**Question 1. (C)**

What is the C type used to work with objects in Obj-C?

1. int
2. structure
3. **pointer**
4. array

**Question 2. (General)**

What is the id type?

1. **A generic C type that Objective-C uses for an arbitrary object.**
2. A memory address type
3. A type to hold serialized objects.
4. The type used for Classes
5. Noe of the above.

**Question 3. (Internals)**

What is a SEL?

1. A selection
2. **The C type of a message selector**
3. The C type of a class
4. A pointer to a method
5. None of the above

**Question 4. (C)**

What will be the output of the following code?

static int  
 a (void)  
 {  
 printf ("a\n");  
 return 0;  
 }  
 static int  
 b (void)  
 {  
 printf ("b\n");  
 return 1;  
 }  
 static int  
 c (void)  
 {  
 printf ("c\n");  
 return 2;  
 }  
 int main (int argc, const char \*argv[])  
{  
 printf ("%d %d %d", a (), b (), c ());  
 return 0;  
 }

1. a  
    b  
    c  
    0 1 2
2. a  
    b  
    c  
    2 1 0
3. **c  
    b  
    a  
    0 1 2**
4. c  
    b  
    a  
    2 1 0
5. None of the above

**Question 5. (Categories)**

What class specifiers are supported?

1. FINAL
2. STATIC
3. FAST
4. ITERATIVE
5. **There is no such thing as class specifiers.**

**Question 6. (Exception)**

Can an exception caught in @catch be re-thrown?

1. **Yes**
2. No

**Question 7. (Protocols)**

Which of the following creates a class that conforms to a protocol?

1. @interface ClassName [ProtocolName]
2. **@interface ClassName <ProtocolName>**
3. @interface ClassName < ProtocolName
4. @interface ClassName::ProtocolName
5. @interface ClassName(ProtocolName)

**Question 8. (C or General)**

What is #import

1. A namespace import rule
2. A namespace definition
3. A recursive include
4. **C preprocessor construct to avoid multiple inclusions of the same file**
5. None of the above

**Question 9.(Memory Management)**

How do you free an object?

1. free(obj)
2. [obj dealloc]
3. **[obj release]**
4. [obj free]
5. None of the above

**Question 10. (Messaging)**

What can you use to avoid the msgSend function overhead?

1. SEL
2. **IMP**
3. You can’t use anything
4. None of the above

**Question 11. (Messaging)**

A class can have two methods with the same name, but with different argument types?

1. True
2. **False**

**Question 12. (Messaging)**

What is true regarding messaging?

1. Messaging is static and messages are replaced by function call at compile time
2. **Messaging is fully dynamic, which means you can compile some code that sends a message to a class that doesn’t implement it, and add a category later, in a dynamic library for example**
3. Messaging is only a syntax sugar to call functions
4. None of the above is true

**Question 13. (Categories)**

What happens if two categories define methods with the same names for the same class?

1. The code won’t compile
2. **At runtime, either method will be called**
3. A runtime exception will be thrown
4. None of the above

**Question 14. (Internal)**

What is the Obj-C runtime?

1. **A C library**
2. A compiler
3. A language
4. A dynamic loader

**Question 15. (Memory Management)**

If you need to allocate custom memory, in which method will you do so?

1. +alloc
2. –alloc
3. +init
4. **–init**
5. None of the above

**Question 16. (C)**

What is true regarding C functions inside .m files?

1. **They can contain Obj-C code**
2. They are slower than if in a .c files
3. They cannot use C libraries
4. **They can be static**

**Question 17. (Best Practices)**

Which of the following is the fastest?

1. @synchronized
2. **Explicit locking**
3. Condition locking
4. Mutex implicit locking

**Question 18. (Categories)**

As categories can’t have instance variable, what class could you use to implement a full class only with categories?

1. NSArray
2. **NSMutableDictionary**
3. NSSet
4. None of the above

**Question 19. (General)**

What does Obj-C not support?

