1.     Method Overloading: Write a class Calculator with overloaded methods add(). Implement add() methods that take:

     - Two integers

     - Two double values

     - Three integers

     - A variable number of integers

Code:-

**package** mypackage;

**import** java.util.Scanner;

**class** Calculator {

// Method to add two integers

**public** **int** add(**int** a, **int** b) {

**return** a + b;

}

// Method to add two double values

**public** **double** add(**double** a, **double** b) {

**return** a + b;

}

// Method to add three integers

**public** **int** add(**int** a, **int** b, **int** c) {

**return** a + b + c;

}

// Method to add a variable number of integers

**public** **int** add(**int**... numbers) {

**int** sum = 0;

**for** (**int** number : numbers) {

sum += number;

}

**return** sum;

}

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

Calculator calc = **new** Calculator();

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

System.***out***.println("Choose the operation:");

System.***out***.println("1. Add two integers");

System.***out***.println("2. Add two double values");

System.***out***.println("3. Add three integers");

System.***out***.println("4. Add a variable number of integers");

**int** choice = scanner.nextInt();

**switch** (choice) {

**case** 1:

System.***out***.println("Enter two integers:");

**int** int1 = scanner.nextInt();

**int** int2 = scanner.nextInt();

System.***out***.println("Result: " + calc.add(int1, int2));

**break**;

**case** 2:

System.***out***.println("Enter two double values:");

**double** double1 = scanner.nextDouble();

**double** double2 = scanner.nextDouble();

System.***out***.println("Result: " + calc.add(double1, double2));

**break**;

**case** 3:

System.***out***.println("Enter three integers:");

**int** int3 = scanner.nextInt();

**int** int4 = scanner.nextInt();

**int** int5 = scanner.nextInt();

System.***out***.println("Result: " + calc.add(int3, int4, int5));

**break**;

**case** 4:

System.***out***.println("Enter the number of integers you want to add:");

**int** count = scanner.nextInt();

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

System.***out***.println("Enter the integers:");

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

numbers[i] = scanner.nextInt();

}

System.***out***.println("Result: " + calc.add(numbers));

**break**;

**default**:

System.***out***.println("Invalid choice");

**break**;

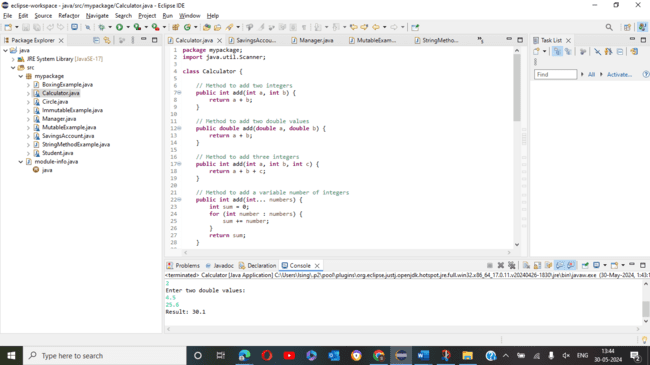
}

scanner.close();

}

}

Output:-



2. Super Keyword: Create a class Person with a constructor that accepts and sets name and age.

   - Create a subclass Student that adds a grade property and initializes name and age using the super keyword in its constructor.

   - Demonstrate the creation of Student objects and the usage of super to call the parent class constructor.

Code:-

**package** mypackage;

**import** java.util.Scanner;

**class** Person {

String name;

**int** age;

**public** Person(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

}

**class** Student **extends** Person {

String grade;

**public** Student(String name, **int** age, String grade) {

**super**(name, age);

**this**.grade = grade;

}

**public** **void** display() {

System.***out***.println("Name: " + name + ", Age: " + age + ", Grade: " + grade);

}

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

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

System.***out***.println("Enter student's name:");

String name = scanner.nextLine();

System.***out***.println("Enter student's age:");

**int** age = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

System.***out***.println("Enter student's grade:");

String grade = scanner.nextLine();

Student student = **new** Student(name, age, grade);

