OOPS

1. Write a method that overloads the talk method by taking in a name and printing “Hello” with that name.
2. Create an interface named Test that has a void talk method and void walk method.
3. Edit this code so the class Beagle is a subclass of the Dog class. When you run the code it should print “woof!” and then “arf arf”

```
public class Dog
{
   public void speak()
   {
      System.out.println("woof!");
   }

   public static void main(String[] args)
   {
      Dog d = new Dog();
      d.speak();
      Dog b = new Beagle();
      b.speak();
   }
}

class Beagle
{
   public void speak()
   {
      System.out.println("arf arf");
   }
}
```

4. Add an equals method to this class that returns true if the current Dog and passed Dog have the same name. The code should print false twice then true twice.

```
public class Dog
{
   private String name;

   public Dog(String name)
   {
      this.name = name;
   }

   public boolean equals(Object other)
```

```
    {
        // ADD CODE HERE
    }

    public static void main(String[] args)
    {
        Dog d1 = new Dog("Rufus");
        Dog d2 = new Dog("Sally");
        Dog d3 = new Dog("Rufus");
        Dog d4 = d3;
        System.out.println(d1.equals(d2));
        System.out.println(d2.equals(d3));
        System.out.println(d1.equals(d3));
        System.out.println(d3.equals(d4));
    }
}
```

5. Override the taste method from the Candy class in the Chocolate class to return “tastes chocolately”. It should print “tastes sweet!” and then “tastes chocolately”.

```
public class Candy
{
    public String taste()
    {
        return "tastes sweet!"
    }

    public static void main(String[] args)
    {
        Candy c1 = new Candy();
        System.out.println(c1.taste());
        Candy c2 = new Chocolate();
        System.out.println(c2.taste();
    }
}

class Chocolate extends Candy
{
    // ADD CODE HERE
}
```

6. Overload the greet method to just print “Hello” if not given any parameters. It should print “Hello” and then “Hello Sansa”.

```
public class Student
{
    public static void greet(String name)
    {
        System.out.println("Hello " + name);
    }

    public static void main(String[] args)
    {
       greet();
       greet("Sansa");
    }
}
```

7. Add a call to Pet’s brag method before printing anything in Dog’s brag method (hint: use super to call an overridden method). It should print “I have the best pet!” and then “I have the best dog”.

```
public class Pet
{

    public void brag()
    {
        System.out.println("I have the best pet!");
    }

    public static void main(String[] args)
    {
        Dog d1 = new Dog();
        d1.brag();
    }
}

class Dog extends Pet
{
    public void brag()
    {
        // ADD CODE HERE

        System.out.println("I have the best dog!");
    }
}
```

8. Finish the Teacher constructor. Use super to use the Person construtor to set the fields inherited from Person. It should print “Destini 20” followed by “Erica 55 Masters in Teaching”.

```
public class Person
{
    private String name;
    private int age;

    public Person(String name, int age)
    {
        this.name = name;
        this.age = age;
    }

    public String getName() { return this.name; }

    public int getAge() { return this.age; }

    public String toString()
    {
        return getName() + " " + getAge();
    }

    public static void main(String[] args)
    {
       Person p = new Person("Destini", 20);
       System.out.println(p);
       Teacher p2 = new Teacher("Erica", 55, "Masters in Teaching");
       System.out.println(p2);
    }
}

class Teacher extends Person
{
    String degree;

    public String getDegree() { return this.degree; }

    public String toString()
    {
        return getName() + " " + getAge() + " " + getDegree();
    }

    public Teacher(String name, int age, String theDegree)
    {
        // ADD CODE HERE
    }
}
```

9. Add public getter and setter methods to the Store class so its variables can be accessed by other classes. It should print the store’s name and address and then change both and print the new values.

```
public class Store
{
    private String name;
    private String address;

    public Store(String theName, String theAddress)
    {
        this.name = theName;
        this.address = theAddress;
    }

    // ADD CODE HERE

    public String toString() { return this.name + "\n" + this.address; }

    public static void main(String[] args)
    {
        Store myStore = new Store("Barb's Store", "333 Main St.");
        System.out.println(myStore);
        myStore.setName("Barbara's Store");
        myStore.setAddress("555 Pine St.");
        System.out.println(myStore);

    }
}
```

10. Correctly finish the Dog subclass for the following abstract Animal class. Override the abstract methods. It should print “woof” and then “num num”.

```
abstract class Animal
{
    public String name;
    public int numLegs;
    public abstract void speak();
    public abstract void eat();

    public static void main(String[] args)
    {
       Dog myDog = new Dog();
       myDog.speak();
       myDog.eat();
    }
}
```

```
public class Dog extends Animal
{
   // ADD CODE HERE

   public static void main(String[] args)
   {
      Dog myDog = new Dog();
      myDog.speak();
      myDog.eat();
   }
}
```

11. Override the compareTo method so that it returns a postive number if the current Person is older than the passed other and a negative number if they are younger. If their age is the same then return the compareTo result on the names.

```
public class Person implements Comparable<Person>
{
   private String name;
   private int age;

   public Person(String name, int age)
   {
      this.name = name;
      this.age = age;
   }

   public int compareTo(Person other)
   {
      // ADD CODE HERE
   }

   public static void main(String[] args)
   {
      Person p1 = new Person("Carlos",17);
      Person p2 = new Person("Lia",18);
      Person p3 = new Person("Asraf", 17);
      Person p4 = new Person("Lia", 17);
      Person p5 = new Person("Karla", 17);
      System.out.println(p1.compareTo(p2));
      System.out.println(p2.compareTo(p3));
      System.out.println(p3.compareTo(p1));
      System.out.println(p4.compareTo(p3));
      System.out.println(p4.compareTo(p5));
    }
}
```

12. Override the Person class’s speak function inside the Student class. Make the function print “I’m a student”.

```
public class Person
{
   public void speak()
   {
      System.out.println("I'm a person");
   }

   public static void main(String[] args)
   {
      Person p1 = new Student();
      p1.speak();
   }
}

class Student extends Person
{
   // ADD CODE HERE
}
```