1. Instance variable
2. **Class variables**
3. Static variables
4. Automatic variables

**Question 20. (Memory Management)**

What happens if you release an unretained object twice?

1. Nothing, too many releases are handled correctly
2. **Undefined behaviour**
3. MemoryException is raised
4. None of the above

**Question 21. (Categories)**

What is a category?

1. A namespace
2. A category is a way to add instance variables to a class which already exists
3. A category is a group of classes
4. **A category is a way to add methods to a class which already exists**
5. None of the above

**Question 22. (Protocols)**

A class can conform to only one protocol?

1. True
2. **False**

**Question 23. (General)**

In which version of Objective-C did the fast enumeration system appear?

1. **2.0**
2. 1.5
3. 1.0
4. 3.0

**Question 24. (General)**

What is true regarding strings?

1. C string literals are automatically mapped to objects
2. **C string literals can be used in Obj-C**
3. Obj-C strings are not of static storage
4. Obje-C strings are like C strings

**Question 25. (C)**

In Obj-C 2.0, what do the fast enumeration protocols rely on to provide fast Enumerations?

1. **C arrays**
2. Java Vectors
3. Ruby hash
4. Obj-C Array
5. None of the above

**Question 26. (General)**

What is nil?

1. **The null object**
2. The null class
3. It doesn’t exist
4. None of the above

**Question 27. (General)**

Which C feature is not supported in Obj-C?

1. Bitfields
2. Compound literals
3. Structures
4. C arrays
5. **Support is compiler dependant**

**Question 28. (Best Practices)**

What type of variable do you need to use to implement singletons?

1. **static**
2. auto
3. const
4. volatile

**Question 29. (C)**

What can be linked to an Obj-C program without any particular process?

1. **C libraries**
2. Java jar files
3. **C++ libraries**
4. scripts
5. Executables

**Question 30. (General)**

What is true regarding @protected?

1. **The instance variable is accessible within the class that declares it and within classes that inherit it**
2. The instance variable is accessible everywhere
3. The instance variable is accessible only with the class that declares it.
4. This is analogous to private\_extern for variables and functions. Any code outside the class implementation’s image that tries to use the instance variable will get a link error
5. None of the above

**Question 31. (Memory Management)**

How do you allocate an object?

1. MyClass \*obj = malloc(sizeof(MyClass));
2. **MyClass \*obj = [MyClass alloc];**
3. MyClass \*obj = alloc(MyClass);
4. MyClass \*obj = [MyClass new];
5. None of the above

**Question 32. (Best Practices)**

When using the garbage collector, which method, that is normally called without the collector, is not called on your objects where they are collected?

1. free
2. **dealloc**
3. destroy
4. uninit

**Question 33. (Messaging)**

A method can be tagged to be called only by a specific class and its subclasses.

1. True
2. **False**

**Question 34. (General)**

What does the following imply?

Worker \*ceo = [[Worker alloc] init];  
ceo->boss = nil;

1. **That the ceo object is statically typed**
2. That the boss instance variable is declared @protected
3. **That the boss instance variable is declared @public**
4. That the ceo is in fact a structure
5. This code is not correct

**Question 35.**

What is true regarding @public?

1. It doesn’t exist in Objective-C
2. **It breaks encapsulation**
3. It can be used only on sigleton objects.
4. None of the above

**Question 36.**

What comments are supported in Obj-C?

1. **//Line comments**
2. **/\* Block comments \*/**
3. # Line comments
4. ; Line comments
5. –[[ block comments ]]

**Question 37. (Memory Management)**

Is the following code a correct allocation?

MyClass myObj;  
[&myObj aMessage];

1. Yes
2. **No**

**Question 38.**

What is not supported in Obj-C?

1. Recursive method call
2. Variable argument count to method
3. Byte manipulation
4. **Method argument default value**
5. None of the above

**Question 39. (Exceptions)**

What are @try and @catch?

