**OO Practical – Overriding equals; implementing interfaces; using our own interfaces.**

A basic bare-bones Person is given:

public class Person

{ private String name; private int age;

public Person(String name, int age)

{ this.name = name; this.age = age;

}

}

**Exercise 1:** Override the toString() method. Don’t forget to use the @Override directive.

**Exercise 2:** Amend the Person class so that a tester (which you must also provide) containing the following code will work properly. Hint: what facility must the arraylist contains() method use in order to search for object matches within its list?

ArrayList<Person> people = new ArrayList<Person>();

//Add some Person objects to the list

if (people.contains(new Person("Adam Ant", 48)))

{

//Do something – e.g. print out match found

} else

{

//Do something else – e.g. print match not found

}

**Exercise 3:** Add Comparable to the Person class, implement CompareTo and call Collections.sort() in order to sort your list by age.

**Exercise 4:**

Identify what is wrong with the following interface:

public interface SomethingIsWrong { void aMethod(int aValue) {

System.out.println("Hi Mom");

}

}

Issue: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Exercise 5:**

Explain what is required to fix the interface from the previous exercise

Fix:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Exercise 6:**

Provide a Student class (using inheritance from the Person class):

The Student class should have the following instance fields: name (String), age (int), course(String), **spendingMoney (double).**

In addition, the Student class is capable of doing part-time jobs. To facilitate we will use a PartTimeAble interface and a Job class ***which are explained on the next pages***.

**You need to provide the interface and the Job class.**

Your student class should:

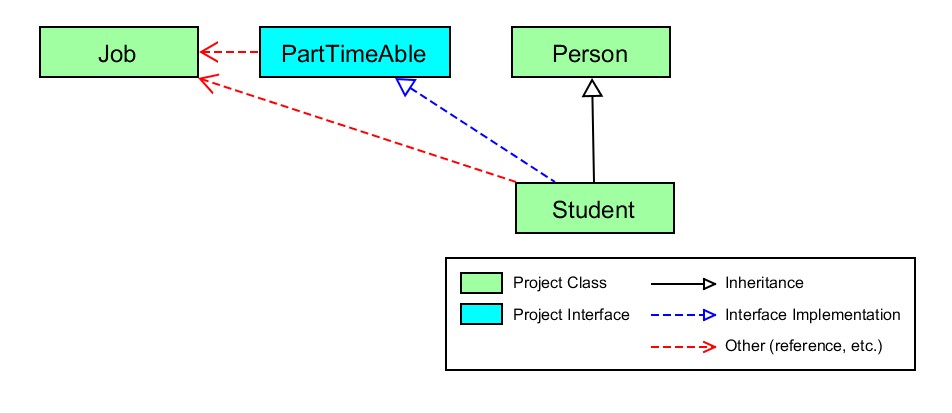
1. Have an overloaded constructor to setup the instance fields of the student objects. Note that initially a student object has no spending money.
2. Implement the PartTimeAble interface. For the Student class implementation it merely updates the student’s spendingMoney with the amount earned from the job.
3. Have a method with the signature public int howManyNoodles (double pricePerPacket). This method should return how many packets of noodles the student can afford to buy.

Provide test code to verify that your classes and interfaces work. This test code should:

* Create at least one Student object.
* Call the doJob() method one or more times to give the student some spending money.
* Find out and report how many packets of noodles the student can afford to buy.

Note: make sure you use access specifier “public” for the method doJob()

The diagram below shows the relationships between the various components.



Please refer to the subsequent pages for more information on PartTimeAble and Job.

