Racket Day 1 – Calculator Functions

Exercise 1: In exercise one you will learn how to execute basic mathematical functions in the racket language.

Examples: To execute basic mathematic functions in racket, you will need to know that racket is an operator first language. Procedures are wrapped in parenthesis. So if I want my code to add 1 and 2 I would type:
(+ 1 2)
in the top window of dr racket and then when I run my code I should see that code executes to give me 3.

Please complete the following math problems…use your phone as a calculator to check your answers.

a.) 23+ 5

b.) 15 – 6

c.) 24 / 6

d.) 5 * 9

Exercise 2: Use what you have learned about the basic operations to now complete these order of operations problems using racket.

In order to do multiple operations you will need to learn "nesting". If I wanted to multiply 2 and 10 first then take that answer and subtract 3 I would start with the multiplication problem and then wrap the subtraction around it:
Step 1: (* 2 10)
Step 2: wrap the subtraction around the multiplication to get (- (* 2 10) 3)

Please complete the following math problems. Keep in mind you need to think about the order of ops and build from the first operation and wrap the problems from there out! Don't forget PEMDAS!

a.) 3 * 5 + 2 – 3 (answer should be 14)

b.) 20 – 5 * 3 (answer should be 5)

c.) 12 / 2 +5 – 3 * 2 (answer should be 5)

Racket Day 1 – Calculator Functions

Exercise 3: Writing functions

To write a function you will need to start with define, give the function a name, tell it what to expect, and finally tell it what to do.

This function takes one number and always adds 2 to it.

```
(define addtwo
        (lambda (num)
                (+ num 2)))
```

1.) Notice I started with an open parenthesis, then a space and then "addtwo" is the name of my function.
2.) Next you will see another open parenthesis and the word "lambda" which means "given" or "expect these things". This function only expects one thing to be given "num" is what I am calling the number this function expects to get.
3.) Last, you tell it what to do with what it was given. Notice another open parenthesis and then it says to take the num and add 2 to it. Notice each set of parenthesis that we opened now gets closed at the end.

Try typing this code on the top of your racket window. When you click run, you wont see anything happen because you have to call the function.

In the bottom half of your window now try out your function:

```
>(addtwo 3)
5

>(addtwo 10)
12
```

Now, write your own function called "squareit" that will square a number. (Multiply a number times itself)

Test out your function with 3 different numbers .

Racket Day 1 – Calculator Functions

Exercise 4: Next, write a function called "cubeit" that will cube a number. (Multiply a number times itself 3 times…. 2^3= 8 because 2*2=4*2=8

Test out your function with 3 different numbers….use a calculator to test out your answers.

Exercise 5: Write a function called "double" that will double a number. (There are two different correct ways to write this one! Can you think of both?)

Test out your function with 3 different numbers…use a calculator to test out your answers.

Exercise 6: Write a function called "upplace" that will take number and increase the place value.

For example (upplace 3) => 30 or (upplace 44) => 440

Test out your function with 3 different numbers.