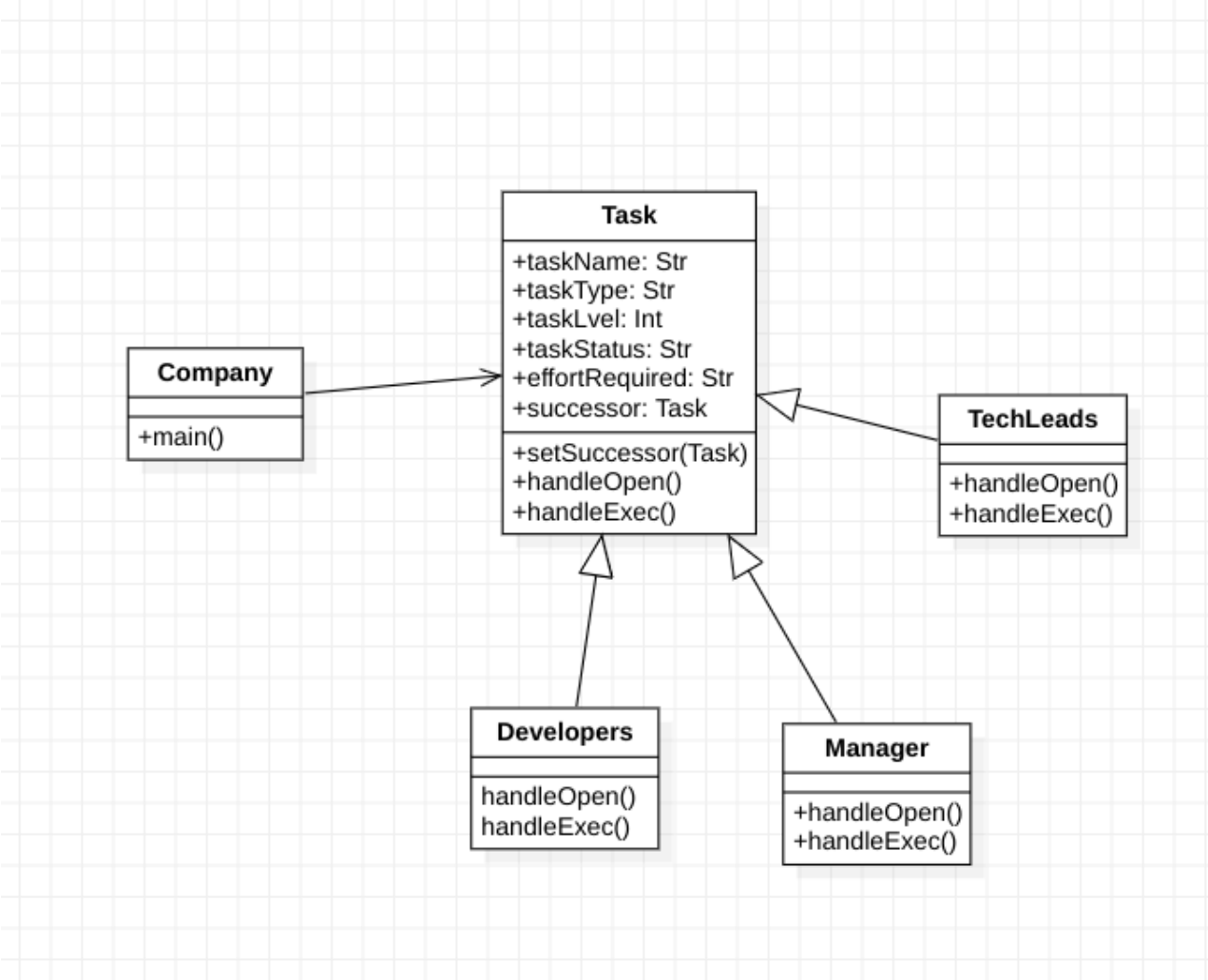
Name: Arvind Krishna	SRN:PES1UG19CS090	Sem: 6, Sec: B

CLASS DIAGRAM:



Design pattern information:

//specify the design patterns used – Specify the need and structure of DP used

Output Screen Shot:

```
arookrishna@whats-poppin:~/Documents$ java Company
Level- 1
Task- Job 1 opened by Tech Lead!
Task- Job 1 executed by Developer!


Level- 2
Task- Job 2 opened by Manager!
Task- Job 2 executed by TechLead!


Level- 3
Task- Job 3 opened by Manager!
Task- Job 3 executed by Manager!
```

