CS 111 - Week 3 Lab Exercise - 2016-09-09

Deadline

Due by the end of lab on 2016-09-09 (Submit whatever you have by the end of lab, even if incomplete.)

How to submit

Submit your Definitions window (`.rkt` file) for this lab using `~st10/111submit` on nrs-labs, with a homework number of `83`.

Purpose

To practice creating named constants and your own function while in the process of designing and creating an animation.

Important notes

- You are required to work in **pairs** on this lab exercise (unless there are special circumstances, such as an odd number of students in lab, in which case a trio may be approved... 8-)).
  - Remember, in pair programming both people work at one computer: one types ("drives"), while the other tells him/her what to type ("navigates").
  - You may switch roles while working on this lab exercise if you would like -- but I will be trying to see if people are really pair-programming.
  - If you are not pair-programming, then you may not receive full credit for your lab exercise.
- If you have a question during lab, and I am helping another pair, add one or both of your names to the "Next:" list on the board, and I will get to you as soon as I can.

Your tasks:

- Begin a new DrRacket **Definitions** window, and type in **comments** containing your names and today's date. Save this in a file with the name `lab4.rkt`
- Put in the expressions to add the `image` and `universe` teachpacks:

```
(require 2htdp/image)
(require 2htdp/universe)
```

- Decide how wide and high you want your scene to be; define named constants `WIDTH` and `HEIGHT` to be these values.
- Define a nice named constant `BACKDROP` using `place-image` to include at least two (**static, unchanging**) images of your choice within a scene.

(**continued on back!)**

- DECIDE: are you going to animate an image whose SIZE changes, or are you going to MOVE it?
  - **IF** you are going to MOVE it, then define or find a small **image** to **move** around in a scene,
  - and then use `define` to make this image-to-be-moved a named constant.
- DECIDE: how will your "world" value change on each `big-bang` ticker tick?
  - Will it increase by 1 or decrease by 1? In those cases, you can use the built-in functions `add1` or `sub1`, respectively, as the function argument for the `on-tick` clause.
  - OR you can write your own, different function to change your "world" value on each ticker tick.

    (Remember: you are expected to develop and include its signature, purpose, and `check-expect`s as well as its function definition. We'll include these for every function definition -- they are our documentation.)
- Decide on the "action" you want in your animation -- it need to be something BESIDES a growing star, and something BESIDES just moving left-to-right.
  - (For example, a **different** shape could grow or shrink, or something could move from top-to-bottom, or something could move diagonally, etc.) Combinations of changes are fine, too.
- Design a drawing function that expects a number, and produces a scene based on that number (to be the function argument for `big-bang`'s `to-draw` clause). This function is expected to:
  - use the `BACKDROP`, `WIDTH`, and `HEIGHT` you defined above
  - if you are MOVING an image, you will place that image in your `BACKDROP` based on the parameter number in an appropriate expression of your choice;

    if you are CHANGING an image's size, you will place an image created with the help of the parameter number in your `BACKDROP`;
  - FIRST: write a SIGNATURE for your drawing function
  - SECOND: write a PURPOSE STATEMENT for your drawing function
  - THIRD: write a FUNCTION HEADER for your drawing function (and for now, put `...` for the function body)
  - FOURTH: write at least two `check-expect` expressions showing what at least 2 different calls of your drawing function SHOULD result in
  - FIFTH: use what you learned writing those `check-expect` expressions to replace the `...` with a functional function body
  - SIXTH: click Run, and see if your tests pass
- Write a `big-bang` expression using your two functions above, to run your animation.
- When you are happy with the above (OR when time is up), save your Definitions window to the U: drive, use `ssh` to connect to nrs-labs, and use `~st10/111submit` with a homework number of **83** to submit your completed lab exercise.
  - Make sure you BOTH have a copy, also!