1. Exception handlers
2. Exceptions
3. **Exception keywords**
4. All of the above

**Question 40. (Messaging)**

Which of the following is false?

1. Method lookup is done at runtime
2. **When a method is called, the send is automatically available as the sender variable, like self or super**
3. Messages can be sent to nil
4. Methods in static libraries must be present at link time

**Question 41. (General)**

What is the default visibility for instance variables?

1. @private
2. @package
3. @public
4. **@protected**
5. None of the above

**Question 42.**

Can a method be declared to accept a variable number of arguments?

1. **Yes**
2. No

**Question 43. (Memory Management)**

What is an autoreleased object?

1. A C Object
2. A static object
3. An object that is garbage collected
4. **An object that will be released when the current AutoreleasePool is deallocated.**
5. None of the above

**Question 44. (Categories)**

Which of the following can be inherited?

1. **Categories**
2. **Protocols**
3. **Classes**
4. None of the above

**Question 45. (Best Practices)**

Which of the following is not recommended?

1. Adding an (id)sender argument to methods
2. **Using static variables inside methods**
3. Using C code inside Obj-C methods
4. None of the above

**Question 46. (General)**

In which version of Objective-C did the properties system appear?

1. 3.0
2. 2.5
3. **2.0**
4. 1.5
5. 1.0

**Question 47. (Exceptions)**

Which of the following does not happen when you throw an exception in a @synchronized block?

1. **The object is deallocated**
2. The object is unlocked
3. An exception is thrown
4. None of the above

**Question 48. (Categories)**

What can you do with categories?

1. Add instance variables to a class without subclassing it
2. **Add methods to a class without subclassing it**
3. **Override methods of a class without subclassing it**
4. None of the above

**Question 49. (Internal)**

What is the isa variable in objects?

1. Object size
2. Object memory footprint
3. **Object class identification**
4. Object serial number
5. None of the above

**Question 50. (C or General)**

Can you send messages to nil?

1. **Yes**
2. No

**Question 51. (General)**

What’s the difference between copy and deepCopy?

1. They are the same
2. deepCopy doesn’t exist
3. **copy creates a copy at the first level, while deepcopy copies the instance variables**
4. copy creates a proxy object, while deepCopy allocate a new object.
5. None of the above

**Question 52. (Exceptions)**

How do you throw an exception?

1. raise Exception
2. **@throw exception**
3. RAISE exception
4. THROW exception
5. None of the above

**Question 53. (Exception)**

What is a @finally block?

1. A block that is executed when the program quites
2. A block that is executed within a dynamic library when it’s unloaded
3. **A block of code that is run whenever an exception is thrown or not**
4. None of the above

**Question 54. (C)**

What can be used as Object instance variables?

1. **int**
2. **structures**
3. **pointers**
4. **unions**
5. None of the above

**Question 55. (Protocol)**

What is a protocol?

1. A class that uses functions instead of methods
2. A method signature
3. ~~A class signature~~
4. **An interface without an implementation**
5. None of the above

**Question 56.**

How do you include the root “Object” class?

1. #include <Object.h>
2. #include <objc/Object.h>
3. #include <Object/Object.h>
4. #include <ROOT.h>
5. **It depends on the compiler**

**Question 57. (Protocol)**

Which of the following declares a protocol?

1. @proto ProtocolName
2. protocol ProtocolName {};
3. **@protocol ProtocolName**
4. @interface <ProtocolName>
5. @interface ProtocolName::Protocol

**Question 58. (Internals)**

What is an IMP?

1. A special type used for computation
2. An alias for SEL
3. A preprocessor directive defined to the implementation name
4. **The C type of a method implementation pointer**
5. None of the above

**Question 59. (Protocol)**

Protocols are like classes; they can inherit

1. **True**
2. False

**Question 60. (Memory Management)**

Which of the following is incorrect?

1. [self release]
2. [super release]
3. **[AClass release]**
4. They are all correct