

}

**Output:**



1. **Overriding in Polymorphism Program.**

class Parent {

public void sayHello() {

System.out.println("Hello from Parent");

}

}

class Child extends Parent {

@Override

public void sayHello() {

System.out.println("Hello from Child");

}

}

public class \_01\_Overriding {

public static void main(String[] args) {

// create an object of Child class

Child j1 = new Child();

j1.sayHello();

// create an object of Parent class

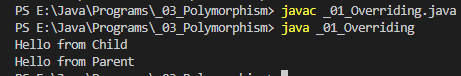
Parent l1 = new Parent();

l1.sayHello();

}

}

**Output:**



1. **Overloading in Polymorphism Program.**

class Pattern {

// method without parameter

public void display() {

for (int i = 0; i < 10; i++) {

System.out.print(".");

}

}

// method with single parameter

public void display(char symbol) {

for (int i = 0; i < 10; i++) {

System.out.print(symbol);

}

}

}

public class \_02\_Overloading {

public static void main(String[] args) {

Pattern d1 = new Pattern();

d1.display();

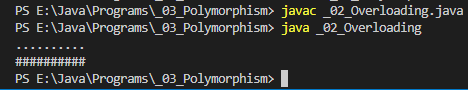
System.out.println();

d1.display('#');

}

}

**Output:**



1. **Single Inheritance**

class Add\_Sub {

int result;

public void addition(int a, int b) {

result = a + b;

System.out.println("sum of numbers:" + result);

}

public void Subtraction(int a, int b) {

result = a - b;

System.out.println("difference between numbers:" + result);

}

}

//inherited class

class Add\_Sub\_Mul extends Add\_Sub {

public void multiplication(int a, int b) {

result = a \* b;

System.out.println("product of numbers:" + result);

}

}

public class \_01\_single\_inheritance {

public static void main(String args[]) {

int a = 15, b = 10;

Add\_Sub\_Mul obj = new Add\_Sub\_Mul();

obj.addition(a, b);

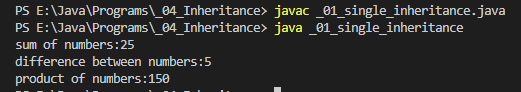
obj.Subtraction(a, b);

obj.multiplication(a, b);

}

}

**Output:**



1. **Multilevel Inheritance**

class Shape {

public void display() {

System.out.println("Inside display");

}

}

class Rectangle extends Shape {

public void area() {

System.out.println("Inside area");

}

}

class Cube extends Rectangle {

public void volume() {

System.out.println("Inside volume");

}

}

public class \_02\_multilevel {

public static void main(String[] arguments) {

Cube cube = new Cube();

cube.display();

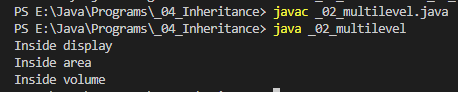
cube.area();

cube.volume();

}

}

**Output:**



1. **Hierarchial Inheritance**

class A {

    public void printA() { System.out.println("Class A"); }

}

class B extends A {

    public void printB() { System.out.println("Class B"); }

}

class C extends A {

    public void printC() { System.out.println("Class C"); }

}

class D extends A {

    public void printD() { System.out.println("Class D"); }

}

public class \_03\_hierarchial {

    public static void main(String[] args)

    {

        B objB = new B();

        objB.printA();

        objB.printB();

        C objC = new C();

        objC.printA();

        objC.printC();

        D objD = new D();

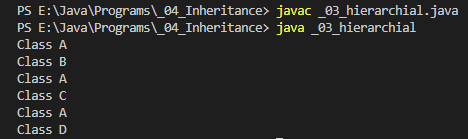
        objD.printA();

        objD.printD();

    }

}

**Output:**



1. **Super Keyword in Inheritance.**

class Super\_class {

int num = 20;

// display method of superclass

public void display() {

System.out.println("This is display method of superclass");

}

}

class Sub\_class extends Super\_class {

int num = 10;

