**Question 01:**

Create a Java program defining a **Person** class with the following specifications:

* Two private fields: **name** (String) and **age** (int).
* Two constructors: a default constructor and a parameterized constructor that takes a name and an age.
* Getter and setter methods for both fields.
* An overridden **toString()** method that returns a string representation of the person's name and age.
* In the **main** method, create an instance of the **Person** class with the name "Alice" and age 28. Print the information using the **toString()** method.
* Use the setter methods to update the person's name to "Bob" and age to 35. Print the updated information.

Write the Java program for the above specifications and include the output that would be generated when the program is executed.

**Sample Answer:**

public class Person {

// Fields

private String name;

private int age;

// Constructors

public Person() {

// Default constructor

}

public Person(String name, int age) {

this.name = name;

this.age = age;

}

// Getters and Setters

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

// toString() method override

@Override

public String toString() {

return "Person [name=" + name + ", age=" + age + "]";

}

}

public class PersonTest {

public static void main(String[] args) {

// Using the default constructor

Person person1 = new Person();

person1.setName("Alice");

person1.setAge(28);

System.out.println("Person 1: " + person1.toString());

// Using the parameterized constructor

Person person2 = new Person("Bob", 35);

System.out.println("Person 2: " + person2.toString());

}

}

**Question 02:**

Create a Java program for a **Book** class with the following specifications:

1. Two private fields: **title** (String) and **author** (String).
2. Two constructors: a default constructor and a parameterized constructor that takes a title and an author.
3. Getter and setter methods for both fields.
4. An overridden **toString()** method that returns a string representation of the book's title and author.
5. In the **main** method, create an array of **Book** objects to store information about three books.
6. Initialize the array with different books using the parameterized constructor.
7. Use a loop to print the information of each book using the **toString()** method.

Write the Java program for the above specifications and include the output that would be generated when the program is executed.

**Sample Answer:**

// Book class

public class Book {

// Fields

private String title;

private String author;

// Constructors

public Book() {

// Default constructor

}

public Book(String title, String author) {

this.title = title;

this.author = author;

}

// Getters and Setters

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

// toString() method override

@Override

public String toString() {

return "Book [title=" + title + ", author=" + author + "]";

}

}

// BookTest class

public class BookTest {

public static void main(String[] args) {

// Creating individual Book objects

Book book1 = new Book("The Great Gatsby", "F. Scott Fitzgerald");

Book book2 = new Book("To Kill a Mockingbird", "Harper Lee");

Book book3 = new Book("1984", "George Orwell");

// Printing information of each book

System.out.println(book1.toString());

System.out.println(book2.toString());

System.out.println(book3.toString());

}

}

**Question 03:**

Create a Java program for a **Car** class with the following specifications:

1. Two private fields: **brand** (String) and **year** (int).
2. Two constructors: a default constructor and a parameterized constructor that takes a brand and a year.
3. Getter and setter methods for both fields.
4. An overridden **toString()** method that returns a string representation of the car's brand and year.
5. In a separate class (**CarTest**), in the **main** method:
   * Create an instance of the **Car** class with the brand "Toyota" and year 2022. Print the information using the **toString()** method.
   * Use the setter methods to update the car's brand to "Honda" and year to 2023. Print the updated information.

Write the Java program for the above specifications and include the output that would be generated when the program is executed.

**Sample Answer:**

// Car class

public class Car {

// Fields

private String brand;

private int year;

// Constructors

public Car() {

// Default constructor

}

public Car(String brand, int year) {

this.brand = brand;

this.year = year;

}

// Getters and Setters

public String getBrand() {

return brand;

}

public void setBrand(String brand) {

this.brand = brand;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

// toString() method override

@Override

public String toString() {

return "Car [brand=" + brand + ", year=" + year + "]";

}

}

// CarTest class

public class CarTest {

public static void main(String[] args) {

// Creating an instance of the Car class

Car car1 = new Car("Toyota", 2022);

