Introduction to Computer Science II
Lab 3
Abstracting `Cell`

1 Different cell types

Revisit the Game of Life page on Wikipedia, and read the section on *Variations*. Specifically, this passage describes that the standard cell in this game—what we will call a *Conway* cell—follows rules that could be varied. It describes a *Highlife* cell that is a modest variation on the *Conway*. A *Highlife* cell, when evolving, is...

- **born**[1] if it has 3 or 6 live neighbors;
- **survives**[2] if it has 2 or 3 live neighbors; and,
- **is dead** under all other circumstances.

2 Modifying your code

Get the code: Start a *Terminal*, make a directory, and grab the code:

```
$ cd
$ curl -L https://bit.ly/cosc-112-24f-l3 -o lab-3.zip
$ unzip lab-3.zip
$ cd lab-3
$ code .
```

You will find most of the same classes from Project-1. If you examine `Life.java`, `Game.java`, and `Grid.java`, you will notice that a `cellType` variable is passed in the process of creating the grid of cells. Just as importantly, notice that the `Cell` class is an *abstract* class—a concept that we will discuss at the beginning of lab. It has two methods, `evolve()` and `toString()`, that are *declared but not defined*: they have no body. Those methods must be defined in any subclass.

Take special notice, in the `Cell` class, of the method, `create()`. This static is responsible for creating a `ConwayCell` or a `HighlifeCell` depending on the cell type requested in the string `cellType`. This kind of creation method encapsulates the problem of turning an input request string into a specific subclass object type—something constructors cannot do.[3]

[1] That is, it is currently *dead* and becomes *alive*.

[2] That is, it is current *alive* and remains so.

[3] Do you see why not? It's the kind of thing that makes a great mid-term exam question.

What you must do: Create two `Cell` subclasses: `ConwayCell` and `HighlifeCell`, which implement those two methods using their respective rules. (Use `+` and `-` to represent live and dead Conway cells; use `*` and `~` to represent live and dead Highlife cells.) Your `ConwayCell` class can use code you wrote from Project-1.

3 How to submit your work

Submit your `ConwayCell.java` and `HighlifeCell.java` files by uploading them into the `lab-3` folder in your shared Google Drive folder for this course.

This assignment is due on Sunday, Oct-06, 11:59 pm.