// display method of sub class

public void display() {

System.out.println("This is display method of subclass");

}

public void my\_method() {

// Instantiating subclass

Sub\_class sub = new Sub\_class();

// Invoking the display() method of sub class

sub.display();

// Invoking the display() method of superclass

super.display();

// printing the value of variable num of subclass

System.out.println("variable named num in sub class:"+ sub.num);

// printing the value of variable num of superclass

System.out.println("variable named num in super class:"+ super.num);

}

}

public class \_04\_super {

public static void main(String args[]) {

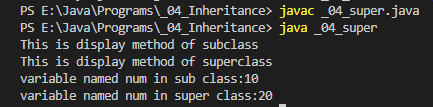
Sub\_class obj = new Sub\_class();

obj.my\_method();

}

}

**Output:**



1. **Program of Abstract Class.**

abstract class abstractClass {

// method of abstract class

public void display() {

System.out.println("This is Java Programming from method defined in abstract class");

}

}

class subclass extends abstractClass {

public subclass() {

display();

}

}

public class \_01\_abstract\_class {

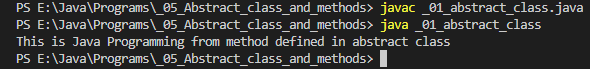
public static void main(String[] args) {

subclass obj = new subclass();

}

}

**Output:**



1. **Program of Abstract Method.**

abstract class MotorBike {

abstract void brake();

}

class SportsBike extends MotorBike {

// implementation of abstract method

public void brake() {

System.out.println("SportsBike Brake");

}

}

class MountainBike extends MotorBike {

// implementation of abstract method

public void brake() {

System.out.println("MountainBike Brake");

}

}

public class \_02\_abstract\_method {

public static void main(String[] args) {

MountainBike obj1 = new MountainBike();

obj1.brake();

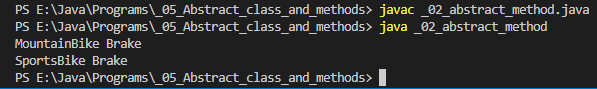
SportsBike obj2 = new SportsBike();

obj2.brake();

}

}

**Output:**



1. **Program for this keyword.**

class C1 {

int age;

C1(int age){

this.age = age;

}

}

public class \_01\_ {

public static void main(String[] args) {

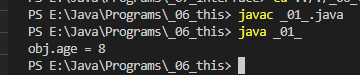
C1 obj = new C1(8);

System.out.println("obj.age = " + obj.age);

}

}

**Output:**



1. **Program for interface in java.**

interface printable{

void print();

}

class Subclass implements printable{

public void print(){System.out.println("Hello from subclass defined method inherited from interface");}

}

public class \_01 {

public static void main(String args[]){

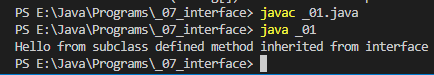
Subclass obj = new Subclass();

obj.print();

}

}

**Output:**



1. **Program of Inheriting Multiple Interface.**

//calculate area of the circle using Interface class.

interface get{                     //creating interface get and declaring a method inside it.

void get\_r(int r);

}

interface ar{                     //creating interface ar and declaring a method inside it.

void area();

}

class calc implements get,ar{       //importing interface using "implements" keyword.

final float pi=3.14f;

float r,ar;

public void get\_r(int r){

this.r= r;                               //for taking input from user.

}

public void area(){

ar=pi\*r\*r;                                     //calculating area using formula.

System.out.println("The area of circle is : " + ar);

}

}

class \_02\_multiple\_inhertance{

public static void main(String args[]){

calc obj=new calc();            //creating object using derived class.

obj.get\_r(4);                    //calling get\_r function using object.

obj.area();                     //calling area function using object.

System.out.println("");

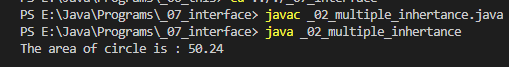
get obj1=new calc();           //creating object using the interface class.

obj1.get\_r(6);               //calling get\_r function using object but we can't call the other methods available in the calc because get interface has only get\_r function.

