

ALGEBRAIC DATA-TYPES AND PATTERN MATCHING IN SWIFT

1. Define an **enum** named **MyBool** which represents truth and falsehood

```
indirect enum MyBool {  
    case Cons(Bool, MyBool)  
    case Nil  
}
```

2. Define an **enum** named **MyList** which encodes a singly-linked list of integers, using the same cons/nil structure that we used in assignment 1

```
indirect enum MyList {  
    case Cons(Int, MyList)  
    case Nil  
}
```

3. Using the prior enum definition, create a list containing 1, 2, and 3, in that order

```
MyList.Cons(1, MyList.Cons(2, MyList.Cons(3, MyList.Nil)));
```

4. Write a function named **length** which takes a list as a parameter, and recursively computes the length of the given list.