NAME :MARIYAM MAHNOOR
COURSE: BLOCKCHAIN PROGRAMMING
SECTION: "B"
DATE: 7-DEC-2020

PROJECT MANAGEMENT TOOL with HERITANCE:

```
// define assignee class
class Assignee{
    constructor(name ,title,status){
        this.name= name
        this.title= title
        this.status= status
        this.Assigneepay='50000pkr'
    }
    ChangeStatus(value){
        this.status= value
        return this.status
    }
    AssigneePay(){
        return this.Assigneepay
    }
    AssigneeDetails(){
        console.log("name :"+this.name)
        console.log("title :"+this.title)
        console.log("status "+this.status)

    }

}
// creating objects with assignee class
var hamza= new Assignee('Hamza khan','finance Manager','active')
var musab= new Assignee('Musab khan','machanical Engineer','active')
var Ali= new Assignee('ali khan','marketing Manager','active')
var Mustafain = new Assignee('Mustafain khan','web designer','fresher')
var owais= new Assignee('Owais khan',' graphic Designer','active')
console.log(hamza.AssigneePay())
console.log(owais)
owais.ChangeStatus("internee")
console.log(owais)

// define tasks class
class tasks extends Assignee{
    constructor(name,tdays,AssigneeId,taskstatus){
        super();
        this.tname = name
        this.tdays = tdays
```

```
        this.AssigneeId= AssigneeId
        this.taskstatus= taskstatus
    }
    ChangeTaskStatus(value){
        this.taskstatus= value
    }

   salary(){
       console.log("salary")
        return super.AssigneePay()
    }
    taskPercentage(tadays,compdays){
        var per = tadays/compdays * 100;
        return per;
    }
}

// creating objects with tasks class
var task1 = new tasks('architecture',5,hamza,'inprogress')
var task2 = new tasks('development_1',10,Ali,'notstarted')
var task3 = new tasks('development_2',4,musab,'done')
var task4 = new tasks('development_3',15,owais,'inprogress')
console.log(task1.salary())

console.log(owais)
// creating map for multiple tasks
var blockchain_task = new Map();
blockchain_task.set('1',task1);
blockchain_task.set('2',task2);
blockchain_task.set('3',task3);
blockchain_task.set('4',task4);


// define project  class
class project extends tasks{
    constructor(name,type,days,tasks,dayscompleted){
        super();
        this.tasks = tasks
        this.status="in progress";
        this.days = days
        this.name=name
        this.type = type
        this.dayscompleted = dayscompleted

    }
    projectDetails(){
        console.log(this.name)
        console.log(this.type)
        console.log(this.days)
        console.log("days completed "+this.dayscompleted)
        console.log(super.salary())
    }
    getTaskPercentage(taskid){
        var task = this.tasks.get(taskid);
        var taskDays = task.tdays;
        var per  = super.taskPercentage(taskDays,this.dayscompleted);
        return 'The Percentage of ' + task.tname + ' is : ' + per;
```

```
    }
    print(){
        return this.tasks
    }
    totaltasks(){
        return this.tasks.size
    }
    daysRemaining(){
        var remainingday=this.dayscompleted-this.days
        return remainingday


    }
    status_(){
        var i=0;
        var j=0;
        var k=0;
        for (let [key, value] of this.tasks.entries()) {


            if(value.taskstatus =="done"){

                console.log(key + ' = ' + value.taskstatus)
                i+=1;

            }

            else if(value.taskstatus =="inprogress"){

                console.log(key + ' = ' + value.taskstatus)
                j+=1;

            }
            else{

                console.log(key + ' = ' + value.taskstatus)
                k+=1;

            }


          }
          console.log(`Completed task ${i}`)
          console.log(`Inprogress task ${j}`)
          console.log(`Not started task ${k}`)
          if(i==this.tasks.size){
              console.log("project completed")

          }
          else if(i < this.tasks.size){
            console.log("Project  is inprogress")

        }
        else if(k == this.tasks.size){
            console.log("project not started")

        }
```

```
        else{
            console.log("not define")
        }

    }
    CompletedTasks(){
        console.log("Completed task")
        for (let [key, value] of this.tasks.entries()) {
            if(value.taskstatus =="done"){

                console.log(key + ' = ' + value.taskstatus)
            }

          }
          console.log('\n---------------')


    }
    PendingTasks(){
        console.log("Pending task")
        for (let [key, value] of this.tasks.entries()) {

            if(value.taskstatus!="done"){

                console.log(key + ' = ' + value.taskstatus)
            }

          }
          console.log('\n---------------')


    }
    TaskAssigneee(taskid) {
        var task = this.tasks.get(taskid);
        var taskSAssignee = task.AssigneeId;
        taskSAssignee.AssigneeDetails()
        return 'The above  Detail of Assignee of this task  ' + task.tname + '  : ' ;

    }
    changeTask(taskid,task){
        this.tasks.set(taskid,task)
        console.log(this.tasks.get(taskid,task))
    }
    DeleteTask(value){
        this.tasks.delete(value)

    }
    AddTask(key,value){
    this.tasks.set(key,value)

    }
    ChangeAssignee(taskid,value){
        this.tasks.get(taskid).AssigneeId= value;


    }
}
```

```
// creating a project object with project class

var realstate_blockchain = new project('blockchain','software',15,blockchain_task,'50')
realstate_blockchain.projectDetails()
console.log(realstate_blockchain.getTaskPercentage('2'))
console.log(realstate_blockchain.TaskAssigneee('1'))
// console.log(realstate_blockchain.print())
// console.log(realstate_blockchain.totaltasks())
// console.log("change task")
// realstate_blockchain.changeTask('1',task2)
// realstate_blockchain.changeTask('2',task1)
// console.log(realstate_blockchain.print())
// realstate_blockchain.CompletedTasks()
// realstate_blockchain.PendingTasks()
// console.log(realstate_blockchain.daysRemaining())
// realstate_blockchain.status_()
// realstate_blockchain.ChangeAssignee('1',Mustafain)
// console.log(realstate_blockchain.print())
```

Output:

Assignee Detail
Assignee {
  name: 'Owais khan',
  title: ' graphic Designer',
  status: 'active',
  Assigneepay: '50000pkr'
}
Name :blockchain
Type: software
completed Days 15
total Days 50
 Assignee salary
50000pkr
The Percentage of development_1 is : 20
name :Hamza khan
title :finance Manager
status active

The above Detail of Assignee of this task architecture :