**OOPs with JAVA LAB MANUAL [21ECI62]**

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**NAGARJUNA COLLEGE OF ENGINEERING & TECHNOLOGY**

(A unit of Nagarjuna Education Society)

### Mudugurki (V), Venkatagirikote (P), Devanahalli (T), Bangalore(R) – 562164

Approved by AICTE/ NBA Accredited/ Affiliated to VTU/ NAAC A+ Accredited

**OOPs with JAVA LAB MANUAL [21ECI62]**

By,

# Mr. MOHAN GOWDA G S

Assistant Professor

Department of Computer Science and Engineering NAGARJUNA COLLEGE OF ENGINEERING & TECHNOLOGY

### A) Develop a Java program for an advanced arithmetic calculator that takes two integer operands and an operator from the user. The program should be capable of performing addition, subtraction, multiplication, and division.

**package** p1;

**import** java.util.Scanner;

**public class** lab1 {

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

**int** a,b; String ch;

Scanner sc = **new** Scanner(System.***in***); System.***out***.println("Enter the operand 1:"); a=sc.nextInt();

System.***out***.println("Enter the operator:"); ch=sc.next();

System.***out***.println("Enter the operand 2:"); b=sc.nextInt();

**switch**(ch) {

**case**"+":

System.***out***.println("the value is:"); System.***out***.println(a+b);

**break**; **case**"-":

System.***out***.println("the value is:"); System.***out***.println(a-b);

**break**; **case**"\*":

System.***out***.println("the value is:"); System.***out***.println(a\*b);

**break**; **case**"/":

System.***out***.println("the value is:"); System.***out***.println((**float**)a/(**float**)b); **break**;

**case**"%":

System.***out***.println(a%b);

### break;

**default**:

System.***out***.println("Invalid Operator!!! Enter the valid operator");

}

}

}

Output:

|  |  |  |
| --- | --- | --- |
| **OUTPUT 1:-**  Enter the operand 1: 6  Enter the operator:  +  Enter the operand 2: 7  the value is: 13 | **OUTPUT 2:-**  Enter the operand 1: 6  Enter the operator:  -  Enter the operand 2: 7  the value is:  **-1** | **OUTPUT 3:-**  Enter the operand 1: 6  Enter the operator:  \*  Enter the operand 2: 7  the value is:  **42** |
| **OUTPUT 4:-**  Enter the operand 1: 6  Enter the operator:  /  Enter the operand 2: 7  the value is: 0.85714287 | **OUTPUT 5:-**  Enter the operand 1: 6  Enter the operator:  %  Enter the operand 2: 7  the value is: 6 | **OUTPUT 6:-**  Enter the operand 1: 4  Enter the operator: 4  Enter the operand 2: 7  Invalid Operator!!! Enter the valid operator |

### B) Write a Java program to generate the first 'n' terms of the Fibonacci series

package p1;

import java.util.Scanner;

public class Lab\_Program\_1b {

public static void main(String[] args)

{

int n, i, first, second, next; System.*out*.println("Enter the value of n"); Scanner sc = new Scanner(System.*in*); n=sc.nextInt();

first=0; second=1;

System.*out*.println("Fibonacci series are:\n"); System.*out*.print(first+"\t"+second);

for(i=2;i<=n-1;i++)

{

next=first+second; System.*out*.print("\t"+next); first=second;

second=next;

}

}}

## OUTPUT:-

Enter the value of n 9

Fibonacci series are:

0 1 1 2 3 5 8 13 21

### A) Develop a Java program showcasing method overloading with a base class "Phone" containing the dial() method, and two subclasses "CameraPhone" and "SmartPhone" that inherit from the base class and enhance its features. The program should demonstrate and print the results of these enhancements.

**package** p1;

**class** phone

{

**void** dial() {

System.***out***.println("Calling friend using this number through a regular phone");

}

}

**class** camera\_phone **extends** phone {

**void** dial(String n) {

System.***out***.println("calling "+n+"using camera phone");

}

**void** take\_photo() {

System.***out***.println("Take photo using camera phone");

}

}

**class** smart\_phone **extends** camera\_phone{

**void** dial(String n , **boolean** b) {

**if**(b) {

}

### else {

}

}

System.***out***.println("calling "+n+"through video call"); System.***out***.println("calling "+n+"through normal voice call");

**void** acces\_internet() {

System.***out***.println("Accessing internet for WWW");

} }

**public class** Lab\_Program2a {

**public static void** main(String[] args) { phone p=**new** phone();

p.dial();

camera\_phone c=**new** camera\_phone(); c.dial();

c.dial("Priya ");

c.take\_photo();

smart\_phone s=**new** smart\_phone(); s.dial("Priya ",**true**);

s .acces\_internet();

}

}

## OUTPUT:-

Calling friend using this number through a regular phone Calling friend using this number through a regular phone calling Priya using camera phone

Take photo using camera phone calling Priya through video call Accessing internet for WWW

### B) Develop a Java program illustrating constructor overloading for calculating the area of a rectangle and a circle using appropriate constructors.