}

}

**Output:**



1. **Exception Handling using try-catch**

public class \_P01 {

public static void main(String[] args) {

int[] arr= {5, 6, 8, 9, 2};

try {

for (int i = 0; i < 7; i++) {

System.out.println("Value at: " + i + " is: " + arr[i]);

}

}

catch (Exception e) {

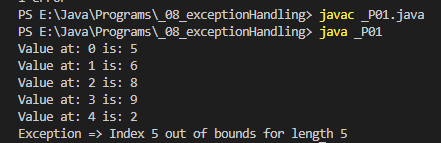
System.out.println("Exception => " + e.getMessage());

}

}

}

**Output:**



1. **Arithmetic Exception using try-catch and finally block.**

public class \_P02\_Arithmethic\_Exception {

public static void main(String[] args) {

try {

// code that generates exception

int result = 5 / 0;

}

catch (ArithmeticException e) {

System.out.println("ArithmeticException => " + e.getMessage());

}

finally {

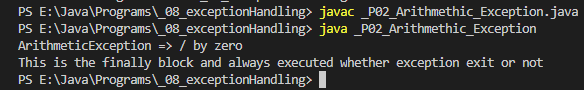
System.out.println("This is the finally block and always executed whether exception exit or not");

}

}

}

**Output:**



1. **Multithreading using Thread**

class A extends Thread{

public void run(){

for(int i = 1; i<5; i++){

System.out.println("Display A");

}

System.out.println("exit A");

}

}

class B extends Thread{

public void run(){

for(int i = 1; i<5; i++){

System.out.println("Display B");

}

System.out.println("exit B");

}

}

class \_01\_using\_thread{

public static void main(String[] args){

A obj1 = new A();

B obj2 = new B();

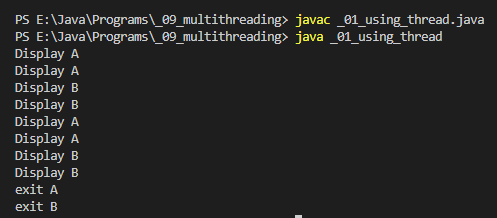
obj1.start();

obj2.start();

}

}

**Output:**



1. **Multithreading using runnable**

class A implements Runnable{

public void run(){

for(int i = 1; i<5; i++){

System.out.println("Display A");

}

System.out.println("exit A");

}

}

class B implements Runnable{

public void run(){

for(int i = 1; i<5; i++){

System.out.println("Display B");

}

System.out.println("exit B");

}

}

class \_02\_using\_runnable{

public static void main(String[] args){

Thread obj1 = new Thread (new A());

Thread obj2 = new Thread(new B());

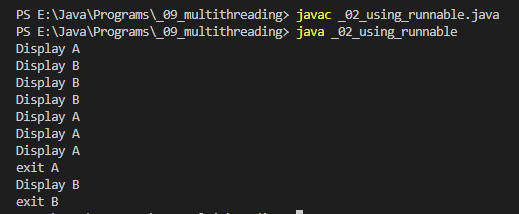
obj1.start();

obj2.start();

}

}

**Output:**



1. **Program for awt button.**

import java.awt.\*;

class awtButton{

public awtButton()

{

Frame f = new Frame();

Button btn=new Button("Hello World");

btn.setBounds(80, 80, 100, 50);

//adding Button.

f.add(btn);

//setting size of frame.

f.setSize(800, 250);

// frame title

f.setTitle("Shubham");

f.setLayout(null);

f.setVisible(true);                   //set frame visibility true.