System.out.println("Original information: " + car1.toString());

// Using setters to update information

car1.setBrand("Honda");

car1.setYear(2023);

System.out.println("Updated information: " + car1.toString());

}

}

Output:

Original information: Car [brand=Toyota, year=2022]

Updated information: Car [brand=Honda, year=2023]

**Question 04:**

Create a Java program for a **Movie** class with the following specifications:

1. Five private fields: **title** (String) , **genre** (String), String **title**, String **director**, int **releaseYear**, String **genre**, double **rating**, int **duration**, String **language**, boolean **isReleased** .
2. Two constructors: a default constructor and a parameterized constructor that takes all the fields.
3. Getter and setter methods for both fields.
4. An overridden **toString()** method that returns a string representation of the movie's title, genre, language, duration and rating.
5. In a separate class (**MovieTest**), in the **main** method:
   * Create an instance of the **Movie** class with the title "Inception" and genre "Sci-Fi". Print the information using the **toString()** method.
   * Use the setter methods to update the movie's title to "The Matrix" and genre to "Action". Print the updated information.

Write the Java program for the above specifications and include the output that would be generated when the program is executed.

**Sample Answer:**

// Movie class

public class Movie {

// Fields

private String title;

private String director;

private int releaseYear;

private String genre;

private double rating;

private int duration; // in minutes

private String language;

private boolean isReleased;

// Constructors

public Movie() {

// Default constructor

}

public Movie(String title, String director, int releaseYear, String genre, double rating,

int duration, String language, boolean isReleased) {

this.title = title;

this.director = director;

this.releaseYear = releaseYear;

this.genre = genre;

this.rating = rating;

this.duration = duration;

this.language = language;

this.isReleased = isReleased;

}

// Getters and Setters

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getDirector() {

return director;

}

public void setDirector(String director) {

this.director = director;

}

public int getReleaseYear() {

return releaseYear;

}

public void setReleaseYear(int releaseYear) {

this.releaseYear = releaseYear;

}

public String getGenre() {

return genre;

}

public void setGenre(String genre) {

this.genre = genre;

}

public double getRating() {

return rating;

}

public void setRating(double rating) {

this.rating = rating;

}

public int getDuration() {

return duration;

}

public void setDuration(int duration) {

this.duration = duration;

}

public String getLanguage() {

return language;

}

public void setLanguage(String language) {

this.language = language;

}

public boolean isReleased() {

return isReleased;

}

public void setReleased(boolean released) {

isReleased = released;

}

// toString() method override

@Override

public String toString() {

return "Movie{" +

"title='" + title + '\'' +

", director='" + director + '\'' +

", releaseYear=" + releaseYear +

", genre='" + genre + '\'' +

", rating=" + rating +

", duration=" + duration +

", language='" + language + '\'' +

", isReleased=" + isReleased +

'}';

}

}

// MovieTest class

public class MovieTest {

public static void main(String[] args) {

// Creating an instance of the Movie class

Movie movie1 = new Movie("Inception", "Christopher Nolan", 2010, "Sci-Fi", 8.8, 148, "English", true);

System.out.println("Original information:\n" + movie1.toString());

// Using setters to update information

movie1.setTitle("The Dark Knight");

movie1.setDirector("Christopher Nolan");

movie1.setReleaseYear(2008);

movie1.setGenre("Action");

movie1.setRating(9.0);

movie1.setDuration(152);

movie1.setLanguage("English");

movie1.setReleased(true);

System.out.println("\nUpdated information:\n" + movie1.toString());

}

}

**Exercise 5: Create a Class with Attributes and Methods**

**Problem:**

Create a class Car with attributes brand, model, and year. Add a method displayInfo() that prints the car's details. Instantiate an object and display its details.

