```
import java.util.ArrayList;
import java.util.List;
import java.util.ListIterator;

public abstract class AdministrativeUnit {
    protected int indentDepth = 0;
    protected String unitName;
    protected AdministrativeUnit parentUnit;
    protected List<AdministrativeUnit> units;
    protected List<String> idTags;

    public AdministrativeUnit(String unitName) {
        this.unitName = unitName;
        this.units = new ArrayList<>();
        this.idTags = new ArrayList<>();
    }

    public abstract void printStaffingInformation();

    public void addUnit(AdministrativeUnit administrativeUnit) {
        units.add(administrativeUnit);
    }

    public final boolean isComposite() {
        return !units.isEmpty();
    }

    public final ListIterator<AdministrativeUnit> listUnits() {
        return units.listIterator();
    }

    protected String indent(int depth) {
        StringBuilder stringBuilder = new StringBuilder();

        do {
            stringBuilder.append("+ ");
        } while (depth-- > 0);

        return stringBuilder.toString();
    }

    public final void printStructure() {
        System.out.println(indent(indentDepth) + unitName +
                           (isComposite() ? " (node)" : " (leaf)"));
        for (AdministrativeUnit area : units) {
            area.indentDepth = indentDepth + 1;
            area.printStructure();
        }
    }
```

```
    public String getSuperiorUnit() {
        String parent;

        if (parentUnit == null)
            parent = "--- top level unit";
        else
            parent = this.parentUnit.unitName;

        return parent;
    }

    public void addIdTag(String idTag){
        idTags.add(idTag);
    }

    public void printTags (){
        System.out.print("[" + idTags + "]");
        if (parentUnit != null)
            parentUnit.printTags();
    }
}

public class Office extends AdministrativeUnit {
    public Office(String areaName) {
        super(areaName);
    }

    public void printStaffingInformation() {
        StringBuilder stringBuilder = new StringBuilder();

        stringBuilder.append("Unit name: ").append(this.unitName)
                     design.append("\n");
        stringBuilder.append("Superior unit: ").append(getSuperiorUnit());

        for (AdministrativeUnit administrativeUnit : units)
            administrativeUnit.printStaffingInformation();

        System.out.println(stringBuilder.toString());
    }
}

public class AdministrativeArea extends AdministrativeUnit {
    public AdministrativeArea(String areaName) {
        super(areaName);
    }

    public void printStaffingInformation() {
        StringBuilder stringBuilder = new StringBuilder();

        stringBuilder.append("unit name: ").append(unitName).append("\n");
        stringBuilder.append("superior unit: ").append(getSuperiorUnit());

        for (AdministrativeUnit area : units)
            area.printStaffingInformation();

        System.out.println(stringBuilder.toString());
    }
}
```

```
public class Application {
    public static void main(String... args) {
        AdministrativeUnit a1 = new AdministrativeArea("a1");
        a1.addIdTag("a1");

        AdministrativeUnit a1b1 = new AdministrativeArea("a1b1");
        a1b1.addIdTag("a1b1");

        AdministrativeUnit a1b1c1 = new AdministrativeArea("a1b1c1");
        a1b1c1.addIdTag("a1b1c1");
        AdministrativeUnit o1 = new Office("o1");

        AdministrativeUnit a1b1c1d1 = new AdministrativeArea("a1b1c1d1");
        a1b1c1d1.addIdTag("a1b1c1d1");
        AdministrativeUnit o2 = new Office("o2");
        AdministrativeUnit o3 = new Office("o3");

        a1b1c1d1.addUnit(o2);
        a1b1c1d1.addUnit(o3);
        a1b1c1.addUnit(a1b1c1d1);
        a1b1c1.addUnit(o1);
        a1b1.addUnit(a1b1c1);
        a1.addUnit(a1b1);

        a1.printStructure();
        System.out.println();
        a1.printStaffingInformation();
    }
}
```