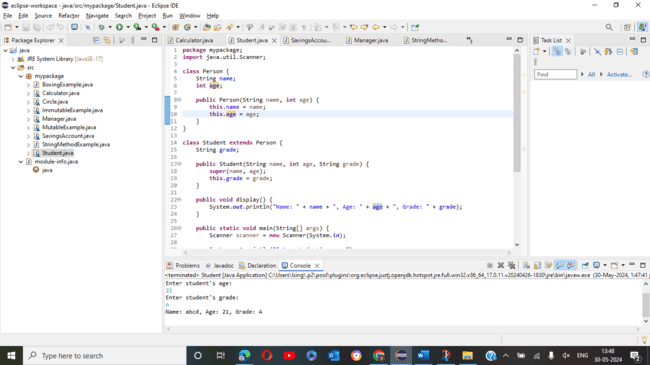
student.display(); // Display student details

scanner.close();

}

}

Output:-



3. Super Keyword: Create a base class Shape with a method draw() that prints "Drawing Shape".

   - Create a subclass Circle that overrides draw() to print "Drawing Circle".

   - Inside the draw() method of Circle, call the draw() method of the Shape class using super.draw().

   - Write a main method to demonstrate calling draw() on a Circle object.

Code:-

**package** mypackage;

**class** Shape {

**public** **void** draw() {

System.***out***.println("Drawing Shape");

}

}

**class** Circle **extends** Shape {

@Override

**public** **void** draw() {

**super**.draw(); // Calls the draw method in Shape

System.***out***.println("Drawing Circle");

}

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

Circle circle = **new** Circle();

circle.draw(); // Drawing Shape, Drawing Circle

}

}

Output:-

A screenshot of a computer

Description automatically generated

4. Create a base class BankAccount with a method deposit(amount) and a constructor that sets the initial balance.

   - Create a subclass SavingsAccount that overrides deposit(amount) to add interest before depositing. Use the super keyword to call the deposit method of the base class.

   - Write a main method to demonstrate creating a SavingsAccount and depositing an amount to see the effect of interest.

Code:-

**package** mypackage;

**class** BankAccount {

**protected** **double** balance;

**public** BankAccount(**double** initialBalance) {

**this**.balance = initialBalance;

}

**public** **void** deposit(**double** amount) {

balance += amount;

}

**public** **double** getBalance() {

**return** balance;

}

}

**class** SavingsAccount **extends** BankAccount {

**private** **double** interestRate;

**public** SavingsAccount(**double** initialBalance, **double** interestRate) {

**super**(initialBalance);

**this**.interestRate = interestRate;

}

@Override

**public** **void** deposit(**double** amount) {

**double** interest = amount \* interestRate / 100;

**super**.deposit(amount + interest);

}

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

SavingsAccount savings = **new** SavingsAccount(1000, 5);

savings.deposit(100);

System.***out***.println("Balance: " + savings.getBalance()); // Balance: 1105.0

}

}

Output:-

A screenshot of a computer

Description automatically generated

5. Define a class Employee with properties name and salary and a method displayDetails().

   - Create a subclass Manager that adds a property department and overrides displayDetails() to include department details. Use the super keyword to call the displayDetails() method of Employee within Manager.

   - In the main method, create objects of Employee and Manager and call displayDetails() to show the details.

Code:-

**package** mypackage;

**import** java.util.Scanner;

**class** Employee {

String name;

**double** salary;

**public** Employee(String name, **double** salary) {

**this**.name = name;

**this**.salary = salary;

}

**public** **void** displayDetails() {

System.***out***.println("Name: " + name + ", Salary: " + salary);

}

}

**class** Manager **extends** Employee {

String department;

**public** Manager(String name, **double** salary, String department) {

**super**(name, salary);

**this**.department = department;

}

@Override

**public** **void** displayDetails() {

**super**.displayDetails();

System.***out***.println("Department: " + department);

}

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

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

// Input for Employee

System.***out***.println("Enter Employee details:");

System.***out***.print("Name: ");

String empName = scanner.nextLine();

System.***out***.print("Salary: ");

**double** empSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

Employee emp = **new** Employee(empName, empSalary);

// Input for Manager

System.***out***.println("Enter Manager details:");

System.***out***.print("Name: ");

String mgrName = scanner.nextLine();

System.***out***.print("Salary: ");

**double** mgrSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

System.***out***.print("Department: ");

String mgrDepartment = scanner.nextLine();

Manager mgr = **new** Manager(mgrName, mgrSalary, mgrDepartment);

