Assignment Question:

Design and implement a console-based Task Management Application that simulates how companies manage tasks. The application should be built using Object-Oriented Programming principles in Java.

Requirements:

1. Create at least 4 classes:
   * User – represents employees with details such as id, name, role, email.
   * Task – represents tasks with id, title, description, status, assigned user.
   * Project – contains multiple tasks, project name, deadline, and team members.
   * TaskManager – service class that manages users, projects, and tasks.
2. Each class must include:
   * ≥4 instance/static variables.
   * A constructor to initialize values.
   * ≥5 methods (getters/setters, business logic like assignTask(), updateStatus(), listTasks()).
3. Demonstrate OOPS Concepts:
   * Inheritance → Specialized roles (e.g., Manager extends User).
   * Method Overloading → Assign tasks with/without deadline.
   * Method Overriding → Custom display of details in subclasses.
   * Polymorphism → Handle different user types (User reference calling Manager methods).
   * Encapsulation → Keep variables private with getters/setters.
4. Write a Main class (TaskAppMain) to test all functionalities:
   * Add users and projects.
   * Create and assign tasks.
   * Update task status (e.g., TODO → IN PROGRESS → DONE).
   * Display all tasks per project.
   * Show reports (e.g., tasks completed by a user).

# User.java

# Package Assignment1;

public class User {

private int id;

private String name; private String role; private String email;

public User(int id, String name, String role, String email) { this.id = id;

this.name = name; this.role = role; this.email = email;

}

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 String getRole() { return role; }

public void setRole(String role) { this.role = role; }

public String getEmail() { return email; }

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

public void displayDetails() {

System.*out*.println("User: " + id + ", " + name + ", Role: " + role + ", Email: " + email);

}

# Manager.java

Package Assignment1;

public class Manager extends User { private int teamSize;

public Manager(int id, String name, String email, int teamSize) { super(id, name, "Manager", email);

this.teamSize = teamSize;

}

public int getTeamSize() { return teamSize; }

public void setTeamSize(int teamSize) { this.teamSize = teamSize; }

public void displayDetails() {

System.*out*.println("Manager: " + getName() + " (Team Size: " + teamSize + ")");

}

public void approveTask(Task task) {

System.*out*.println("Manager " + getName() + " approved task: " + task.getTitle());

}

}

# Task.java

# Package Assignment1;

# public class Task {

private int id; private String title;

private String description;

private String status;

private User assignedUser;

public Task(int id, String title, String description) { this.id = id;

this.title = title; this.description = description; this.status = "TODO";

}

public int getId() { return id; }

public String getTitle() { return title; }

public String getDescription() { return description; } public String getStatus() { return status; }

public User getAssignedUser() { return assignedUser; }

public void setStatus(String status) { this.status = status; } public void assignUser(User user) { this.assignedUser = user; }

public void displayTask() {

System.*out*.println("Task: " + id + ", " + title + " [" + status + "] "

+ (assignedUser != null ? " -> " + assignedUser.getName() : " (Unassigned)"));

}

}

# Project.java

# Package Assignment1;

# public class Project {

private String projectName; private String deadline; private Task[] tasks;

private User[] teamMembers; private int taskCount;

private int memberCount;

private final int MAX\_TASKS = 50; private final int MAX\_MEMBERS = 20;

public Project(String projectName, String deadline) { this.projectName = projectName;

this.deadline = deadline;

this.tasks = new Task[MAX\_TASKS]; this.teamMembers = new User[MAX\_MEMBERS]; this.taskCount = 0;

this.memberCount = 0;

}

public String getProjectName() { return projectName; } public String getDeadline() { return deadline; }

public void addTask(Task task) {

if (taskCount < MAX\_TASKS) { tasks[taskCount++] = task;

} else {

System.*out*.println("Task limit reached for project: " + projectName);

}

}

public void addTeamMember(User user) { if (memberCount < MAX\_MEMBERS) {

teamMembers[memberCount++] = user;

} else {

System.*out*.println("Team member limit reached for project: " + projectName);

}

}

public void listTasks() {

System.*out*.println("Tasks in Project: " + projectName); for (int i = 0; i < taskCount; i++) {

tasks[i].displayTask();

}

}

public void showTeam() {

System.*out*.println("Team Members in " + projectName + ":"); for (int i = 0; i < memberCount; i++) {

teamMembers[i].displayDetails();

}

}

public Task[] getTasks() { return tasks; } public int getTaskCount() { return taskCount; }