**Solution:**

public class Car {

String brand;

String model;

int year;

// Constructor

Car(String brand, String model, int year) {

this.brand = brand;

this.model = model;

this.year = year;

}

// Method to display car information

void displayInfo() {

System.out.println("Car: " + brand + " " + model + ", Year: " + year);

}

public static void main(String[] args) {

Car car1 = new Car("Toyota", "Corolla", 2022);

car1.displayInfo();

}

}

**Output:**

Car: Toyota Corolla, Year: 2022

**Exercise 6: Class with Getters and Setters**

**Problem:**

Create a class Student with private attributes name and age. Use getter and setter methods to access and modify these attributes.

**Solution:**

class Student {

private String name;

private int age;

// Setter methods

public void setName(String name) {

this.name = name;

}

public void setAge(int age) {

if (age > 0) {

this.age = age;

} else {

System.out.println("Age must be positive.");

}

}

// Getter methods

public String getName() {

return name;

}

public int getAge() {

return age;

}

public static void main(String[] args) {

Student student1 = new Student();

student1.setName("Alice");

student1.setAge(20);

System.out.println("Student Name: " + student1.getName());

System.out.println("Student Age: " + student1.getAge());

}

}

**Output:**

Student Name: Alice

Student Age: 20

**Exercise 7: Class with a Constructor and Multiple Objects**

**Problem:**

Create a class Book with attributes title, author, and price. Use a constructor to initialize values. Create multiple objects and display their details.

**Solution:**

class Book {

String title;

String author;

double price;

// Constructor

Book(String title, String author, double price) {

this.title = title;

this.author = author;

this.price = price;

}

// Method to display book details

void displayBook() {

System.out.println("Book: " + title + " by " + author + ", Price: $" + price);

}

public static void main(String[] args) {

Book book1 = new Book("Java Programming", "James Gosling", 45.99);

Book book2 = new Book("Python Basics", "Guido van Rossum", 39.99);

book1.displayBook();

book2.displayBook();

}

}

**Output:**

Book: Java Programming by James Gosling, Price: $45.99

Book: Python Basics by Guido van Rossum, Price: $39.99

**Exercise 8: Class with a Static Variable**

**Problem:**

Create a class Employee with attributes name and salary. Use a static variable to keep track of the total number of employees.

**Solution:**

class Employee {

String name;

double salary;

static int employeeCount = 0;

// Constructor

Employee(String name, double salary) {

this.name = name;

this.salary = salary;

employeeCount++;

}

// Method to display employee details

void displayEmployee() {

System.out.println("Employee: " + name + ", Salary: $" + salary);

}

public static void main(String[] args) {

Employee emp1 = new Employee("John", 50000);

Employee emp2 = new Employee("Emma", 60000);

emp1.displayEmployee();

emp2.displayEmployee();

System.out.println("Total Employees: " + Employee.employeeCount);

}

}

**Output:**

Employee: John, Salary: $50000.0

Employee: Emma, Salary: $60000.0

Total Employees: 2

**Exercise 9: Bank Account Class**

**Problem:**

Create a BankAccount class with the attributes accountNumber, accountHolder, and balance. Implement methods deposit(double amount) and withdraw(double amount), ensuring withdrawal does not exceed balance.

**Solution:**

public class BankAccount {

private String accountNumber;

private String accountHolder;

private double balance;

// Constructor

public BankAccount(String accountNumber, String accountHolder, double balance) {

this.accountNumber = accountNumber;

this.accountHolder = accountHolder;

this.balance = balance;

}

// Deposit method

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("Deposited: $" + amount);

} else {

System.out.println("Invalid deposit amount.");

}

}

// Withdraw method

public void withdraw(double amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

System.out.println("Withdrawn: $" + amount);

} else {

System.out.println("Insufficient balance or invalid amount.");

}

}

// Display account details

public void displayAccount() {

System.out.println("Account Number: " + accountNumber);

System.out.println("Account Holder: " + accountHolder);

System.out.println("Balance: $" + balance);

}

public static void main(String[] args) {

BankAccount acc = new BankAccount("123456", "John Doe", 1000);

acc.displayAccount();

acc.deposit(500);

acc.withdraw(300);

acc.displayAccount();

}

}

**Output:**

Account Number: 123456

Account Holder: John Doe

Balance: $1000.0

Deposited: $500.0

Withdrawn: $300.0

Account Number: 123456

Account Holder: John Doe

Balance: $1200.0

**Exercise 10: Rectangle Class**

**Problem:**

Create a Rectangle class with attributes length and width. Implement methods to calculate and return the **area** and **perimeter**.

