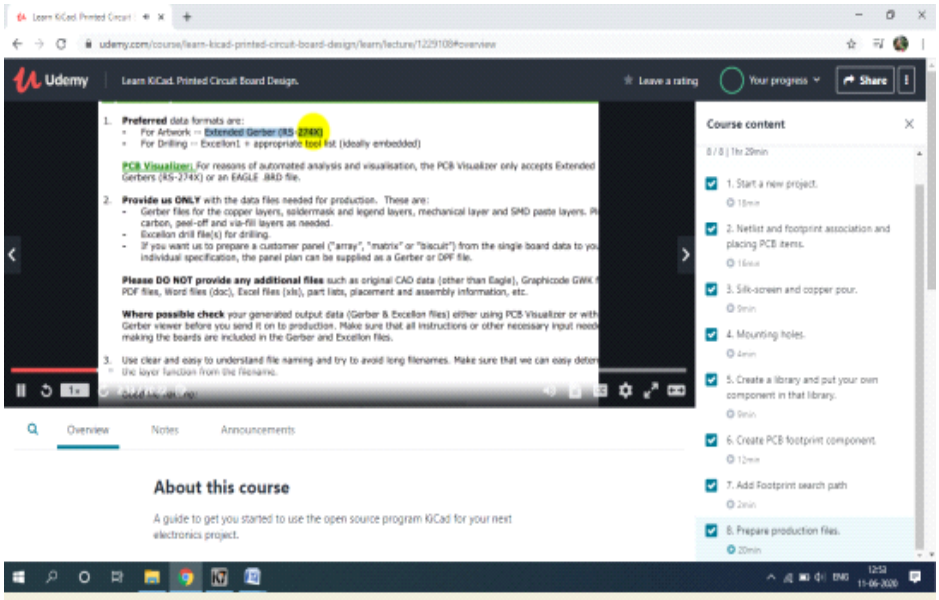
DAILY ASSESSMENT FORMAT

Date:	12-06-2020	Name:	POOJA K S
Course:	PCB design	USN:	4AL17EC070
Topic:	Add footprint search path, prepare production files	Semester & Section:	6th SEM 'B' SEC
Github Repository:	pooja-shivanna		

FORENOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.

Add footprint search path:When creating a custom PCB footprint for a component, it is stored somewhere on your computer. In order for Design Entry CIS to find where a custom footprint is stored and associate it with a schematic component, the *library search path* must be changed so that Design Entry CIS knows where to look. Save your custom

footprints in the symbols folder on your computer. Depending on how Cadence is installed on your computer, the full path should be similar to:
C:\Program Files\Cadence\SPB_17.2\share\pcb\pcb_lib\symbols
When creating a new footprint drawing, the New Drawing dialog box will show the default pathLaunch Design Entry CIS. Note the full path for the Capture.ini file shown on the Start Page (see Figure 2). Depending on how Cadence is installed on your computer, the full path should be similar to:C:\Cadence\SPB_Data-Silent\cdssetup\OrCAD_Capture\17.2.0\Capture.ini
or, if you made a custom HOME variable:
%HOME%\cdssetup\OrCAD_Capture\17.2.0\Capture.iniThe Capture.ini file will open in Notepad. Under the [Allegro Footprints] section, add the full library search path from step 1 above if it is not already listed (see Figure 4). Note that you must increment the number after Dir for each path added (e.g., Dir0, Dir1, Dir2). Do not delete any existing paths from the list. You have successfully added a library search path to Design Entry CIS. If you are still not able to attach your custom footprints to schematic symbols, re-check the above steps and make sure your custom footprint name is correct.

What is a Gerber file?

The most widely used file format for PCB manufacturing is called Gerber. When manufacturers request “Gerbers” or “Gerber files,” they are referring to ASCII files that contain Gerber- formatted data. A Gerber file knows nothing about design rules, net connectivity, or component libraries; it is simply two-dimensional artwork that indicates where the manufacturing equipment will place copper, solder mask, or silkscreen. One Gerber file provides information for one PCB feature on one layer. Thus, if you have a two-layer board and each side has copper, solder mask, and silkscreen, you will need six Gerber files. You may also need a separate Gerber file to identify the board outline. Generating Gerber files can be somewhat complicated. The process involves various configuration details, and different manufacturers have different requirements. The following screen capture shows the options that you have to consider when generating Gerber files with DipTrac If you don’t have much experience with Gerber generation, I suggest the following approach: First, choose a manufacturer that provides specific instructions on how to generate Gerber files with specific CAD tools. Second, use one of these CAD tools to design your board. If you follow the instructions carefully, you will almost certainly avoid the two potential consequences of improper. Gerber files: a delay in the manufacturing process (more likely), or a nonfunctional PCB (nowadays probably quite rare).

