PIC 10A 1C. Week 6a Exercises. TA: Eric Kim. [Solutions] *(Updated: 2/9/2016 v2)*

1. Apples and Oranges

Consider the following class interfaces:

```
class Apple {
private:
  int mya;
public:
  int myb;
  Apple();
  Apple(int a);
  void foo(Apple a);
};
```

```
class Orange {
public:
  int myc;
  Orange(int c);
  void garply(Apple a, Orange
b);
};
```

Consider the following code. Are there any issues?

```
Apple apple1(2);
Orange orange1(3);
Apple apple2(orange1.myc);
Orange orange2(apple2.mya);
garply(apple1,orange1);
orange1.foo(apple1);
orange1.garply(Apple(1), Orange(2));
```

[Solution]

```
Apple apple1(2);
Orange orange1(3);
Apple apple2(orange1.myc);
Orange orange2(apple2.mya); // mya is private! can't access
garply(apple1,orange1); // need to call garply on Orange object, ie
    // orange1.garply(apple1,orange1);
orange1.foo(apple1); // foo is not a method of the Orange class,
    // is a method of the Apple class.
orange1.garply(Apple(1), Orange(2));
```

2. const Practice

Identify any possible issues with the code:

[Solution]

```
int a = 1;
const int b = 1;
int& c = a;
int& d = b; // ERROR: Can't point non-const reference to const.
c = 0;
```

```
const int& aa = a;
const int& bb = b;
aa = 4; // ERROR: Can't modify a const reference
bb = 4; // ERROR: Can't modify a const reference
```

3. Robbers Robbing Robbers

Define a Robber class interface that satisfies the following code:

```
Robber rusty("Rusty");
Robber dan("Dan")
rusty.greet(dan);
dan.greet(rusty);
int item_to_steal = dan.lookat(rusty);
dan.steal(rusty, item_to_steal);
cout << rusty.yell();
```

[Solution]

Note: Many of the method return types are ambiguous (ie Robber::greet could return, say, an int, and still adhere to the above code), but here's one valid interface:

```
class Robber {
public:
    string myname;
    Robber(string name);
    void greet(Robber r);
    int lookat(Robber r);
    void steal(Robber r, int i);
    string yell();
};
```