Code:

TechLead.java

```
public class TechLead extends Task
{
   //  constructor with taskName, taskType, taskStatus, taskLevel,
taskSize, effortRequired.
   TechLead(String TaskName, String TaskType, String TaskStatus, int
TaskLevel, String TaskSize, String EffortRequired)
   {
       super(TaskName, TaskType, TaskStatus, TaskLevel, TaskSize,
EffortRequired);
   }


   public void handleOpen() {
       if(super.taskLevel==  1) {
           System.out.println("Task- "+ super.taskName + " opened by Tech
Lead!");
       }
```

```
        else if (successor != null) {
            successor.handleOpen();
        }
    }

    public void handleExec() {
        if(super.taskLevel==2) {
            System.out.println("Task- " + super.taskName + " executed by
TechLead!");
        }

        else if (successor != null) {
            successor.handleExec();
        }
    }
}
```

Task.java

```
public abstract class Task
{
    String taskName;
    String taskType;
    String taskStatus;
    int taskLevel;
    String taskSize;
    String effortRequired;
    protected Task successor;

    Task(String taskName, String taskType, String taskStatus, int taskLevel,
String taskSize, String effortRequired)
    {
        this.taskName = taskName;
        this.taskType = taskType;
        this.taskStatus = taskStatus;
        this.taskLevel = taskLevel;
        this.taskSize = taskSize;
        this.effortRequired = effortRequired;
```

```
    }

    public void setSuccessor(Task successor) {
        this.successor = successor;
    }

    public abstract void handleOpen();
    public abstract void handleExec();
}
```

Manager.java

```
public class Manager extends Task
{
    public Manager(String TaskName, String TaskType, String TaskStatus, int
TaskLevel, String TaskSize, String EffortRequired)
    {
        super(TaskName, TaskType, TaskStatus, TaskLevel, TaskSize,
EffortRequired);
    }

    public void handleOpen() {
        if( super.taskLevel==  2 || super.taskLevel==3) {
            System.out.println("Task- "+ super.taskName + " opened by
Manager!");
        }

        else if (successor != null) {
            successor.handleOpen();
        }
    }

    public void handleExec() {
        if(super.taskLevel==3){
            System.out.println("Task- " + super.taskName + " executed by
Manager!");
        }
```

```
        else if (successor != null) {
            successor.handleExec();
        }
    }
}
```

Developer.java

```
public class Developer extends Task{
    public Developer(String TaskName, String TaskType, String TaskStatus,
int TaskLevel, String TaskSize, String EffortRequired)
    {
        super(TaskName, TaskType, TaskStatus, TaskLevel, TaskSize,
EffortRequired);
    }

    public void handleOpen(){
        successor.handleOpen();
    }

    public void handleExec(){
        if(super.taskLevel==1){
            System.out.println("Task- " + super.taskName + " executed by
Developer!");
        }

        else if (successor != null){
            successor.handleExec();
        }
    }
}
```

Company.java

```
class Company{
    //  create task with taskName, taskType, taskStatus, taskLevel,
```

```
taskSize, effortRequired.
   // public static createTask(String taskName, String taskType, String
taskStatus, int taskLevel, String taskSize, String effortRequired){
   //      //  create task object
   //      Task task = new Task(taskName, taskType, taskStatus, taskLevel,
taskSize, effortRequired);
   //      //  add task to taskList
   //      taskList.add(task);
   // }

   public static void main(String[] args){

        String[] work=new String[3];
        work[0]="Job 1";
        work[1]="Job 2";
        work[2]="Job 3";

        for(int i=1;i<=3;i++)
        {
            System.out.println("Level- "+ i);
            Developer developer = new Developer(work[i-1],"placeholder",
"placeholder",i,"placeholder","placeholder");
            TechLead tech = new TechLead(work[i-1],"placeholder",
"placeholder",i,"placeholder","placeholder");
            Manager manager = new Manager(work[i-1],"placeholder",
"placeholder",i,"placeholder","placeholder");

            manager.setSuccessor(tech);
            tech.setSuccessor(developer);

            manager.handleOpen();
            manager.handleExec();

            System.out.println("\n");
        }
   }
}
```