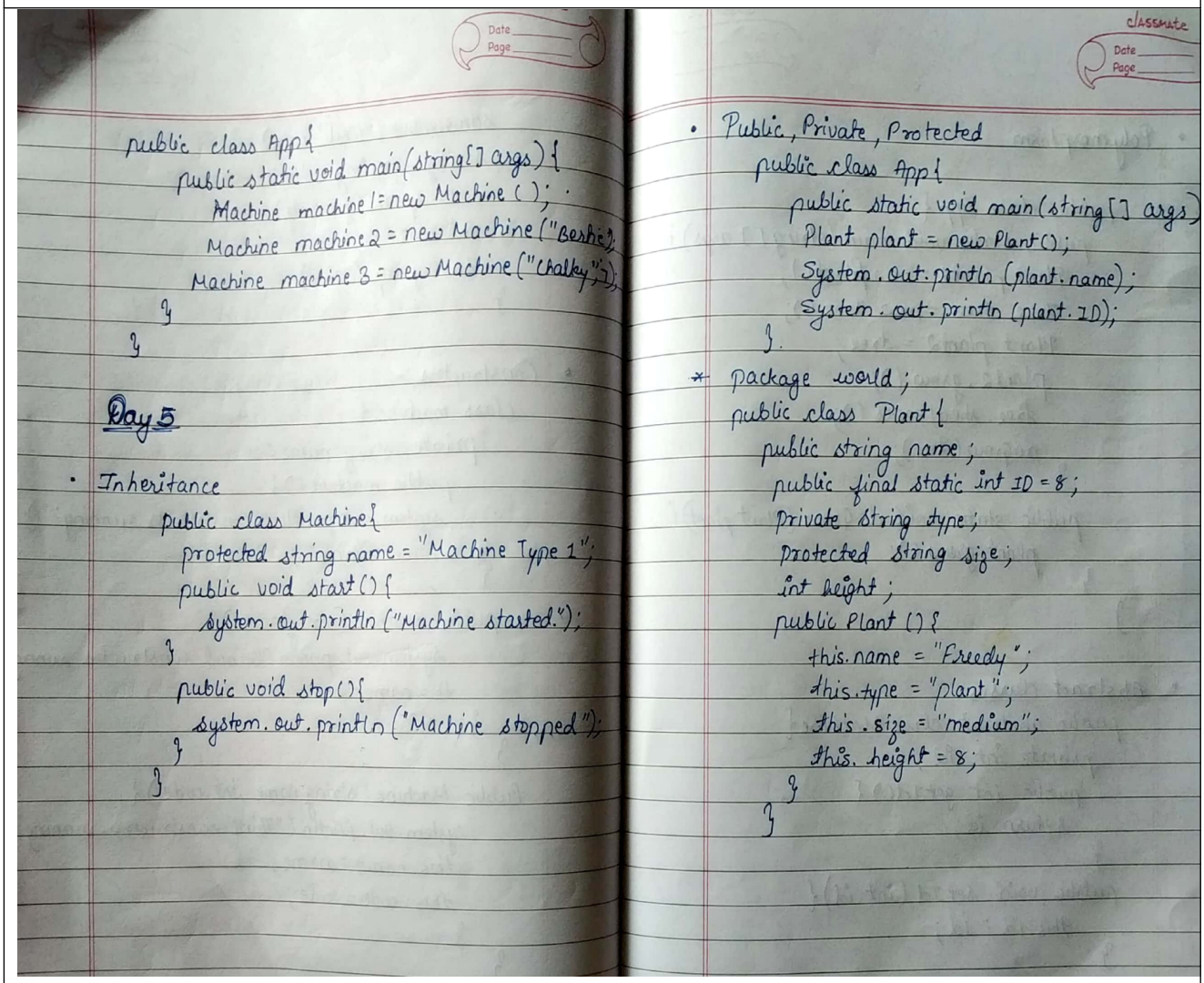
The drill file:You will also need to generate a file that indicates the position and size of every hole that will be drilled through your board, i.e., both through-holes (for mounting components) and vias. This is called the NC (numeric control) drill file; you may also see “Excellon drill file” (which comes from Excellon Automation, a company that makes

equipment used in PCB manufacturing). Again, the safest approach here is to follow specific instructions provided by a PCB manufacturer.

DAILY ASSESSMENT FORMAT

Date:	12-06-2020	Name:	POOJA K S
Course:	Java	USN:	4AL17EC070
Topic:	Programming core java	Semester & Section:	6TH SEM 'B' SEC

AFTERNOON SESSION DETAILS



```
public class App{
    public static void main(string[] args){
        Machine machine1=new Machine();
        Machine machine2 = new Machine("Bosh");
        Machine machine3 = new Machine("Chalky");
    }
}
```

Day 5

- Inheritance

```
public class Machine{
    protected string name = "Machine Type 1";
    public void start(){
        system.out.println("Machine started.");
    }
    public void stop(){
        system.out.println("Machine stopped");
    }
}
```

- Public, Private, Protected

```
public class App{
    public static void main(string[] args)
    Plant plant = new Plant();
    System.out.println(plant.name);
    System.out.println(plant.ID);
}
```

* package world;

```
public class Plant{
    public string name;
    public final static int ID = 8;
    private string type;
    protected string size;
    int height;
    public Plant(){
        this.name = "Freedy";
        this.type = "plant";
        this.size = "medium";
        this.height = 8;
    }
}
```

- Polymorphism

```
public class App {
    public static void main (string [] args) {
        Plant plant1 = new Plant();
        Tree tree = new Tree ();
        Plant plant2 = tree;
        plant2.grow ();
        tree.shedLeaves ();
        doGrow (tree);
    }
    public static void doGrow (Plant plant) {
        plant.grow ();
    }
}
```

- Abstract classes

```
public abstract class Machine {
    private int id;
    public int getId() {
        return id;
    }
    public void setId (int id) {
        this.id = id;
    }
    public abstract void: start() ;
    public abstract void: dostuff();
    public abstract void: shutdown();
    public void run () {
        start();
        doStuff();
        shutdown ();
    }
}
```

- Passing by Value:-

```
public static void main (string [] args) {
    App app = new App();
    int value = 7;
    system.out.println ("1. value is: "+ value
    app.show (value);
    system.out.println ("4. value is: "+ value
    system.out.println ();
    Person person = new Person ("Bob");
    system.out.println ("1. Person is: "+ pers
    app.show (person);
    system.out.println ("4. Person is: "+ perso
}
```