**Solution:**

class Rectangle {

private double length;

private double width;

// Constructor

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

// Method to calculate area

public double getArea() {

return length \* width;

}

// Method to calculate perimeter

public double getPerimeter() {

return 2 \* (length + width);

}

public static void main(String[] args) {

Rectangle rect = new Rectangle(10, 5);

System.out.println("Area: " + rect.getArea());

System.out.println("Perimeter: " + rect.getPerimeter());

}

}

**Output:**

Area: 50.0

Perimeter: 30.0

**Exercise 11: Temperature Converter**

**Problem:**

Create a class TemperatureConverter with methods to **convert Celsius to Fahrenheit** and **Fahrenheit to Celsius**.

**Solution:**

class TemperatureConverter {

// Convert Celsius to Fahrenheit

public double celsiusToFahrenheit(double celsius) {

return (celsius \* 9/5) + 32;

}

// Convert Fahrenheit to Celsius

public double fahrenheitToCelsius(double fahrenheit) {

return (fahrenheit - 32) \* 5/9;

}

public static void main(String[] args) {

TemperatureConverter converter = new TemperatureConverter();

double celsius = 25;

double fahrenheit = 77;

System.out.println(celsius + "°C = " + converter.celsiusToFahrenheit(celsius) + "°F");

System.out.println(fahrenheit + "°F = " + converter.fahrenheitToCelsius(fahrenheit) + "°C");

}

}

**Output:**

25.0°C = 77.0°F

77.0°F = 25.0°C

**Exercise 12: Movie Class**

**Problem:**

Create a class Movie with attributes title, genre, and rating. Implement a method displayMovie() to print movie details.

**Solution:**

class Movie {

private String title;

private String genre;

private double rating;

// Constructor

public Movie(String title, String genre, double rating) {

this.title = title;

this.genre = genre;

this.rating = rating;

}

// Display movie details

public void displayMovie() {

System.out.println("Movie: " + title);

System.out.println("Genre: " + genre);

System.out.println("Rating: " + rating + "/10");

}

public static void main(String[] args) {

Movie movie = new Movie("Inception", "Science Fiction", 8.8);

movie.displayMovie();

}

}

**Output:**

Movie: Inception

Genre: Science Fiction

Rating: 8.8/10

**Exercise 13: Contact Class**

**Problem:**

Create a class Contact with attributes name, phoneNumber, and email. Implement a method displayContact() to show the details.

**Solution:**

class Contact {

private String name;

private String phoneNumber;

private String email;

// Constructor

public Contact(String name, String phoneNumber, String email) {

this.name = name;

this.phoneNumber = phoneNumber;

this.email = email;

}

// Display contact details

public void displayContact() {

System.out.println("Name: " + name);

System.out.println("Phone: " + phoneNumber);

System.out.println("Email: " + email);

}

public static void main(String[] args) {

Contact contact = new Contact("Alice", "123-456-7890", "alice@example.com");

contact.displayContact();

}

}

**Output:**

Name: Alice

Phone: 123-456-7890

Email: alice@example.com

**Exercise 14: Laptop Class**

**Problem:**

Create a class Laptop with attributes brand, processor, and price. Implement a method displayLaptop() to print details.