**package** p1;

**import** java.math.\*;

**class** Shape\_A\_C{

Shape\_A\_C(**int** r){

System.***out***.println("A circle is created");

System.***out***.println("Area of circle which was created is "+(Math.***PI***\*r\*r)+" cm2");

}

Shape\_A\_C(**int** l,**int** b){

System.***out***.println("A rectangle is created");

System.***out***.println("Area of rectangle which was created is "+(l\*b)+"cm2");

}

}

**public class** Lab\_Program2b {

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

**new** Shape\_A\_C(4);

**new** Shape\_A\_C(3,4);

}

}

## OUTPUT:-

A circle is created

Area of circle which was created is 50.26548245743669 cm2 A rectangle is created

Area of rectangle which was created is 12cm2

### A) Create a Java program with a vehicle hierarchy, including Vehicle, Car, SportsCar, and Truck classes. Implement methods for starting and stopping in the base class and specialized methods for accelerating, adding turbo boost, and loading cargo in the subclasses, with appropriate method overrides.

**package** p1;

**class** vehicles

{

**void** start()

{

System.***out***.println("vehicle started");

}

**void** stop() { System.***out***.println("vehicle stopped");

}

}

**class** car **extends** vehicles

{

@Override

**void** start()

{

System.***out***.println("car started");

}

@Override

**void** stop() { System.***out***.println("car stopped");

}

**void** accelerate() { System.***out***.println("accelerating in a car");

}

}

**class** sports\_car **extends** vehicles{ @Override

**void** start() { System.***out***.println("sports car started");

}

@Override

**void** stop() {

System.***out***.println("sports car stopped");

}

**void** turbo() {

System.***out***.println("Sports turbo boosting in my sports car");

}

}

**class** truck **extends** vehicles{ @Override

**void** start() { System.***out***.println("truck started");

}

@Override

**void** stop() { System.***out***.println("truck stopped");

}

**void** load\_cargo() {

System.***out***.println("load cargo into the truck");

}

}

**public class** Lab\_Program3a{

**public static void** main(String[] args) { car c1=**new** car();

c1.start(); c1.accelerate(); c1.stop();

sports\_car s1=**new** sports\_car(); s1.start();

s1.turbo();

s1.stop();

truck t1=**new** truck(); t1.start(); t1.load\_cargo(); t1.stop();

}

}

## OUTPUT:

car started accelerating in a car car stopped

sports car started

Sports turbo boosting in my sports car sports car stopped

truck started

load cargo into the truck truck stopped

### 3.B) Create a Java program that models electronic devices (e.g., smartphones, laptops, and tablets) using a common interface for power management. The program should allow users to interact with the devices and control their power state.

**package** p1;

**interface** power\_management{

**void** poweron();

**void** poweroff();

**boolean** ispoweron();

}

**class** electronic\_devices **implements** power\_management{

**boolean** state=**false**; @Override

**public void** poweron() { state=**true**;

System.***out***.println("electronic device is turnes on");

}

@Override

**public void** poweroff() { state=**false**;

System.***out***.println("electronic device is turnes off");

}

@Override

**public boolean** ispoweron(){

**return** state;

}

}

**class** smart\_phone1 **extends** electronic\_devices{

**public void** poweron() { state=**true**;

System.***out***.println("smart phone is turned on");

}

@Override

**public void** poweroff() { state=**false**;

System.***out***.println("smart phone is turned off");

}

### void

dial() {

**if**(ispoweron())

System.***out***.println("call my friend using smart phone");

### else

System.***out***.println("hey idiot,first turn on the phone");

}

}

**class** laptop **extends** electronic\_devices{

**public void** poweron() { state=**true**;

System.***out***.println("laptop is turnes on");

}

@Override

**public void** poweroff() { state=**false**;

System.***out***.println("laptop is turned off");

}

**void** access\_internet() {

**if**(ispoweron())

System.***out***.println("accessing internet using laptop");

### else

System.***out***.println("please turn on laptop");

}

}

**class** tablet **extends** electronic\_devices{

**public void** poweron() { state=**true**;

System.***out***.println("tablet is turned on");

}

@Override

**public void** poweroff() { state=**false**; System.***out***.println("tablet off");

}

**void** takephoto() {

**if**(ispoweron())

System.***out***.println("taking photo using tablet");

### else

System.***out***.println("please turn on tablet");

}

}

**public class** Lab\_Program3b {

**public static void** main(String[] args) { smart\_phone1 oneplus=**new** smart\_phone1(); oneplus.poweron();

oneplus.dial(); oneplus.poweroff(); oneplus.dial(); System.***out***.println(); tablet ipad=**new** tablet(); ipad.poweron(); ipad.takephoto(); ipad.poweroff(); ipad.takephoto();

}

}

**OUTPUT:**

smart phone is turned on

call my friend using smart phone smart phone is turned off

hey idiot,first turn on the phone

tablet is turned on taking photo using tablet tablet off

please turn on tablet

### A) Develop a Java program that emulates a library system. Create two packages,

**`library` and `patron`. In the `library` package, define a `Book` class with a private title field. In the `patron` package, implement a `Patron` class that can borrow books. Demonstrate the use of packages, access protection, and class imports. Ensure that the book title remains inaccessible from outside the `library` package due to the `private` access modifier. Create a scenario where a patron, Alice, borrows a book from the library.**



