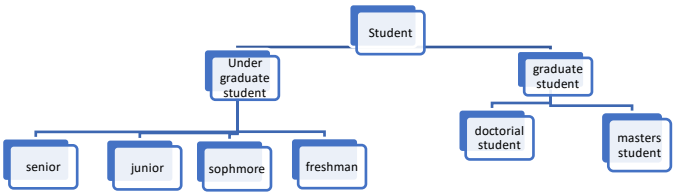
Noah Parker
Chapter 9 Study Sheet

Inheritance Hierarchies is a relationship between a general class which is a superclass and a more specialized class which is called a



subclass the chart below is an example using students. So, you have the superclass student which has a subclass of undergraduate student and graduate student and the undergraduate student has its own subclasses of senior, junior, sophomore and freshman. While the graduate student has the subclasses of doctorial and master student.

Implementing Subclasses, we use the term extends to show that the class is inherited from another class. The subclass which is also called a child class and the super class is also called a parent class. A subclass inherits from another class while the superclass is the one that is being inherited from. A code example for the students we used in the chart is

```
class Student {
  protected String year = "third
year";        // Student attribute
  public void timeleft() {
//   Student method
    System.out.println("I have one
year left!!");
  }
}

class Undergraduate extends
Student {
  private String typeofstudent =
"undergraduate";    //
undergraduate attribute
  public static void main(String[]
args) {

    // Create a mystudent object
    Student mystudent = new
Student();

    // Call the timleft() method (from
the Student class) on the mystudent
object
    mystudent.timeleft();

    // Display the value of the year
attribute (from the Student class)
and the value of the typeofstudent
from the undergraduate class

System.out.println(mystudent.year+
" " + mystudent.typeofstudent);
  }
}
```

This example is a little crude, but it gets the idea we need. What it does is the student is saying they have one year left as the parent class, but we need more information so in the undergraduate class that extends student we show that they are an undergraduate student. If they were third year in the student class, they could be a third-year masters or doctoral student with one year left but specifying in the class that extends it they are undergraduate which makes them a junior if they have one year left or a second year senior but it makes it so the confusion of what type of student more clear.

Overriding Methods the subclass inherits the methods from the parent class. If you are not happy with the behavior of the inherited method, you can override it by implanting it in the subclass. An example in code would be →

So, in this example we have students who all work on school and sometimes research projects And then we a freshman class who does schoolwork but no research projects.in the test method we set student which is the parent class to the subclass of freshman. When the output goes the first output of a.work does the work method. And the output of the b.work is the noresearch method. This shows how you can override the parent class method.

Polymorphism is the ability of the message to be displayed in more than one form. An example of it in works is a student at the same time can have different characteristic. Like a junior at the same time is a teaching assistant, a club leader. So, the same student possesses different behavior in different situations. This is called polymorphism. There are two different types of polymorphism method overloading which is when you create multiple methods of the same name in the same class while all those methods work in different ways. The other type is method overriding which is when the subclass or child class has the same method as declared in the parent class. An example of this is below

```
class Student{  method
  void work(){System.out.println("I'm doing work");}
}
//Creating a child class
class work2 extends Student{
  //defining the same method as in the parent class
  void work(){System.out.println("I'm doing a biology take home lab");}

  public static void main(String args[]){
 work2 obj = new work2();//creating object
  obj.work();//calling method
  }
}
```

This overrides the original method to be more specific about what type of work the student is doing. The Cosmic Superclass every class that is not declared with a extends automatically extends the class Object. The class Object is direct or indirect superclass of every class in java.

```
class Student {
  public void work() {
    System.out.println("Students do
work and sometimes research
projects");
  }
}

class freshman extends Student {
  public void noresearch() {
    System.out.println("freshman
do work and do not do research
projects ");
  }
}

public class TestFreshman {

  public static void main(String
args[]) {
    Student a = new Student();   //
Animal reference and object
    Student b = new freshamn();   //
Student reference but freshman
object

    a.work();   // runs the method in
Student class
    b.work();   // runs the method in
freshman class
  }
}
```