**Solution:**

class Laptop {

private String brand;

private String processor;

private double price;

// Constructor

public Laptop(String brand, String processor, double price) {

this.brand = brand;

this.processor = processor;

this.price = price;

}

// Display laptop details

public void displayLaptop() {

System.out.println("Brand: " + brand);

System.out.println("Processor: " + processor);

System.out.println("Price: $" + price);

}

public static void main(String[] args) {

Laptop laptop = new Laptop("Dell", "Intel i7", 1200.50);

laptop.displayLaptop();

}

}

**Output:**

Brand: Dell

Processor: Intel i7

Price: $1200.5

**Exercise 15: Student Class**

**Question:**

Create a Student class with the following **fields**:

* id (int)
* name (String)
* age (int)
* gender (String)
* email (String)
* phone (String)
* address (String)
* department (String)
* cgpa (double)
* yearOfStudy (int)

Implement:

1. A **constructor** to initialize all fields.
2. **Getter and setter methods** for each field.
3. **Override toString() method** to return all details of the student.
4. Write a **separate main class** to create a Student object and print the student details.

**Solution:**

class Student {

private int id;

private String name;

private int age;

private String gender;

private String email;

private String phone;

private String address;

private String department;

private double cgpa;

private int yearOfStudy;

// Constructor

public Student(int id, String name, int age, String gender, String email,

String phone, String address, String department, double cgpa, int yearOfStudy) {

this.id = id;

this.name = name;

this.age = age;

this.gender = gender;

this.email = email;

this.phone = phone;

this.address = address;

this.department = department;

this.cgpa = cgpa;

this.yearOfStudy = yearOfStudy;

}

// Getters and Setters

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public int getAge() { return age; }

public void setAge(int age) { this.age = age; }

public String getGender() { return gender; }

public void setGender(String gender) { this.gender = gender; }

public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }

public String getPhone() { return phone; }

public void setPhone(String phone) { this.phone = phone; }

public String getAddress() { return address; }

public void setAddress(String address) { this.address = address; }

public String getDepartment() { return department; }

public void setDepartment(String department) { this.department = department; }

public double getCgpa() { return cgpa; }

public void setCgpa(double cgpa) { this.cgpa = cgpa; }

public int getYearOfStudy() { return yearOfStudy; }

public void setYearOfStudy(int yearOfStudy) { this.yearOfStudy = yearOfStudy; }

// Overriding toString()

@Override

public String toString() {

return "Student{" +

"id=" + id +

", name='" + name + '\'' +

", age=" + age +

", gender='" + gender + '\'' +

", email='" + email + '\'' +

", phone='" + phone + '\'' +

", address='" + address + '\'' +

", department='" + department + '\'' +

", cgpa=" + cgpa +

", yearOfStudy=" + yearOfStudy +

'}';

}

}

// Separate Main Class

class StudentMain {

public static void main(String[] args) {

Student student = new Student(101, "Alice Johnson", 20, "Female", "alice@example.com",

"123-456-7890", "123 Elm Street", "Computer Science", 3.8, 2);

System.out.println(student.toString()); // Using overridden toString()

}

}

**Exercise 16: Car Class**

**Question:**

Create a Car class with the following **fields**:

* brand (String)
* model (String)
* year (int)
* color (String)
* engineCapacity (double)
* fuelType (String)
* mileage (double)
* transmission (String)
* seatingCapacity (int)
* price (double)

Implement:

1. A **constructor** to initialize all fields.
2. **Getter and setter methods** for each field.
3. **Override toString() method** to return all details of the car.
4. Write a **separate main class** to create a Car object and print the car details.

