**COMP 333 Final Practice Exam**

The final exam is cumulative. This practice exam, **in addition to** the prior practice exams, assignments, and in-class handouts, is intended to be a comprehensive guide for studying. This practice exam only focuses on material since the last exam.

**Swift**

1.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine<A, B>(a: A, b: B) -> (A, B) {

 return (a, b)

}

2.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine2<A, B>(a: A) -> ((B) -> (A, B)) {

 return { b in (a, b) }

}

3.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine3<A, B>(tup: (A, B)) -> A {

 let (a, \_) = tup

 return a

}

4.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine4<A, B>(a: A, f: (A) -> B) -> (A, B) {

 return (a, f(a))

}

5.) Consider the following enum definition:

enum Something<A, B, C> {

 case alpha(A)

 case beta(B)

 case gamma(C)

}

5.a.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine5<A, B, C>(s: Something<A, B, C>) -> (A, B, C) {

 Impossible to implement. s holds **one** of an A, B, or C, and the return type requires **all** three

}

5.b.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine6<A>(s: Something<A, A, A>) -> A {

 switch s {

 case .alpha(let a): return a

 case .beta(let a): return a

 case .gamma(let a): return a

 }

}

6.) Write the body of the following function, or say if it's impossible to implement. If it's impossible to implement, explain why.

func combine7<A, B>(f: (A) -> B, b: B) -> A {

 Impossible to implement. f needs an A, but we only have a B.

}

7.) Consider the following code:

let i1 = 5.add(3);

let i2 = 7.add(10);

print(i1); // prints 8

print(i2); // prints 17

Define any Swift code below to make the above code have the correct output. As a hint, you'll need to use extension.

extension Int {

 func add(\_ other: Int) -> Int {

 return self + other

 }

}