**TYPESCRIPT TASK**

**Step 1:**

**A screenshot of a computer program

Description automatically generated**

**Step 2:**

**A screen shot of a computer

Description automatically generated**

**Step 3:**

**A black background with green text

Description automatically generated**

**Step 4:**

**A screen shot of a computer code

Description automatically generated**

**Step 5:**

**A screen shot of a computer program

Description automatically generated**

**Step 6:**

**A screen shot of a computer program

Description automatically generated**

**Step 7:**

**A screen shot of a computer code

Description automatically generated**

**Step 8 & 9:**

****

**Step 10:**

class Employee {

    constructor(private currentProject: string, private name: string) { }

    getCurrentProject(): string {

        return this.currentProject;

    }

    getName(): string {

        return this.name;

    }

}

class Company {

    private employees: Employee[] = [];

    add(employee: Employee) {

        this.employees.push(employee);

    }

    getProjectList(): string[] {

        const projectList: string[] = new Array();

        this.employees.map(employee=>projectList.push(employee.getCurrentProject()));

        return projectList;

    }

    getNameList(): string[] {

        const namesList: string[] = new Array();

        this.employees.map(employee=>namesList.push(employee.getName()));

        return namesList;

    }

}

class Frontend extends Employee {

}

class Backend extends Employee {

}

const company: Company = new Company();

const frontend: Frontend = new Employee("Java", "Supriya");

const backend: Backend = new Employee("Python", "Pavan");

company.add(frontend);

company.add(backend);

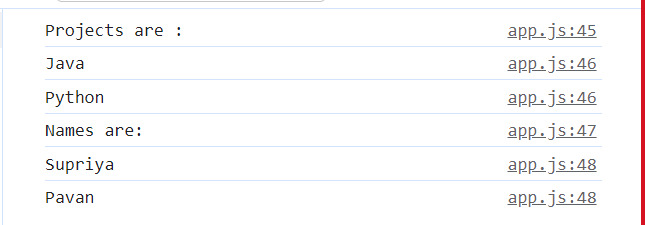
console.log("Projects are:")

company.getProjectList().forEach(project => console.log(project));

console.log("Names are:")

company.getNameList().forEach(name => console.log(name));

**Output:**

****

**Step 11:**

interface IEmployee{

    currentProject: string;

    name: string

    getCurrentProject(): string;

    getName(): string

}

class Company {

    private employees: IEmployee[] = [];

    add(employee: IEmployee) {

        this.employees.push(employee);

    }

    getProjectList(): string[] {

        const projectList: string[] = new Array();

        this.employees.map(employee=>projectList.push(employee.getCurrentProject()));

        return projectList;

    }

    getNameList(): string[] {

        const namesList: string[] = new Array();

        this.employees.map(employee=>namesList.push(employee.getName()));

        return namesList;

    }

}

class Frontend implements IEmployee {

    currentProject: string;

    name: string;

    constructor(currentProject:string,name:string){

        this.currentProject=currentProject;

        this.name=name;

    }

    getCurrentProject(): string {

        return this.currentProject;

    }

    getName(): string {

        return this.name;

    }

}

class Backend implements IEmployee {

    currentProject: string;

    name: string;

    constructor(currentProject:string,name:string){

        this.currentProject=currentProject;

        this.name=name;

    }

    getCurrentProject(): string {

        return this.currentProject;

    }

    getName(): string {

        return this.name;

    }

}

const company: Company = new Company();

const frontend: Frontend = new Frontend("Java", "Supriya");

const backend: Backend = new Backend("Python", "Pavan");

company.add(frontend);

company.add(backend);

console.log("Projects are:")

company.getProjectList().forEach(project => console.log(project));

console.log("Names are:")

company.getNameList().forEach(name => console.log(name));

**Output:**

**A screenshot of a computer

Description automatically generated**

**Step 12:**

class Employee {

    constructor(private currentProject: string, private name: string) { }

    getCurrentProject(): string {

        return this.currentProject;

    }

    getName(): string {

        return this.name;

    }

}

interface ILocation{

    addPerson(employee: Employee):void;

    getPerson(index:number): Employee;

    getCount():number;

}

class Location1{

    constructor(private locationName:string){}

    getLocationName():string{

        return this.locationName;

    }

}

class CompanyLocationArray implements ILocation{

    private employees: Employee[] = [];

    addPerson(employee: Employee): void {

        this.employees.push(employee);

    }

    getPerson(index: number): Employee {

       if(index<this.getCount()){

        return this.employees[index];

       }else{

        throw new Error("Employee Not Found");

       }

    }

    getCount(): number {

        return this.employees.length;

    }

}

class CompanyLocationLocalStorage implements ILocation{

    private employees: Employee[] = [];

    addPerson(employee: Employee): void {

        this.employees.push(employee);

    }

    getPerson(index: number): Employee {

       if(index<this.getCount()){

        return this.employees[index];

       }else{

        throw new Error("Employee Not Found");

       }

    }

    getCount(): number {

        return this.employees.length;

    }

}

class Company {

    private employees: Employee[] = [];

    constructor(location: Location1){}

    add(employee: Employee) {

        this.employees.push(employee);

    }

    getProjectList(): string[] {

        const projectList: string[] = new Array();

        this.employees.map(employee=>projectList.push(employee.getCurrentProject()));

        return projectList;

    }

    getNameList(): string[] {

        const namesList: string[] = new Array();

        this.employees.map(employee=>namesList.push(employee.getName()));

        return namesList;

    }

}

const hyderabadCompany: Company = new Company(new Location1("Hyderabad"));

const chennaiCompany: Company = new Company(new Location1("Chennai"));

hyderabadCompany.add(new Employee("Java","Supriya"));

hyderabadCompany.add(new Employee("Python","Pavan"))

chennaiCompany.add(new Employee("Adobe","Keerthi"));

chennaiCompany.add(new Employee("ASE","Ramya"));

console.log("Hyderabad Project List :");

hyderabadCompany.getProjectList().forEach(project => console.log(project));

console.log("Chennai Project List :");

chennaiCompany.getProjectList().forEach(project => console.log(project));

console.log("Hyderabad Company Name List :");

hyderabadCompany.getNameList().forEach(name => console.log(name));

console.log("Chennai Company Name List :");

chennaiCompany.getNameList().forEach(name => console.log(name));

**Output:A screenshot of a computer

Description automatically generated**