TEST – 3 (SET -2)

1. Create a base Employee class with a method getSalary(). Implement subclasses Manager and Developer, each overriding the getSalary() method to include their specific bonus structure. Additionally, overload the getSalary method in each subclass to include the possibility of receiving a bonus as an argument.

PROGRAM:

```
class Employee {
    public double getSalary() {
        return 50000.0;
    }
}

class Manager extends Employee {
    @Override
    public double getSalary() {
        return 80000.0;
    }

    public double getSalary(double bonus) {
        return getSalary() + bonus;
    }
}

class Developer extends Employee {
    @Override
    public double getSalary() {
```

```
        return 60000.0;
    }

    public double getSalary(double bonus) {
        return getSalary() + bonus;
    }
}

public class Main {
    public static void main(String[] args) {
        Manager manager = new Manager();
        Developer developer = new Developer();

        System.out.println("Manager's Salary: " + manager.getSalary());
        System.out.println("Manager's Salary with Bonus: " +
manager.getSalary(10000.0));

        System.out.println("Developer's Salary: " + developer.getSalary());
        System.out.println("Developer's Salary with Bonus: " +
developer.getSalary(5000.0));
    }
}
```

OUTPUT :

Manager’s Salary : 80000.0

Manager’s Salary with Bonus : 90000.0

Developer’s Salary : 60000.0

Developer’s Salary with Bonus : 65000.0

3. Design a library management system using inheritance. Create a base class LibraryItem and derived classes Book, Magazine, and DVD. Each derived class should override a method getItemDetails() to provide specific details about the item.

PROGRAM:

```
class LibraryItem {
    private String title;
    private String location;
    private boolean available;

    public LibraryItem(String title, String location, boolean available) {
        this.title = title;
        this.location = location;
        this.available = available;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getLocation() {
        return location;
    }
```

```
public void setLocation(String location) {
    this.location = location;
}

public boolean isAvailable() {
    return available;
}

public void setAvailable(boolean available) {
    this.available = available;
}

public void checkOut() {
    if (available) {
        available = false;
        System.out.println(title + " has been checked out.");
    } else {
        System.out.println(title + " is currently not available.");
    }
}

public void checkIn() {
    if (!available) {
        available = true;
        System.out.println(title + " has been checked in.");
    } else {
```

```
            System.out.println(title + " is already available.");
        }
    }

    public void getItemDetails() {

    }
}

class Book extends LibraryItem {
    private String author;

    public Book(String title, String author, String location, boolean available) {
        super(title, location, available);
        this.author = author;
    }

    @Override
    public void getItemDetails() {
        System.out.println("BOOK: " + getTitle() + " by " + author + ". Location: "
+ getLocation() + ". Available: " + isAvailable());
    }
}
class Magazine extends LibraryItem {
    private String issue;

    public Magazine(String title, String issue, String location, boolean available)
{
```

```
        super(title, location, available);
        this.issue = issue;
    }

    @Override
    public void getItemDetails() {
        System.out.println("MAGAZINE: " + getTitle() + ", Issue: " + issue + ". Location: " + getLocation() + ". Available: " + isAvailable());
    }
}

class DVD extends LibraryItem {
    private String director;

    public DVD(String title, String director, String location, boolean available) {
        super(title, location, available);
        this.director = director;
    }


    @Override
    public void getItemDetails() {
        System.out.println("DVD: " + getTitle() + ", Directed by " + director + ". Location: " + getLocation() + ". Available: " + isAvailable());
    }
}

public class Main {
```

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


        Book book1 = new Book("The Catcher in the Rye", "J.D. Salinger",
"Fiction Section", true);
        Magazine magazine1 = new Magazine("National Geographic", "June
2023", "Periodicals Section", true);
        DVD dvd1 = new DVD("Inception", "Christopher Nolan", "Movies
Section", false);




        dvd1.checkOut();




        book1.getItemDetails();
        magazine1.getItemDetails();
        dvd1.getItemDetails();
    }
}
```

OUTPUT:

Inception is currently not available

BOOK : The Catcher in the Rye by J.D.Salingeer . Locatio : Fiction Section . Available : true

MAGAZINE : Geographic , Issue : June 2023 . Location : Periodicals Section . Available : true

DVD : Inception , Directed by Christopher Nolan.Location : Movies Section , Available : false