}

}

public class \_P01\_awt\_button {

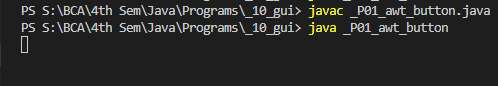
public static void main(String[] args) {

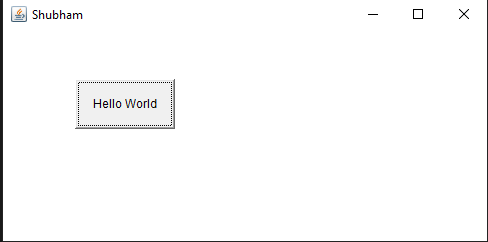
awtButton awt = new awtButton();    //creating frame.

}

}

**OUTPUT:**





1. **Program for AWT GUI extending Frame class.**

import java.awt.\*;

class Awt extends Frame{

Awt(){

Label f\_name = new Label("First Name");

f\_name.setBounds(20, 50, 80, 20);

Label l\_name = new Label("Last Name");

l\_name.setBounds(20, 80, 80, 20);

Label dob = new Label("Date of Birth");

dob.setBounds(20, 110, 80, 20);

TextField f\_ameTF = new TextField();

f\_ameTF.setBounds(120, 50, 100, 20);

TextField l\_NameTF = new TextField();

l\_NameTF.setBounds(120, 80, 100, 20);

TextField dobTF = new TextField();

dobTF.setBounds(120, 110, 100, 20);

Button sbmt = new Button("Submit");

sbmt.setBounds(20, 160, 100, 30);

Button reset = new Button("Reset");

reset.setBounds(120,160,100,30);

add(f\_name);

add(l\_name);

add(dob);

add(f\_ameTF);

add(l\_NameTF);

add(dobTF);

add(sbmt);

add(reset);

setSize(300,200);

setLayout(null);

setVisible(true);

}

}

public class \_P02\_awt extends Frame {

public static void main(String[] args) {

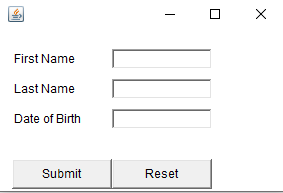
Awt awt = new Awt();

}

}

**OUTPUT:**





1. **Swing JButton Program**

import javax.swing.\*;

import java.awt.event.\*;

import java.awt.\*;

public class \_01\_JButton extends JFrame

{

\_01\_JButton()

{

// Button.

JButton bt1 = new JButton("Shubham Dahiya");

//adding close operation --> Monica mam way

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new FlowLayout());          //setting layout using FlowLayout object

setSize(400, 100);          // size of Jframe

//adding button to frame

add(bt1);

setVisible(true);

}

public static void main(String[] args)

{

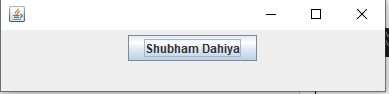
new \_01\_JButton();

}

}

**OUTPUT:**





1. **Swing JTextField Program**

import javax.swing.\*;

import java.awt.event.\*;

import java.awt.\*;

public class \_02\_JTextField extends JFrame

{

public \_02\_JTextField()

{

//creating JTextField.

JTextField jtf = new JTextField(20);

//adding JTextField to frame.

add(jtf);

setLayout(new FlowLayout());

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(400, 100);

setVisible(true);

}

public static void main(String[] args)

{

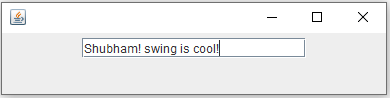
new \_02\_JTextField();

}

}

**OUTPUT:**





1. **Swing JCheckBox Program**

import javax.swing.\*;

import java.awt.event.\*;

import java.awt.\*;

public class \_03\_JCheckBox extends JFrame

{

public \_03\_JCheckBox()

{

//creating JCheckBox.

JCheckBox jcb = new JCheckBox("First");

//adding JCheckBox to frame.

add(jcb);

jcb = new JCheckBox("Second");

add(jcb);

jcb = new JCheckBox("Dahiya");

add(jcb);

setLayout(new FlowLayout());

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(400, 100);

setVisible(true);

}

public static void main(String[] args)

{

new \_03\_JCheckBox();

}

}

**OUTPUT:**



