1 Generics in Java

[10 pts] Consider this Java class:

```java
class Cell<T> {
    private T t;
    public Cell(T t) { this.t = t; }
    public void set(T t) { this.t = t; }
    public T get() { return t; }
    public boolean equalTo(Cell<T> other) { return this.t.equals(other.t); }
    public Cell<T> copy() { return new Box<T>(t); }
}
```

Which statements in the following code produce a compile-time error? Briefly explain.

```java
Cell<?> cell = new Cell<String>("hello");
cell.set("world");
cell.set(null);
String s = cell.get();
Object o = cell.get();

boolean equal = cell.equalTo(cell);
equal = cell.equalTo(new Cell<String>("hello");

Cell<?> cell1 = cell.copy();
Cell<String> cell2 = cell.copy();

Cell<? extends Number> cell3 = new Cell<Integer>(new Integer(1));
cell3.set(new Integer(2));
Number n = cell3.get();

Cell<? super Integer> cell4 = new Cell<Integer>(new Integer(1));
cell4.set(new Integer(2));
Integer i = cell4.get();
Object p = cell4.get();
```
2 Prototypes

Consider the following JavaScript code:

```javascript
function Point(x, y) { this.x = x; this.y = y; }
Point.prototype.add = function(p) { return new Point(x + p.x, y + p.y); }
var z = new Point(0,0)

(a) [5 pts] Draw a sketch of the layout in memory of the object stored in z. Include its fields and any other objects z itself transitively has references to.

(b) [5 pts] Write JavaScript code to create a single new object that behaves exactly like z but also contains a color field.

(c) [5 pts] Create a constructor for ColorPoint that returns an object containing x, y, and color fields. The expression new ColorPoint(0,1,"blue") should create a new object with the x, y, and color fields initialized to 0, 1, and "blue", respectively.

(d) [5 pts] Now, create a constructor for ColorPoint that returns an object containing a color field, but which delegates to a Point object for the x and y fields.