CS 596 Fall 2014 Exam

Name _____

1. (17 points) What is the result of evaluating each of the following. They are evaluated one at a time.

```
(get {:a 1 :b 2} :b)
(get {:a 1 :b 2} :b 10)
({:a 1 :b 2} :b)
({:a 1 :b 2} :b 10)
(:b {:a 1 :b 2})
(:b {:a 1 :b 2} 10)
(= 2 2.0)
(== 2 2.0)
(first '(1 2 3))
(first [1 2 3])
(pop '(1 2 3))
(pop [1 2 3])
(cons :a '(1 2 3))
(cons :a [1 2 3])
(conj '(1 2 3) :a)
(conj [1 2 3] :a)
(number? :a)
```

2. (10 points) Write a function sdsu-last. The function takes a variable number of arguments, none of them collections. Sdsu-last returns the last argument in its argument last. For example (sdsu-last 5 4 3 2 1) returns 1 and (sdsu-last 5) returns 5.

3. (10 points) Write a clojure function (isSumOrProduct coll, x) where coll is a collection of numbers, and x is a number. isSumOrProduct returns true if and only if x is either the sum or the product of the numbers in coll. For example (isSumOrProduct [2 2 3] 7) and (isSumOrProduct [2 3] 6) both return true, while (isSumOrProduct [2 3] 4) returns false. You can assume that the sum of the empty list is 0 and the product of the empty list is 1. Use a precondition to insure that x is a number.

4. (10 points) Let a binary tree be represented by a map. For example here is a tree with two nodes: {:key 5 :left: nil :right {:key 10 :left nil :right nil}. The height of an empty tree is 0. The height of a non-empty tree is 1 + max height of the trees two sub-trees. Write a clojure function with one argument, a tree, that returns the height of the tree.

5. a. (5 points) Give an example of destructuring a map in a let form.

b. (5 points) Give an example of destructuring a vector in a let form.

6. a. (3 points) What is a pure function?

b. (3 points) Why is it important in Functional Programming?

c. (3 points) Given an example.

7. a. (3 points) What is a higher-order function?

b. (3 points) Why is it important in Functional Programming?

c. (3 points) Given an example.

8. (10 points) Explain how the threading macro -> works. Give an example.

9. a. (3 points) What is a lazy evaluation?

b. (3 points) Why is it important in Functional Programming?

c. (3 points) Given an example.

10. (10 points) What is tail recursion optimization? Show how Clojure mimics tail recursion optimization.

11. (10 points) Write a function sdsu-partial which will act like closure's partial. Sdsu-partial takes a function f and one other argument x. Assume that the function f requires two arguments. Sdsu-partial returns a function that takes one argument y. When called, the returned function calls (f x y). For example let (def g (sdsu-partial + 5)) then (g 3) returns 8. Do not use the function partial.