

Attempt 1

Review Feedback
9/27/2023Attempt 1 Score:
100/100

View feedback

Unlimited Attempts Allowed

10/1/2023

Details

[Activity 1: Red Squa](#)[Activity 2: Blue Circl](#)[Activity 3: Barbie Pir](#)[Activity 4: Outline M](#)[Activity 5: Compound](#)[Activity 6: Rotating](#)[Activity 7: Flag of Cl](#)[Double Check your I](#)[Submitting to Canva](#)[Late Penalty Waiver](#)

In this assignment you'll play with simple functions for making images that we'll use later on in the course.

Note: We've tried to make this assignment accessible to those who have trouble seeing colors, but please let us know if we can provide additional help.

In order to start this assignment, please download the starter files:

Exercise 1 Starter Files (https://bain-cs111.github.io/course-files/exercises/exercise_1_template.zip)

This download is a ZIP file which *must be extracted* before the files can be edited. On a Mac, you can double click on the ZIP folder and it will extract the files for you. On a Windows computer, you can use [these instructions from Microsoft](https://support.microsoft.com/en-us/windows/zip-and-unzip-files-8d28fa72-f2f9-712f-67df-f80cf89fd4e5) (<https://support.microsoft.com/en-us/windows/zip-and-unzip-files-8d28fa72-f2f9-712f-67df-f80cf89fd4e5>).

Note: We highly recommend moving the extracted folder to a CS 111 work folder somewhere on your machine (so you have all your 111 stuff in the same place) and renaming it to make sure you submit the file you mean to later when you go to submit.

Now, open up the extracted file. For this exercise, it's called `exercise_1.rkt`.

Activity 1: Red Square

First, let's make a 100x100 red square like the below:



Now you should give this square a name. In the Definitions Window type:

```
(define a-red-square your-code-for-the-square)
```

and hit run.

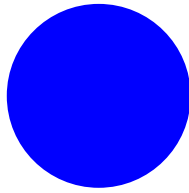
You can now refer to the square by typing `a-red-square` into the interaction window or any code you write.

VERY Important Note: The names you give your images (i.e. the names after the word "define") **must be**

<https://canvas.northwestern.edu/courses/201068/modules/items/2824138><https://canvas.northwestern.edu/courses/201068/modules/items/2820279>

Activity 2: Blue Circle

Now let's make a blue circle of radius 50.



Define it as the name: `a-blue-circle`.

Activity 3: Barbie Pink Triangle

There's a ton of built-in colors in the ISL, stored in special data object called the `color-database`. We won't worry about the details of this more complex piece of data right now, but suffice it to say that when Racket sees you've given it a String as a color input, it looks that string up [in this database](https://docs.racket-lang.org/draw/color-database.html) to see if it has a matching color and replaces that string with the actual "recipe" for that color.

Colors are represented on computers as mixtures of Red, Green, and Blue values – you can imagine it like mixing together red, green, and blue paints to get different colors. Each pixel (i.e. tiny square) on your computer screen gets assigned a different "intensity" of red, green, and blue which mix together to form its color. This means that we don't actually have to use a color's name as inputs for these shape functions; instead we can specify the exact mixing of red, green, and blue light that we'd like (these intensities are lowest at a numeric value of 0 and highest at a numeric value of 255).

For instance, confoundingly `"Barbie Pink"` is not a recognized color in ISL. `"Barbie Pink"` [is officially Pantone Color 219C](https://www.google.com/search?q=pantone+219C) which has a red value of 218, a green value of 24, and a blue value of 132. To make a color in the ISL, we can use the `make-color` function which has the following type signature (i.e. recipe):

```
; make-color: number number number -> Color
; or slightly more specifically...
; make-color: (integer-in 0 255) (integer-in 0 255) (integer-in 0 255) -> Color
```

The first input must be the red intensity, the second must be the green intensity, and the third must be the blue intensity.

Use the `make-color` function to create our new color and then use that as an input to some function that allows us to make an upward pointing [equilateral triangle](https://docs.racket-lang.org/teachpack/2htdpimage.html#%28def._%28lib._2htdp%2Fimage._rkt%29._triangle%29%29) that is solid and has a side length of 100.



Define it as the name: `a-barbie-triangle`.

