1, what is Core Date?

Core Data is a framework that is used to manage model layer objects. It has the ability to persist object graphs to a persistent store. Data is organized into relational entity-attribute model. NSUserDefault is typically used to store small bits of data (settings, preferences, etc.). Core Data is used to store a large list of elements.

Core Data:

https://developer.apple.com/documentation/coredata

2. what is managed object context?

First, managed object context is an instance of NSManagedObjectContext. It is the central object in the Core Data stack. It is used to create and fetch managed objects, and to manage undo and redo operations. Although it is allowed to have multiple managed object contexts, there is typically at most one managed object to represent any given record in a persistent store.

3. what are some ways to debugging in iOS?

- NSLog and print functions can be used for output into console.
- Breakpoints can also be used together with the Debug bar and Variables view as an alternative.
- Senior developers often use other tools such as Instruments and Crash Logs instead of the two above.

4. what is MVVM?

MVVM stands for **Model-View-ViewModel**. It is a software architecture pattern for implementing user interfaces.

MVVM is an augmented version of MVC where the presentation logic is moved out of the controller and into the view model. The view model is responsible for handling most, if not all, of the view's display logic.

A common occurrence in MVC is where you have a massive-view-controller (some joke this is what MVC stands for). In order to shrink the size of your view controller and make the logic and readability of your code easier to follow along, the MVVM will be used.

5. what is JSON? What are the pros and cons?

JSON stands for JavaScript Object Notation. According to wiki, it is a file format that uses human-readable text to transmite data objects consisting of attribute-value pairs and array data types.

Pros:

- It is lighter than XML meaning that it can represent the same data in XML in fewer bytes. This makes network transmissions and read/writes faster
- Since it is native to JavaScript, computationally-expensive XSL transformations are not needed in order to extract data

Cons:

- Not as widespread as XML
- Data is not readily streamable and has to be broken up into individual objects
- Can't use comments

6. what is a memory leak?

A memory leak commonly occurs when an object is allocated in such a way that when it is no longer in use or needed, it is not released. In iOS programming, you create certain objects with weak references in order to avoid a strong to strong relationship that creates a retain cycle and a memory leak.

7. How memory management handled on iOS?

iOS uses something called ARC which stands for Automatic Reference Counting. When an object is said to have a strong reference to it, ARC increase its retain count by 1. When the retain count of an object reaches 0, the object will typically be deallocated if there are no more strong references to it. Unlike garbage collection, ARC does not handle reference cycles automatically.

ARC:

https://docs.swift.org/swift-book/LanguageGuide/AutomaticReferenceCounting.html

8. what's the difference between public and open? Why is it important to have both?

Open access imposes limitations on class inheritance. Classes declared with open level access can be subclassed by modules they are defined in, modules that import the module in which the class is defined, and class members as well. While this sounds similar to the public access level defined in Swift 2, there is a small difference. In Swift 3, the meaning of public access level means that classes declared public can only be subclassed in the module they are defined in. This includes public class members which can be overridden by subclasses defined int he module they are defined in.

Some classes of libraries and frameworks are not designed to be subclasses. For example, in the Core Data framework, Apple states that some methods of NSManagedObject should not be overridden. To prevent any unexpected behavior that may result from overriding those methods, Apple declares those methods public rather than open. As a result, those methods are not marked as open for developers to override.

Access Control:

https://docs.swift.org/swift-book/LanguageGuide/AccessControl.html

9. what's the difference between var and let?

var is a variable that can be changed while *let* denotes a constant that cannot be changed once set.

10. what's the difference between a struct and a class?

The main difference to note is that structs are value types (stored on stack) while classes are reference types (stored on heap).

Classes have capabilities that structs do not:

- Inheritance enables one class to inherit the characteristics of another.
- Type casting enables you to check and interpret the type of a class instance at runtime.

- Deinitializers enable an instance of a class to free up any resources it has assigned.
- Reference counting allows more than one reference to a class instance.

Structures and Classes

https://docs.swift.org/swift-book/LanguageGuide/ClassesAndStructures.html

11. what is GCD and how is it used?

GCD stands for Grand Central Dispatch. According to Ray Wenderlich, it offers the following benefits

- Improving your app's responsiveness by helping to defer computationally expensive tasks and run them in the background.
- Providing an easier concurrency model than locks and threads and helps to avoid concurrency bugs.
- Potentially optimize your code with higher performance primitives for common patterns such as singletons.

In other words, GCD provides and manages queues of tasks in the iOS app. This is one of the most commonly used API to manage concurrent code and execute operations asynchronously. Network calls are often performed on a background thread while things like UI updates are executed on the main thread.

12. What is the purpose of unit/UI testing? What are the benefits?

Unit/UI testing are the basic of test-driven development. This development approach allows you to codify requirements for your code before you implement it. Unit tests are important to make sure that code meets its design and requirements and behaves as expected. Parts of the program are segregated and tested to ensure that individual parts are working correctly.

13. what is autolayout?

Auto Layout is used to dynamically calculate the size and position of views based on constraints.

14. What is the difference between viewDidLoad and viewDidAppear? Which should you use to load data from a remote server to display in the view?

viewDidLoad is only called when the view is loaded (after loadView is called). viewDidAppear, on the other hand, is called everytime the view appears on the device.

If the data is static for the most part, it can be loaded in viewDidLoad and cached. But if the data is dynamic and likely to change often, it is preferrable to use viewDidAppear. In both instances, data should be loaded asynchronously on a background thread to avoid blocking the UI.

15. What is the reuseIdentifier for?

The reuseIdentifier indicates that cells for a UITableView (or UICollectionView) can be reused. UITableView maintains an internal cache of UITableViewCell with the appropriate identifier and allows them to be reused when dequeueForCellWithReuseIdentifier is called. As a result, this increases performance of UITableView since a new view does not have to be created for a cell.

16. 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”)
}
```

The code will output “areEqual”. While one might think this is obvious, it’s not. Here’s why:

Comparing pointer values equates to checking if they point to the same object. Pointers will have the same value if and only if they actually point to the exact same object (whereas pointers to different objects will not have the same value, even if the objects they point to have the same value).

In the above code snippet, firstUserName and secondUserName are each pointers to string objects. One could easily assume that they are pointing to different string objects, despite the fact that the objects that they point to both have the same value. However, the iOS compiler optimizes references to string objects that have the same value (i.e., it reuses them rather than allocating identical string objects redundantly), so both pointers are in fact pointing to same address and the condition therefore evaluates to true.