1. Consider the following `UITableViewCell` constructor:

```
- (id)initWithStyle:(UITableViewCellStyle)style
reuseIdentifier:(NSString *)reuseIdentifier
```

What is the purpose of the `reuseIdentifier`? What is the advantage of setting it to a non-`nil` value?

This is an unique ID number for a single table cell within a UITableView, means this cell can be reused.

Why can a cell be reused? It is because an iphone screen can only show 5 table cells (iphone 5 can show 6 table cells). So when user scroll up a table view, the top table cell will disappear and will be reused and show up at the bottom of screen with a new content (as the app required).

In this case, UITableView will first to find the reused cell. If there is not, it will allocate a new one. Therefore, the advantage is to save effort.

2. What are different ways that you can specify the layout of elements in a `UIView`?

Three ways

1. This is what I did for my sticker table view. Put component/layout in a xib file.
2. Auto layout. Give constrains in storyboard. This was the way that I used for finalizing the layout for all iphone screen size.

3. Set up the layout in code. My teammates use this way by function rect()

3. What is the difference between `atomic` and `nonatomic` properties? Which is the default for synthesized properties? When would you use one vs. the other?

Atomic is a **default** value. When we read something, atomic can guarantee us to give us a value (good data, not a junk memory). If multiple threads including reader and writer pointing to the same variable, atomic will guarantee give us a value either before writer or after writer. Atomic cannot guarantee which one we could get. We must use atomic property when we know my thread is safe.

Nonatomic will return us a garbage data if reader is in the middle of writer. AND nonatomic can make things faster. If I need to consider the speed, I would use nonatomic.

Reference: https://realm.io/news/tmi-objective-c-property-attributes/

4. Imagine you wanted to record the time that your application was launched, so you created a class that defined a global variable in its header: `NSString *startTime;`. Then, in the class' implementation, you set the variable as follows:

```
+ (void)initialize {
    NSDateFormatter *formatter =
[[NSDateFormatter alloc] init];
```

```
    [formatter
setDateStyle:NSDateFormatterNoStyle];
    [formatter
setTimeStyle:NSDateFormatterMediumStyle];
    startTime = [formatter
stringFromDate:[NSDate date]];
}
```

If you then added the following line to the `application:didFinishLaunchingWithOptions:` method in your `AppDelegate`:

```
NSLog(@"Application was launched at: %@",
startTime);
```

what would you expect to be logged in the debugger console? How could you fix this to work as expected?

startTime shows nothing.

The initialize method in Objective-C will be called before the first message is sent to the class.

Use Load instead.

5. Consider the following code:

```
#import "TTAppDelegate.h"

@interface TTParent : NSObject

@property (atomic) NSMutableArray *children;

@end

@implementation TTParent
@end

@interface TTChild : NSObject

@property (atomic) TTParent *parent;

@end

@implementation TTChild
@end

@implementation TTAppDelegate

- (BOOL)application:(UIApplication *)application
  didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions
{
    TTParent *parent = [[TTParent alloc] init];
```

```
    parent.children = [[NSMutableArray alloc]
init];
    for (int i = 0; i < 10; i++) {
        TTChild *child = [[TTChild alloc] init];
        child.parent = parent;
        [parent.children addObject:child];
    }
    return YES;
}
@end
```

What is the bug in this code and what is its consequence? How could you fix it?

Retain cycle.

Parent has a strong reference to children array; children array has a strong reference to parent.

Solution is to change one of reference to be weak reference.