}

# TaskManager.java

# Package Assignment1;

public class TaskManager {

private User[] users = new User[50]; private Project[] projects = new Project[20]; private int userCount = 0;

private int projectCount = 0;

public void addUser(User user) { if (userCount < users.length) {

users[userCount++] = user;

}

}

public void addProject(Project project) { if (projectCount < projects.length) {

projects[projectCount++] = project;

}

}

public void assignTask(Task task, User user) { task.assignUser(user);

System.*out*.println("Task '" + task.getTitle() + "' assigned to " + user.getName());

}

public void assignTask(Task task, User user, String deadline) {

task.assignUser(user);

System.*out*.println("Task '" + task.getTitle() + "' assigned to " + user.getName()

+ " with deadline " + deadline);

}

public void updateTaskStatus(Task task, String status) { task.setStatus(status);

System.*out*.println("Task '" + task.getTitle() + "' updated to " + status);

}

public void showUserReport(User user) { System.*out*.println("Report for " + user.getName() + ":"); for (int i = 0; i < projectCount; i++) {

Project p = projects[i];

for (int j = 0; j < p.getTaskCount(); j++) { Task t = p.getTasks()[j];

if (t.getAssignedUser() != null && t.getAssignedUser().getId() == user.getId()) { System.*out*.println(" - " + t.getTitle() + " [" + t.getStatus() + "]");

}

}

}

}

public void listAllProjects() {

for (int i = 0; i < projectCount; i++) { Project p = projects[i];

System.*out*.println("Project: " + p.getProjectName() + " (Deadline: " + p.getDeadline() + ")"); p.listTasks();

}

}

}

# TaskMain.java

package Assignment;

public class TaskMain {

public static void main(String[] args) { TaskManager manager = new TaskManager();

User u1 = new User(1, "Alice", "Developer", "[alice@company.com](mailto:alice@company.com)"); User u2 = new User(2, "Bob", "Tester", "[bob@company.com](mailto:bob@company.com)"); Manager m1 = new Manager(3, "Charlie", "[charlie@company.com](mailto:charlie@company.com)", 5);

manager.addUser(u1); manager.addUser(u2); manager.addUser(m1);

Project p1 = new Project("Website Revamp", "2025-12-31"); p1.addTeamMember(u1);

p1.addTeamMember(u2); p1.addTeamMember(m1);

Task t1 = new Task(101, "Design Homepage", "Create modern homepage layout"); Task t2 = new Task(102, "Test Login", "Ensure login works properly");

p1.addTask(t1); p1.addTask(t2);

manager.addProject(p1); manager.assignTask(t1, u1);

manager.assignTask(t2, u2, "2025-10-15");

manager.updateTaskStatus(t1, "IN PROGRESS"); manager.updateTaskStatus(t1, "DONE");

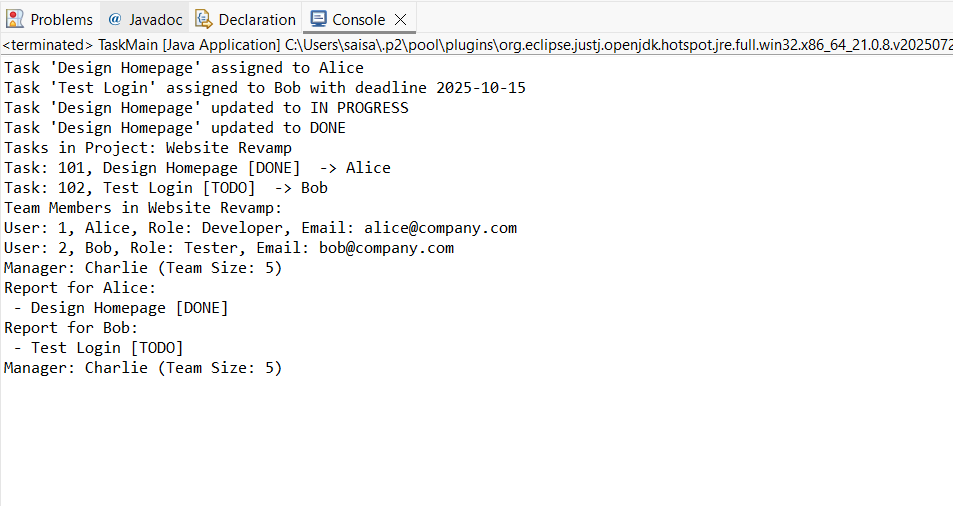
p1.listTasks(); p1.showTeam(); manager.showUserReport(u1); manager.showUserReport(u2);

User ref = m1; ref.displayDetails();

}

}

**Output:**

****

# GitHub Repository Link: