Objective C

1) Local and Remote notification

A) Local Notification

a) Local Notification : Exists only in ios
UILocalNotification -

b) Push Notification

a) APNS is the gateway for Push notifications.

b) Each device establishes an persistent IP connection with the APNS.

c) Provider (server) sends data to APNS and APNS delivers it to device.

d) SSL certificate needs to be generated in development center - which is limited to single application

e) SSL certificate is limited to one or two environments (one for dev and one for prod)
(IP and port are also different - From provider to APNS)
(From provider to APNS - Binary interface + TCP connection)

f) Feedback service - Maintains a per-application list of devices for which there were failed-delivery attempts.

```
If (app is in foreground)
     application:didReceiveRemoteNotification:
     or application:didReceiveLocalNotification:
else
     launchOptions
```

Device token is unique to Device. The application in a device to which push to be delivered will be decided on Bundle id.

* Feedback service stores the latest push notification.
* Max 64 push notifications can be delivered at a time.

2) Notification Center

KVO
Add observer on specific key and broadcast the notification/message.

3) Core Data & Sqlite

A) Sqlite3:

a) Dumping tables = .dump

b) To redirect the output of a file = .output

c) CREATE = To create a table, ALTER = To alter the table, DROP = To drop the table

d) LIMIT -> To limit the number of items

e) ORDER BY -> ASC & DESC

f) HAVING
Ex:
SELECT sum(OrderPrice) AS Total, Customer FROM Orders GROUP BY Customer HAVING sum(OrderPrice)>1000;

g) CONSTRAINTS (NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, DEFALUT)

h) PRIMARY KEY:
i) The primary key uniquely identifies each record in a database table.

ii) Primary key becomes Foreign keys in other tables, when creating relations among tables.

iii) Primary Key may be NULL in sqlite (which is not supported in any other database)

i) FOREIGN KEY:
i) Foreign key in one table points to Primary key in another table.

j) SQL JOIN - INNER & OUTER

App Integration:

a) Create a .sql file using terminal command (sqlite3 file.sql)

b) Add this file to project (This will act as a schema - not an editable copy)

c) Create an editable copy (using NSFileManager)

d) Sqlite3_open, Sqlite3_stmt, sqlite3_prepare_v2, sqlite3_step, sqlite3_coloumn_int, sqlite3_finalize, sqlite3_close.

B) Core Data

a) NSManagedObjectModel - Like a database schema.

- It is a class that contains definitions for each of the objects (also called entities).

b) Persistent store co-ordinator - Like a database connection.

- This is where we up the actual names and locations of database needs to be connected.
- The managed object contexts will be saved to database through this co-ordinator.

c) NSManagedObjectContext - Like a "scratch pad" for the objects that comes from the database.

- All operations on database will be done on this.

Other terms:

a) NSManagedObject
- Every object that core data stores that is derived from NSManagedObject.

b) NSFetchRequest

- NSEntity description

c) NSPredicate - To add a query for the fetch request.

d) NSFetchResultsController -

- Sort descriptor - How we want our result sorted.

- ControllerWillChangeContent - delegate method - Table view begin updates needs to be called.

e) Faulting - Mechanism core data employs to reduce your application's memory usage.

- A fault is a placeholder object that represents a managed object that has not yet been fully realized, or a collection object that represents a relationship.

- Can be disabled by setting 'setReturnsObjectsAsFaults' to NO in fetch request.

My Understanding… (Need to be checked)

- When we fetch an object the associated object (having relationship with other entity) will be returned as 'faults' and actual associated object will be fetched when we try to access like 'objectForKey'

f) Inverse Relationship
- App recommends that whenever you create a link from one object to other (i,e relationship), you create a link from the other object going back as well.
(i,e Inverse - relationship)

i) Entity
- This can be treated as a table in sqlite.

- This contains either attributes or relationships.

ii) Attribute -

iii) Relationship -

a) to one relationship - One to one relationship
b) to many relationship - One to many relationship
c) many to many relationship - where relationship and its inverse are both one to many relationship.

Relationship deleting rules

Deleting rules - Specifies, what should happen if an attempt is made to delete the source object.

a) Deny - If there is at least one object at the relationship destination, then the source object cannot be deleted.

b) Nullify - Set the inverse relationship for objects at the destination to null.

c) Cascade - Delete the objects at the destination of the relationship.

d) No Action - Do nothing to the object at the destination of the relationship.

Core data migration:

a) Light-weight migration

```
                    NSDictionary *options = @{

NSMigratePersistentStoresAutomaticallyOption : @YES,

NSInferMappingModelAutomaticallyOption : @YES
                                   };
```

b) Manual

c) Custom code.

Actual migration may involve one or more of these techniques.

http://www.raywenderlich.com/12170/core-data-tutorial-how-to-preloadimport-existing-data-updated

http://www.raywenderlich.com/27657/how-to-perform-a-lightweight-core-data-migration

4) In App Purchase - storekit

* Before we code 'in-app purchase' , we have to set them up in iTunes Connect.

* Types --

a) Consumable - This means things you can buy more than once and can be used up. Ex: Extra lives

b) Non-consumable - This means something that you buy once, and expect to have it permanently. Ex: Next level

* Define product id and details in iTunes

* Get these product ids from the backend (just remove hard coded values)
* Call SKProductRequest using these product ids to get the details from the iTunes.
* SKPaymentQueue or Product queue (As an observer) and controls the transaction.

5) Design patterns

- Design pattern is a template for a design that solves a general, recurring problem in a particular context.

* MVC
* delegate, target-action (communication between V and C) - Adapter
* KVO and KVC Notification (Communication between Modal and C)
* Singleton.

6) Protocols and Categories

* Using Protocols we can relate two different objects.
* The object which is going to confirm to that protocol should implement all the methods (required) of that protocol.

* Categories are meant to extend the classes without actually inheriting that class.
* Categories allows you to add methods the an existing class - even to one for which you do not have the source.
* Its not possible to declare a member variable in categories.

By using these 2 we can partially achieve multiple inheritance.

7) Fast Enumeration

9) Class & Instance methods - Difference between static and Class

* Instance methods can be called only using Instance of an object.
* Class methods can be called using class name itself.

static prefix for function name in c and c++ means - Its scope is available only in that file.
static -> Single instance through out the project.

10) Chain responder

11) XML & JSON Parsing

* XML
    - SAX and DOM parser
    - NSXMLParser is sax parser.
    - TinyXMl is DOM parser

* JSON
    - NSJSONSerialization

Favor XML over JSON when any of these is true

a) You need message validation
b) You are using XSLT
c) You messages include a lot of marked-up text
d) You need to interoperate with environments that don't support JSON.

Favor JSON over XML when all of these are true

a) Messages don't need to be validated, or validating their deserialization is simple.
b) You are not transforming messages, or transforming their deserialization is simple.
c) Your messages are mostly data, not marked-up text.
d) The messaging endpoints have good json tools.

http://stackoverflow.com/questions/5615352/xml-and-json-advantages-and-disadvantages

12) Webservice - REST and SOAP

* REST - Representational state transfer.
    - Light weighted
    - Each unique URL is a representation of some object. You can get the contents of that object using HTTP GET, or modify that object using PUT, POST or DELETE.
    - Its supports different data representation (i,e JSON and XML)

* SOAP - Simple object access protocol.
    - SOAP also uses HTTP. Its xml based content and Soap headers and body will be added in the HTTP.
    - It has successful/retry logic built in and

provides end-to-end reliability.

Advantages

- Using SOAP over HTTP allows for easier communication through proxies and firewalls than previous remote execution technology.
- SOAP is versatile enough to allow for the use of different transport protocols. The standard stacks use HTTP as a transport protocol, but other protocols are also usable. (E.g SMTP)
- SOAP is platform independent.
- SOAP is language independent.

Disadvantages

- Because of the verbose XML format, SOAP can be considerably slower.

13) OAuth

* 2.0
* Client ID,
  Client Secret,
  Scope -> specific feature/Module - ex:smsipgw
  redirect URI ->
  Authorize URL ->
  Token URL ->

1) Directly using username and password

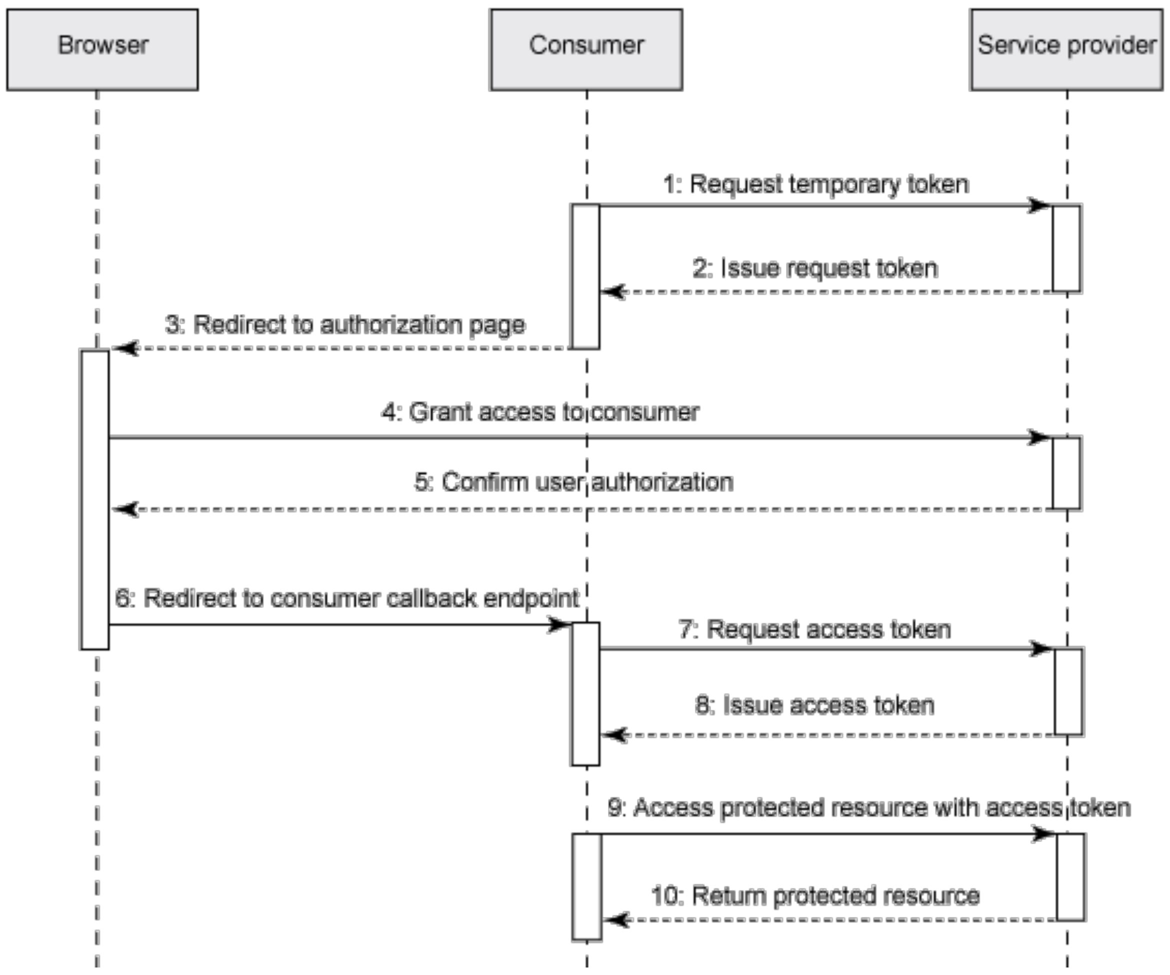
   - Framework takes the responsibility of internal calls and getting access token.

2) Via webview

   - Framework returns prepared URL which app should open in web view - where user will enter the username and password

3) Via external agent

   - Framework will redirect the app to safari


Browser
Consumer
Service provider
1: Request temporary token
2: Issue request token
3: Redirect to authorization page
4: Grant access to consumer
5: Confirm user authorization
6: Redirect to consumer callback endpoint
7: Request access token
8: Issue access token
9: Access protected resource with access token
10: Return protected resource

14) SIP – Overview (Related to Project)

15) Core location & Mapkit

∗ Core location - CLLocationManager, CLLocation, ReverseGeoCoding, Startupdating location, stop updating location.

∗ Map kit -

∗ Geo-fensing - Distance filter or time filter

16) Social network integration

17) HTTP & HTTPS (Rest Kit)

HTTPS = HTTP + SSL (Secure socket Layer)
Encryption just works like codec in sip.

GET:

POST:

PUT:

http://stackoverflow.com/questions/630453/put-vs-post-in-rest

18) Code Signing Identity (Provisioning profiles & P12 files & Certificates) – SSL Certificate detail

19) Developer account details and eligibality

20) ScrollView + Navigation controller + Tabbar (Customization of these views)

21) Core Animation & UIAnimation & Core drawing

22) Differences between Mutable & Non mutable data source

23) KVC and KVO (practical usage)

24) Unit testing

25) Sharing documents between the apps

26) Accessing shared resource in iPhone

27) Tools & Third party frameworks used

28) UIApplication & UiDevice class reference (Application states)

29) ios Architecture

∗ Cocoa touch layer

```
          - Address Book UI
          - UIKit

     * Media layer
          - Audio toolbox
          - Core graphics
          - Core Audio

     * Core services
          - Address book
          - Core foundation
          - Core location

     * Core OS
          - CFNetwork
          - Security
```

30) XIB & Storyboard

31) Memory management in ios
alloc, dealloc, retain, copy, assign, atomic, non-atomic, @syncronized,
strong, weak, unsafe_unretained.

32) NSOperation Queue, GCD and Mulit-threading

33) Regular expressions

34) UIWebview & delegates + Loading java script
-> Redirect url

35) Integrating java script in code

36) Understanding of NSCalender & NSDate & Timezome

37) Some utility methods or class references

38) KeyChain + Plist

39) Run loop & Dynamic loading & Dynamic typing

40) File Management & Accessing different formats of file.

41) Audio/Video Programming

42) Leak/Performance Testing and Instruments.

43) NSCoder and NSCopying

44) Blocks

- This is an extension to Objective C. They allow you to wrap up the chunks of code in self-contained units and pass them around as objects.

45) CFNetwork - Socket programming

46) Extensions - Difference between Categories and Extensions

1) Extensions are mainly used for creating private categories.

2) You can create it for user defined classes (for which definition is there)

47) Exception handling - Signals

@try, @catch, @finally
raise - run time specific
@throw - Compiler specific

48) NSURLConnection + Delegates

How to call ??

Asynchronous

```
NSURLRequest *request = [NSURLRequest requestWithURL:[NSURL
URLWithString:@"http://google.com"]];

// Create url connection and fire request
NSURLConnection *conn = [[NSURLConnection alloc]
initWithRequest:request delegate:self];
```

POST:

```
// Create the request.
NSMutableURLRequest *request = [NSURLRequest requestWithURL:
[NSURL URLWithString:@"http://google.com"]];

// Specify that it will be a POST request
request.HTTPMethod = @"POST";

// This is how we set header fields
[request setValue:@"application/xml; charset=utf-8"
forHTTPHeaderField:@"Content-Type"];

// Convert your data and set your request's HTTPBody property
NSString *stringData = @"some data";
NSData *requestBodyData = [stringData
dataUsingEncoding:NSUTF8StringEncoding];
request.HTTPBody = requestBodyData;

// Create url connection and fire request
NSURLConnection *conn = [[NSURLConnection alloc]
initWithRequest:request delegate:self];
```

Synchronous

```
// Send a synchronous request
NSURLRequest * urlRequest = [NSURLRequest requestWithURL:
[NSURL URLWithString:@"http://google.com"]];
NSURLResponse * response = nil;
NSError * error = nil;
NSData * data = [NSURLConnection
sendSynchronousRequest:urlRequest
                                          returningResponse:&r
esponse
                                                      error:&e
rror];

if (error == nil)
{
    // Parse data here
}
```

Delegate methods

```
- (void)connection:(NSURLConnection*)connection
didReceiveResponse:(NSURLResponse *)response
{
    NSLog(@"Did Receive Response %@", response);
    responseData = [[NSMutableData alloc]init];
}
- (void)connection:(NSURLConnection*)connection
didReceiveData:(NSData*)data
{
    //NSLog(@"Did Receive Data %@", data);
    [responseData appendData:data];
}
- (void)connection:(NSURLConnection*)connection
didFailWithError:(NSError*)error
{
    NSLog(@"Did Fail");
}
- (void)connectionDidFinishLoading:(NSURLConnection
*)connection
{
    NSLog(@"Did Finish");
    // Do something with responseData
}
```

NSURLRequestCachePolicy

```
   An NSURLConnection may be used for loading of
resource data
      directly to memory, in which case an
NSURLConnectionDataDelegate
      should be supplied, or for downloading of
resource data directly to a file,
      in which case an
NSURLConnectionDownloadDelegate is used.
      The delegate is retained by the NSURLConnection
until a terminal condition is
      encountered.  These two delegates are logically
subclasses of the base protocol,
```

NSURLConnectionDelegate.<p>

** The NSURLConnectionSynchronousLoading category adds**

+sendSynchronousRequest:returningResponse:error, which blocks
**   the current thread until the resource data is available or an**
**   error occurs. It should be noted that using this method on an**
**   applications main run loop may result in an unacceptably long**
**   delay in a user interface and its use is strongly**
**   discourage.<p>**

** The NSURLConnectionQueuedLoading category implements**

+sendAsynchronousRequest:queue:completionHandler, providing
**   similar simplicity but provides a mechanism where the current**
**   runloop is not blocked.<p>**

NSRunLoop

- It provides a programmatic interface to objects that manages the input resources like mouse, keyboard, sockets, ports, timers etc.

- Our application can not create or explicitly manage NSRunloop objects.

- Each NSThread object including main thread, has an NSRunLoop object automatically created for it as needed.

C - Refresh

1) Data Structures & Linked list

2) Static concepts with Variables & functions

C++ - Refresh

1) Oops Concepts & Comparing with Objective C

   a) Abstraction

   b) Encapsulation

   c) Inheritance

   d) Dynamic typing

   e) Dynamic loading

2) Polymorphism & Friend funciton understanding

Importants

1) Convertion of nsstring to char *

   [NSString UTF8String]

Questions

My Questions

1) Developer profile creation

   1) Provisioning profiles
   2) Certificates
   3) APNS Enabling

   a) Creating developer account

   - Having Mac download Xcode - Objective C language

   - Enroll to Apple Dev account for distribution ($99/year)

   - Choose an enrollment type (Individual/Company)

   - If enrolling for company need Legal entity Name and D-U-N-S number

      => D-U-N-S == Dun & Bradstreet (D & B) provides a D-

U-N-S number, a unique nine digit identification number for each physical location of your business.

- Create your own account in developer.apple.com

- Register for iOS developer program

- Three roles that can be assigned to Apple developer program - assign certain responsibilities
    - Accepting program agreements
    - Inviting additional members
    - Creating certificates
    - Submitting apps to itunesConnect.
    - Admin or member roles are only available to developers enrolled as an organization.

- Team Agent - The original enrollee - Responsible for accepting all Apple Developer program agreement as well as renewing memberships.
- Admin - They can invite members to the team, assign roles, and access membership resource and benefits.
- Member - Access to development certificates, pre-release softwares and technical support.

b) Provisioning profiles and certificates

- Creating p12 files

a) Keychain Access - Certificate Assistant - Request a certificate from a certificate authority

b) This will create a private key (key size = 2KB & Algorithm = RSA)

c) Login to dev account using Team Agent

d) Certificates & Request certificates & upload the certificate (keychain)

e) Download .cer file from portal

f) Then double click on .cer file and export both key & certificate from keychain access to create .p12 file.

- Provisioning profiles

a) iOS provisioning portal

c) Create APNS Certificates

a) Select app in the app IDs list and click on the settings button in the selected App ID.

b) Development SSL Certificate – Enable push notifications – Creating CSR

c) After that wizard process completes will be able to download SSL certificate. (.cer file)

d) Double click the downloaded SSL file and double click on it. Then in keychain access there should be able to see 'Apple development iOS push services'

e) Then export it (Apple dev iOS push services)

2) AppStore submission

=> Enroll in program –> Provisioning devices –> Create iTunes Record –> Submit app –> Ship app

a) Brief description of the app that iTunes displays to customers
b) An app icon (512x512) ready for upload
c) At least one screen shot of your app ready for upload
d) An internal version number for your submission
e) A bundle ID that you have set to match your App ID

Meta data = Application name + Version number + Primary Category + concise description + keywords + support URL

iTunes Artwork: 1024px x 1024px (required)
iPad/iPad Mini: 72px x 72px **and** 114px x 114px (required)

iPhone/iPod Touch: 57px x 57px **and** 114px x 114px (required)

Search Icon: 29px x 29px **and** 58px x 58px (optional)

Settings Application: 50px x 50px **and** 100px x 100px (optional)

3) Enterprise application details – 299$/year

4) Background modes details

5) Questions on Scrollview, Tabbar view and Navigation controller

6) ios5, ios6 and ios7

iOS5 - iCloud, ARC, Storyboards, Newsstand, Core Image, Twitter framework

ios6 - Apple Map, Facebook integration, Passbook, FaceTime, AutoLayout,

iOS 7 - Control centre - Gives quick access to the controls
Notification Centre - Knows about new mails, missed calls, to-dos. Today column give all things about today (Any birthday, weather report)
Multitasking - Learns when you like to use your apps and update your content before you launch them.
Camera - Square front
Photos - Smart grouping of your photos based on time and place.
iCloud - photo sharing
Airdrop
AppStore - Show a collection of popular apps relevant to your current location.

7) Project related preparation

8) NSThread + GCD + NSOperationQueue

NSOperationQueue

a) Create NSoperationQueue
b) Create NSInvocationOperatoin - pass method to be called.
c) Add NSInvocationOperation in NSoperationQueue
d) Create another NSInvocationOperation add into the queue
e) Both the task will run concurrently.

GCD
a) Create dipatch_queue_t = dipatch_queue_create
b) dipatch_async (queue, ^(void){});

Advantages of NSOperationQueue over GCD

a) Bandwidth-constrained queues that only run N operations.
b) Establishing dependencies between operations.

9) Universal application and Orientation support

```
    - (NSUInteger)supportedInterfaceOrientations {
        return UIInterfaceOrientationMaskLandscapeLeft |
UIInterfaceOrientationMaskLandscapeRight;
    }
```

10) AFNetworking

11) Social network integration

12) C++ concepts

13) Abstraction and Encapsulation

Abstraction : Hiding the implementation detail and showing only that much information which is required to present.

Encapsulation : Binding properties and methods in a single entity.

For ex:

```
    class student
    {
        public:
            int rollNo;
             -(void) method;
        private:
            char name[];
    }
```

Here hiding private information to the external classes is Abstraction.

Binding properties and methods is encapsulation.

14) KVO vs Notification

KVO -> Notifies whenever the value of the property get changed.
addObserver - for key path (self.attribute)
valueForKeypath -

Notification -> Notifies an event (ApplicationwillTerminate) - Can not be achieved through KVO
addObserver -> call method.

Trello

a) How to use exception handling ? When an exception is catched in utility method and wants to throw it to VC ?? How to do that ??
b) How to add security in web services ?? Other than making the url as https ?? Any API's provided by iPhone os ??
c) Why do u say using categories and protocols its not possible to achieve multiple inheritance completely ??
d) How the cross-platform works ??
e) What are the api's are supported in iOS to do complex mathematics ??
f) When do we use SOAP and REST ?? What is the advantages of using SOAP over REST ??

Capzemini

a) How to create an object which is thread safe ??
b) How to launch an application from our app ??
c) Configuration for Universal app ??
d) Difference between mutex and nslock ??
e) NSOperation queue -> code ..
f) URL Scheme ?

CA

a) what is mach o ? format of iOS app executable ??
b) How to find a minimum height of the tree using root node ??
c) 20 numbers, 1 and 2 ?? how u will pick it ??
d) In a array 2 powers are there ?? return false or true ??
e) method swizzling in iOS ??

ServiceMax

a) Is it possible to make one queue to execute the task of another queue ??
b) How to cancel the request in dispath_queue ?? (which is added with dispatch_after)
c) How to find cyclic referencing in instruments ?
d) How u will write your own getter method ?
e) Difference between categories and subclassing ??
f) What is the main difference between objective c and C ??
g) Is it possible to do static polymorphism in objective c ??
h) Why in objective C it is not supported multiple

inheritance ??
i) How does categories actually works ??
j) How does protocol actually works ??
k) How was the mysql connection actually implemented ??
l) How human interface guidelines are different from best practices ?? y apple declared it ??

Accenture

a) In https request, how do you make sure that the response has received from trusted server ??
b) In singleton class, if someone else directly alloc the shared instance object then singleton behavior breaks.. how can we avoid it ??
c) Difference between categories and subclassing ?? When exactly categories are useful ??
d) What is the difference between formal and informal protocols ??
e) What is the difference between waterfall and Agile methodologies ??

Aditi

1) Core data - Key & relationship
2) ios7 features
3) UI View life cycle
4) Deque reusable identifier
5) Multi-threaded concept
6) Blocks vs function pointers.

EF
Cultural

1) Tell me about your last project - business logic
2) Tell me about your hobbies - y do u like it
3) Tell about ur strengths and weakness
4) Introvert or extrovert
5) Tell about our company
6) Y do u want to join our company ?? what u expect from the company, what can company expect from u ??
7) Tell about the interest thing in you ?

Technical

1) 1UIVC -> 2UIVC
     How 2 UIVC can call a method in 1UIVC ?
2) Repeat the same with 3 UIViews
3) Repeat the same 3 UIViews and try to remove the 2 subview

from the parent view.
4) How to swap 2 numbers without using third variable
5) What is categories ?? can we create categories for custom object ?? How its different from subclassing ??
6) what is protocol and delegates ?? when do we use it ??
7) What actually delegate means ??
8) What are the 3 main components in core data ??
9) If a method is taking too much time to execute and come out .. what may be the reason ??
10) What is multi-threading ?? From when multi-threading is there in iOS ??
11) What is NSOperationQueue and GCD and its difference ??
12) NSOperation queue performance is very high comparatively gcd ?? how ?? (Wrong it seems)
13) What is the difference between put and post ??
14) How can u mention the response type in the request ??
15) what is the difference between json and xml ?? which one u choose most of the time and why ?? what is the pros and cons over each other ??
16) What you would like to suggest to the server to improve the web service performance ??
     Zip - What kind of zip (gzip) - what that means ?
17) Instruments - Name the components which are used to improve the performance ??
18) How do u find where exactly the leak has happened in the code using instruments ?? what is the component u use it for ??

BSC

1) What is the difference between xml and json. when do we choose what ??
2) What is Geo fencing ??
3) What is MDM and BYOD ??

Altimetrik

1) How to add 2 tableviews in a single vc ??
2) How to set cookies in NSURLConnection ??
3) Cryptography concept ??
4) How to implement cacheing ??
5) AFNetworking- Synchronous and Asynchronous calls ??
6) How to open PDF file ??
7) Where these PDF files will be stored ?? and How u r cacheing it ??
8) Core data ??
9) Sqlite ??
10) Project related questions ??

http://iphonematerials.blogspot.in/2012/11/ios-interview-questions-and-answers.html

http://nagamuraliiphonetutorialsblog.blogspot.in/2013/03/1.html

http://iosbricks.blogspot.in/2012/12/very-typical-ios-interview-questions.html

http://rambabumutchu.blogspot.in/2012/11/ios-latest-interview-questions.html

http://way2ios.com/development/ios-development-2/ios-interview-questions-for-experience/

http://sivasankariphone.blogspot.in/2013/01/ios-interview-questions.html

http://ashishjindal-ashu.blogspot.in/2013/04/ios-interview-frequently-asked-question.html

http://share.pdfonline.com/1a0fdf53754243dd95f3353efc6077bc/IOS%20INTERVIEW%20QUESTIONS.htm

http://www.sawaal.com/ios-interview-questions

****

http://abhijeetbargeios.blogspot.in/

http://www.200monkeys.com/index.php/2012/12/23/modern-ios-interview-questions/

http://iphone-rahulvarma.blogspot.in/2012/07/ios-interview-questions-part-2.html

http://sivasankariphone.blogspot.in/2013/01/latest-ios-interview-questions.html

http://spvarma.blogspot.in/2012/05/ios-interview-questions-part-1.html http://jobs2ap.blogspot.in/2012/06/iphone-interview-questions-and-answers_8286.html

Tough Questions

1. How do you handle asynchronous networking?

2. Have you ever worked with multi-threaded Core Data?
* How was it?
* What approach did you use?

3. Have you ever worked with Core Animation?
* Can you tell me about that?
* What kind of animations? (grace notes for an app versus canned table view insertion or bulk affine transforms)

4. Have you ever worked with Core Graphics? (answer for Mac devs should be a resounding "HELL YES I DID")
* What kinds of things did you do?

5. Have you ever filed any verified issues against Apple frameworks on radar?
* Can you describe them to me?
* Better still, can you point them out on open radar?

6. Compare and contrast NSNotification vs KVO.
* When would you use one or the other? What are the implications of using them?

7. Have you ever worked with NSOperationQueue?
* What did you use it for?

8. Please tell me about your Core Text experience.

9. How would you use NSURLConnection on a background thread?

10. What kinds of issues should someone be aware of when working with blocks?

11. Have you ever built any custom frameworks or libraries?
* What's your preferred method of working on an application in parallel with a library?

110 Favor XML over JSON when any of these is true:

- You need message validation
- You're using XSLT
- Your messages include a lot of marked-up text
- You need to interoperate with environments that don't support JSON

Favor JSON over XML when all of these are true:

- Messages don't need to be validated, or validating their deserialization is simple
- You're not transforming messages, or transforming their deserialization is simple
- Your messages are mostly data, not marked-up text
- The messaging endpoints have good JSON tools