6. Identify the bug in the following code:

```
@interface TTWaitController : UIViewController
```

```
@property (strong, nonatomic) UILabel *alert;

@end

@implementation TTWaitController

- (void)viewDidLoad
{
    CGRect frame = CGRectMake(20, 200, 200, 20);
    self.alert = [[UILabel alloc]
initWithFrame:frame];
    self.alert.text = @"Please wait 10
seconds...";
    self.alert.textColor = [UIColor whiteColor];
    [self.view addSubview:self.alert];

    NSOperationQueue *waitQueue =
[[NSOperationQueue alloc] init];
    [waitQueue addOperationWithBlock:^{
        [NSThread sleepUntilDate:[NSDate
dateWithTimeIntervalSinceNow:10]];
        self.alert.text = @"Thanks!";
    }];
}

@end

@implementation TTAppDelegate

- (BOOL)application:(UIApplication *)application
  didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions
{
```

```
    self.window = [[UIWindow alloc]
initWithFrame:[[UIScreen mainScreen] bounds]];
    self.window.rootViewController =
[[TTWaitController alloc] init];
    [self.window makeKeyAndVisible];
    return YES;
}
```

How could you fix this issue?

There is no guarantee that the block will be executed on main thread when use method addOperationWithBlock.

So it might not show up the label.

Solution is to force the block execute on the main thread

```
[waitQueue addOperationWithBlock:^{
    [NSThread sleepUntilDate:[NSDate
dateWithTimeIntervalSinceNow:10]];
    [[NSOperationQueue mainQueue] addOperationWithBlock:^{
        self.alert.text = @"Thanks!";
    }];

}];
```

reference: https://www.toptal.com/ios/interview-questions

7. What's the difference between an "app ID" and a "bundle ID" and what is each used for?

An app id “is a two-part string used to identify one or more apps from a single development team”. Team ID + Bundle ID.
A team ID is assigned by Apple and uniquely specific a Development Team.

Bundle id is an id composed by “com.yourwebsite.yourprojectname” (a reverse domain name). That used to match the app in xcode to your iTuneconnect account (apple developer account).

8. What are “strong” and “weak” references? Why are they important and how can they be used to help control memory management and avoid memory leaks?
Strong reference means an “Ownership”. We control the lifecycle of the objects that assigned to strong reference. All objects will not be destroyed (by compiler) as long as we assign a strong reference to those ojbects. When we order the compiler to destroy these (set them to be nil), they will be destroyed.

Weak reference means we are not controlling this object. We can access its properties when someone else holds a/some strong reference(s) to it .If someone loses the strong reference(s) to this object, we will lose this object too.

Reference: http://stackoverflow.com/questions/11013587/differences-between-strong-and-weak-in-objective-c

The reason we have weak reference is to avoid retain cycle. A retain cycle (a memory leak) happens when two objects pointing to each other by strong reference, therefore, these two objects will never be destroyed by ARC.

So we need to make one of two references to be a weak reference.
Reference:
http://www.informit.com/articles/article.aspx?p=1856389&seqNum=5

9. Describe **managed object context** and the functionality that it provides.

A managed object context represents a single object space in a core data application and it is an instance of NSManagedObjectContext. Its primary responsibility is to manage a collection of managed objects. We use it to create and fetch managed object.

Reference:
https://developer.apple.com/library/ios/documentation/DataManagement/Devpedia-CoreData/managedObjectContext.html

10. Compare and contrast the different ways of achieving concurrency in OS X and iOS.

There are basically 3 ways: threads, dispatch queues, and operation queues.

Threads, disadvantage is that developer needs to dynamically create and adjust the number of threads when the condition changes.

Dispatch queues uses first in first out data structure. So tasks always started in the same order that they are added.

Operation queue “execute tasks in FIFO order and support the creationg of complex execution-order graphs for tasks.”

Reference: http://help-la-ios.blogspot.ca/2016/01/compare-and-contrast-different-ways-of.html

11. Will the code below log “areEqual” or “areNotEqual”? Explain your answer.

```
NSString *firstUserName = @"nick";
NSString *secondUserName = @"nick";

if (firstUserName  == secondUserName)
{
  NSLog(@"areEqual");
}
else
{
  NSLog(@"areNotEqual");
}
```

“areEqual”

IOS compiler will optimize references to string objects that have the same value so that pointer firstUserName and secondUserName will point to the same address and then return “areEqual”.

//Reference for all:
https://www.toptal.com/ios/interview-questions