**Solution:**

class Car {

private String brand;

private String model;

private int year;

private String color;

private double engineCapacity;

private String fuelType;

private double mileage;

private String transmission;

private int seatingCapacity;

private double price;

// Constructor

public Car(String brand, String model, int year, String color, double engineCapacity,

String fuelType, double mileage, String transmission, int seatingCapacity, double price) {

this.brand = brand;

this.model = model;

this.year = year;

this.color = color;

this.engineCapacity = engineCapacity;

this.fuelType = fuelType;

this.mileage = mileage;

this.transmission = transmission;

this.seatingCapacity = seatingCapacity;

this.price = price;

}

// Getters and Setters

public String getBrand() { return brand; }

public void setBrand(String brand) { this.brand = brand; }

public String getModel() { return model; }

public void setModel(String model) { this.model = model; }

public int getYear() { return year; }

public void setYear(int year) { this.year = year; }

public String getColor() { return color; }

public void setColor(String color) { this.color = color; }

public double getEngineCapacity() { return engineCapacity; }

public void setEngineCapacity(double engineCapacity) { this.engineCapacity = engineCapacity; }

public String getFuelType() { return fuelType; }

public void setFuelType(String fuelType) { this.fuelType = fuelType; }

public double getMileage() { return mileage; }

public void setMileage(double mileage) { this.mileage = mileage; }

public String getTransmission() { return transmission; }

public void setTransmission(String transmission) { this.transmission = transmission; }

public int getSeatingCapacity() { return seatingCapacity; }

public void setSeatingCapacity(int seatingCapacity) { this.seatingCapacity = seatingCapacity; }

public double getPrice() { return price; }

public void setPrice(double price) { this.price = price; }

// Overriding toString()

@Override

public String toString() {

return "Car{" +

"brand='" + brand + '\'' +

", model='" + model + '\'' +

", year=" + year +

", color='" + color + '\'' +

", engineCapacity=" + engineCapacity +

", fuelType='" + fuelType + '\'' +

", mileage=" + mileage +

", transmission='" + transmission + '\'' +

", seatingCapacity=" + seatingCapacity +

", price=" + price +

'}';

}

}

// Separate Main Class

class CarMain {

public static void main(String[] args) {

Car car = new Car("Tesla", "Model S", 2023, "Red", 0.0, "Electric",

400, "Automatic", 5, 79999.99);

System.out.println(car.toString()); // Using overridden toString()

}

}

**Exercise 17: Employee Class**

**Question:**

Create an Employee class with the following **fields**:

* id (int)
* name (String)
* age (int)
* gender (String)
* email (String)
* phone (String)
* address (String)
* designation (String)
* department (String)
* salary (double)
* dateOfJoining (String)
* experience (int)

Implement:

1. A **constructor** to initialize all fields.
2. **Getter and setter methods** for each field.
3. **Override toString() method** to return all details of the employee.
4. Write a **separate main class** to create an Employee object and print the employee details.

**Solution:**

public class Employee {

private int id;

private String name;

private int age;

private String gender;

private String email;

private String phone;

private String address;

private String designation;

private String department;

private double salary;

private String dateOfJoining;

private int experience;

// Constructor

public Employee(int id, String name, int age, String gender, String email, String phone,

String address, String designation, String department, double salary,

String dateOfJoining, int experience) {

this.id = id;

this.name = name;

this.age = age;

this.gender = gender;

this.email = email;

this.phone = phone;

this.address = address;

this.designation = designation;

this.department = department;

this.salary = salary;

this.dateOfJoining = dateOfJoining;

this.experience = experience;

}

// Getters and Setters (similar to above examples)

// Overriding toString()

@Override

public String toString() {

return "Employee{" +

"id=" + id +

", name='" + name + '\'' +

", age=" + age +

", gender='" + gender + '\'' +

", email='" + email + '\'' +

", phone='" + phone + '\'' +

", address='" + address + '\'' +

", designation='" + designation + '\'' +

", department='" + department + '\'' +

", salary=" + salary +

", dateOfJoining='" + dateOfJoining + '\'' +

", experience=" + experience +

'}';

}

}

// Separate Main Class

class EmployeeMain {

public static void main(String[] args) {

Employee emp = new Employee(501, "John Smith", 35, "Male", "john.smith@example.com",

"987-654-3210", "456 Oak Avenue", "Software Engineer", "IT",

75000.00, "2015-06-10", 8);

System.out.println(emp.toString()); // Using overridden toString()

}

}