Activity 4: Outline Mode

Repeat Activities 1, 2, and 3 but use outline mode:



(<https://canvas.northwestern.edu/courses/201068/modules/items/2824138>)


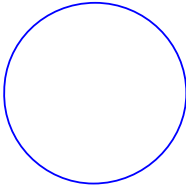
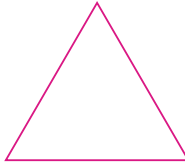


(<https://canvas.northwestern.edu/courses/201068/modules/items/2820279>)

outlined-square

outlined-circle

outlined-triangle

Define them as the names `outlined-square`, `outlined-circle`, and `outlined-triangle` respectively.


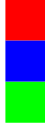

Activity 5: Compound Images

Now let's make compound images from simpler images. Use `overlay`, `above`, and `beside` to make the following compound images, defined as follows:

row-of-squares

column-of-squares

nested-squares

Make sure to draw out the dataflow diagrams for each of these. You don't need to turn them in—just draw them on some scratch paper or a whiteboard to make sure you understand how the data moves through the chain of calls.

Activity 6: Rotating

Read the [documentation](https://docs.racket-lang.org/teachpack/2htdpimage.html)  (<https://docs.racket-lang.org/teachpack/2htdpimage.html>) for the `rotate` function and try making an image that looks like this:



Define it as `barbie-bowtie`.

Activity 7: Flag of Chicago

Now, make the flag of Chicago:



(<https://docs.racket-lang.org/teachpack/2htdpimage.html>) and your image doesn't need to be exact! Define it as `flag-of-chicago`. And again, make sure that you can sketch out the dataflow diagram for it.

Note: there is no need for any lambda expression in ANY of these activities.

Double Check your Work

Pro-tip 1: do not MOVE an already open file. If you open a file in DrRacket (or any app) and THEN move the file in your operating system, the app doesn't know the file was moved. It will assume you want to save it in the older folder.

Pro-tip 2: do not submit a file that you currently have open. Instead, finish your work then close DrRacket. That way, there's no possible way to submit an older file.

Pro-tip 3: do not have multiple versions of the same file. It just makes it harder for you to keep track of what's what.

Make sure you've uncommented all of those built-in `check-expect` as they will double check you've defined all of your variables correctly. Also, after you submit, keep an eye out for the Canvas comment that the TypeChecker will post for you that double checks your built-in tests as well as the types of the various things we've asked you to define.

Submitting to Canvas

Questions about submitting to Canvas? About DrRacket features? Checkout our FAQ pages:

- [DrRacket Cheat Sheet \(https://canvas.northwestern.edu/courses/201068/pages/drracket-cheatsheet\)](https://canvas.northwestern.edu/courses/201068/pages/drracket-cheatsheet)
- [Checking for Runtime and Syntax Errors \(https://canvas.northwestern.edu/courses/201068/pages/checking-for-syntax-or-runtime-errors-in-racket\)](https://canvas.northwestern.edu/courses/201068/pages/checking-for-syntax-or-runtime-errors-in-racket)
- [Submission and Grading FAQ \(https://canvas.northwestern.edu/courses/201068/pages/exercise-submission-and-grading-faq\)](https://canvas.northwestern.edu/courses/201068/pages/exercise-submission-and-grading-faq)
- [Autograder FAQ \(https://canvas.northwestern.edu/courses/201068/pages/autograder-and-submission-faq\)](https://canvas.northwestern.edu/courses/201068/pages/autograder-and-submission-faq)

Remember, **close DrRacket before you submit** to ensure that you've saved your latest work. The Automated Type Checker (ConnorBot) will post its results of its type checks and the built-in `check-expect`s every 30 minutes. If you get an unexpected failure, it may be because you submitted the wrong file.

Late Penalty Waiver

Remember, the deadline for submitting this waiver is 24 hours before the DUE AT time on Canvas.

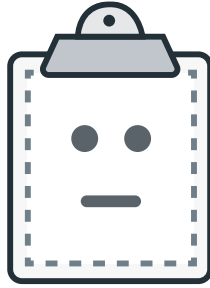
If you need to request a late penalty waiver for this assignment, use the [Late Penalty Waiver form](#).



(<https://canvas.northwestern.edu/courses/201068/modules/items/2824138>)



(<https://canvas.northwestern.edu/courses/201068/modules/items/2820279>)



Preview Unavailable

exercise_1.rkt

 [Download](#)

(https://canvas.northwestern.edu/files/17296129/download?download_frd=1&verifier=6VDOCmO9SkanWHoCLa3EXeamaFwchksZL2Dax4oo)



(<https://canvas.northwestern.edu/courses/201068/modules/items/2824138>)



(<https://canvas.northwestern.edu/courses/201068/modules/items/2820279>)