**package** library;

**public class** Book { **private** String title=**null**; **public** Book(String title)

{

**this**.title=title;

System.***out***.println("The book titled "+**this**.title+" is added to the library");

}

**public** String getTitle()

{

**return** title;

}

}





**package** patron; **import** library.Book; **public class** Patron

{

**public** String name=**null**;

Patron(String name)

{

**this**.name=name;

System.***out***.println("New patron "+**this**.name+" is added to the database");

}

**void** borrow\_books(Book b)

{

System.***out***.println(**this**.name+" borrowed "+b.getTitle());

}

}



**package** patron;

**import** library.Book;

**public class** Demo\_Library\_Management\_System {

**public static void** main(String[] args) { Book b1=**new** Book("AR sir's notes"); Book b2=**new** Book("Little champs"); Patron p1=**new** Patron("Chandana"); Patron p2=**new** Patron("Deeksha"); p1.borrow\_books(b1); p1.borrow\_books(b2);

}

}

## OUTPUT:

The book titled AR sir's notes is added to the library The book titled Little champs is added to the library New patron Chandana is added to the database

New patron Deeksha is added to the database Chandana borrowed AR sir's notes Chandana borrowed Little champs

### 4.B) Develop a Java lab program that handles exceptions for division by zero and invalid input. Use `try-catch` blocks to catch `ArithmeticException` for division by zero and

**`InputMismatchException` for non-integer input and provide user-friendly error messages.**

package p1;

import java.util.Scanner;

import java.util.InputMismatchException;

public class Lab\_Program4b {

public static void main(String[] args) { Scanner s=new Scanner(System.in); int c=0,a=0,b=0;

try {

while(true) {

try {

}

System.out.println("Enter the numerator:"); a=s.nextInt();

break;

catch(InputMismatchException f) {

System.out.println("Give only integer input for numerator"); s.nextLine();

}

}

while(true){

try {

}

System.out.println("Enter the denominator:"); b=s.nextInt();

c=a/b; break;

catch(InputMismatchException f) {

System.out.println("Give only integer input for denominator"); s.nextLine();

}

catch(ArithmeticException f) {

System.out.println("The denominator value should be greater than zero"); s.nextLine();

}

}

}

finally{

s.close();

}

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

}

}

## OUTPUT:

Enter the numerator:

4

Enter the denominator:

3

c=1

### A) Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer for every 1 second; second thread computes the square of the number and prints; third thread will print the value of cube of the number.

**package** p1;

**import** java.util.Random;

**class** RandomNumbers **implements** Runnable{ @Override

**public void** run() {

**for**(**int** i=0;i<10;i++) { Random rn=**new** Random(); **int** num=rn.nextInt(10); System.***out***.println(num); **new** Sq\_Of\_Random(num);

**new** Cube\_Of\_Random(num);

**try** { Thread.*sleep*(1000);

}**catch**(InterruptedException e) { e.printStackTrace();

}

}

}

}

**class** Sq\_Of\_Random **implements** Runnable{

**int** num; Sq\_Of\_Random(**int** num){ **this**.num=num;

**new** Thread(**this**).start();

}

@Override

**public void** run() { num=num\*num; System.***out***.println(num);

}

}

**class** Cube\_Of\_Random **implements** Runnable{

**int** num; Cube\_Of\_Random(**int** num){ **this**.num=num;

**new** Thread(**this**).start();

}

@Override

**public void** run() { num=num\*num\*num; System.***out***.println(num);

}

}

**public class** Lab\_Program\_5a {

**public static void** main(String[] args) { Runnable r1=**new** RandomNumbers(); **new** Thread(r1).start();

}

}

## OUTPUT:

4

16

64

4

16

64

6

36

216

1

1

### 5.B) Design a Java lab program to demonstrate string handling, including creating strings using constructors and literals, concatenating strings, extracting characters at a specified index, and comparing strings for equality.

**package** p1;

**import** java.util.Scanner;

**public class** Lab\_Program\_5b {

**public static void** main(String[] args) { Scanner Scn = **new** Scanner(System.***in***); System.***out***.println("Enter the first string :"); String S1 = Scn.next(); System.***out***.println("Enter the second string : "); String S2 = Scn.next();

String S3 = **new** String(S1.concat(S2)); System.***out***.println("The concatenation string is :" + S3);

System.***out***.println("Extract portion from concatenated string is :" + S3.substring(3));

**if** (S1.equals(S2)) {

System.***out***.println("Strings you have entered are equal ");

} **else** {

System.***out***.println("Strings you have entered are not equal");

}

Scn.close();

}

}

**OUTPUT:**

|  |  |
| --- | --- |
| **OUTPUT 1:**  Enter the first string :  CAT  Enter the second string :  RAT  The concatenation string is :CATRAT Extract portion from concatenated string is  :RAT  Strings you have entered are not equal | **OUTPUT 2:**  Enter the first string :  CAT  Enter the second string :  CAT  The concatenation string is :CATCAT Extract portion from concatenated string is  :CAT  Strings you have entered are equal |