// Display details

System.***out***.println("\nEmployee Details:");

emp.displayDetails();

System.***out***.println("\nManager Details:");

mgr.displayDetails();

scanner.close();

}

}

Output:-

A screenshot of a computer

Description automatically generated

6. Write the same programme for the class ImmutableExample, to achieve object value ‘Hi’.

Code:-

**package** mypackage;

**final** **class** ImmutableExample {

**private** **final** String value;

**public** ImmutableExample(String value) {

**this**.value = value;

}

**public** String getValue() {

**return** value;

}

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

ImmutableExample example = **new** ImmutableExample("Hi");

System.***out***.println(example.getValue()); // Hi

}

}

Output:- A screenshot of a computer

Description automatically generated

7. Write the same programme for the class MutableExample, to output the object values ‘hello 2’ and ‘hello3’.

Code:-

**package** mypackage;

**class** MutableExample {

**private** String value;

**public** MutableExample(String value) {

**this**.value = value;

}

**public** String getValue() {

**return** value;

}

**public** **void** setValue(String value) {

**this**.value = value;

}

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

MutableExample example1 = **new** MutableExample("hello 2");

MutableExample example2 = **new** MutableExample("hello 3");

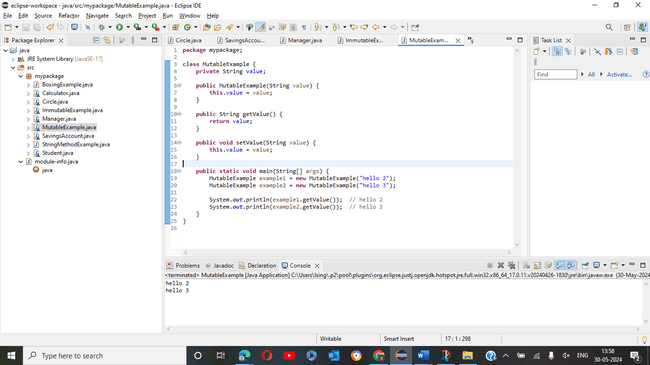
System.***out***.println(example1.getValue()); // hello 2

System.***out***.println(example2.getValue()); // hello 3

}

}

Output:-



8.     Write a java class to implement any 10 string methods:

● replace ● contains ● replaceAll ● indexOf ● substring ● Equals ● lastIndexOf ● startsWith

● endsWith ● EqualsIgnoreCase ● toLowerCase ● toUpperCase ● isEmpty ● Length ● split

Code:-

**package** mypackage;

**public** **class** StringMethodExample {

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

String str = "Hello, World!";

// replace

String replaced = str.replace('o', 'a');

System.***out***.println(replaced); // Hella, Warld!

// contains

**boolean** containsHello = str.contains("Hello");

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

// replaceAll

String replacedAll = str.replaceAll("o", "a");

System.***out***.println(replacedAll); // Hella, Warld!

// indexOf

**int** indexOfW = str.indexOf('W');

System.***out***.println(indexOfW); // 7

// substring

String substring = str.substring(7);

System.***out***.println(substring); // World!

// equals

**boolean** equalsStr = str.equals("Hello, World!");

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

// lastIndexOf

**int** lastIndexOfO = str.lastIndexOf('o');

System.***out***.println(lastIndexOfO); // 8

// startsWith

**boolean** startsWithHello = str.startsWith("Hello");

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

// endsWith

**boolean** endsWithWorld = str.endsWith("World!");

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

// equalsIgnoreCase

**boolean** equalsIgnoreCaseStr = str.equalsIgnoreCase("hello, world!");

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

// toLowerCase

String lowerCaseStr = str.toLowerCase();

System.***out***.println(lowerCaseStr); // hello, world!

// toUpperCase

String upperCaseStr = str.toUpperCase();

System.***out***.println(upperCaseStr); // HELLO, WORLD!

// isEmpty

**boolean** isEmptyStr = str.isEmpty();

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

// length

**int** lengthStr = str.length();

System.***out***.println(lengthStr); // 13

// split

String[] splitStr = str.split(", ");

**for** (String s : splitStr) {

System.***out***.println(s); // Hello, World!

}

}

}

Output:-

A screenshot of a computer

Description automatically generated