CS65 Homework 2 written questions

Peng He

1. What are your (min 3, max 5) favorite sources of technical information on the Internet, especially related to the course? What is a downside that you've experienced with each?

- Stack-overflow
- Google
- Apple MemberCenter
- Beginning iPhone Development with Swift Exploring the iOS SDK-Apress(2014)
- Swift in 24 Hours, Sams Teach Y - BJ Miller
- Group gmail in the future.

Downside:

For the first two, they are search engine based and have a huge mount of questions, by filleting keywords can quickly look into the thing I want but sometimes **hard to describe the issue** and other people's problem may not the one you want.

For the Last three, document based resources, to handle these book, the first thing I need is to know where is the problem and without reading or getting into swift for a deep look, it's hard to tell what problem is, where should I look up. Furthermore is the reading, and I sometime don't understand what going on in the book.

9. What goes wrong when the IntCacheable protocol declaration of LimitedIntCache is left out? How about Printable? How are these two types of mishaps fundamentally different?

The IntCacheable will provide two methods :{get, set} this general benefits us a cache function for the class “LimitedIntCache”.

If we left out the IntCashable, Int type will not be cached.

The Printable make this class function is able to print something out when called by println().

11. Why is maxSize declared as a let but the other class member declared as var?

The maxSize should decide the dimension of a cache array and should not be changed once it’s initialized. The other vars are the cache array, this actual increase/decrease in the run time, and EntryAge, of course it’s different with different entry.

13. The subscript function is defined to return an Optional Int. What specific fact(s) can you deduce about the current state of the cache if nil is returned for a particular key k?

Since the nil is returned, we can deduce that the value associate with key ‘k’ is no longer exists, therefore, the entry has been wiped out by lack of cache memory.

16. There are three ways of managing memory for objects that are allocated and then no longer needed:

- Manual free & release
- Garbage collection
- Automatic Reference Counting (ARC)

CITATIONS ABSOLUTELY REQUIRED EVEN IF YOU ALREADY KNOW BY PRACTICE. Write 4 sentences for each:

a) A major language that uses it as its principal or only method, besides Swift

The Objective-C uses ARC but the programer needs to insert `retain` and `release` to allocate and relocate the memory. Like C++, the program can't call `retain` function but it is call automatically, as well as the `release`. The programer needs to overwrite these two functions in order to get them work with the procreate program. If a `release` is not correct, a memory leak may happen.

The Java use Garbage collection.

The embedded C or assembly language may use manual memory management.

b) How essentially each one works, especially the division of responsibility; between the language runtime and the programmer;

I think the major responsibility is will a programmer care or not care about the memory management (short for MM) in a runtime.

First, I want to talk a little bit on MM and runtime. The storage of a program when it runs is usually distributed on a stack or a heap. Furthermore, a type variable may like on the stack and an instance or reference to a class may likely on a heap. So the MM takes the role of when a instance is no longer needed or has reference to any other codes, so it dies, then the MM will free the space it used and make the memory available for other codes.

Back to top, the Java uses Garbage Collection (short for GC), which a programmer do not care about the memory because the GC will runs a thread in background and keep tracking the reference number of a instance in memory. If the instance has no related reference, GC will collect the memory. On the other hand, the ARC has no background to supervise the garbage memory. So the programer have to tell the OS when an instance is no longer needed and please free this for me. For example, set an instance of a class to `nil` in swift, may, free

the memory if it's a weak connection. Finally, the manual free usually happen on a embedded chip and as my understanding, this will directly manipulate the memory with it's address. In this case, we don't have a stack or heap (Virtual Memory) because we have a very limited storage space and you want everything is fix as well as under your control directly instead of a OS.

c) A major disadvantage, relative to the others;

A con of ARC is sometimes it's not very efficient. It tracks the counting number of reference and when a context switch happens, a lot of things need to be update. There is not really bad things against using the other memory management methods.

d) A major advantage.

I would like to say, ARC is fast. It frees an instance immediately and do not wait for a command circle. Comparing to Garbage Collection, ARC is done in compile time, so no runtime background resource consumed; ARC is quick response, no long time pause will may performed when free a large heap memory (this is because the GC runs by program circle, it's more likely to have a large instance dies in the meantime and GC frees the space will take a long pause).

Citations

- https://en.wikipedia.org/wiki/Automatic_Reference_Counting
- https://developer.apple.com/library/mac/documentation/Swift/Conceptual/Swift_Programming_Language/AutomaticReferenceCounting.html
- https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)
- https://developer.apple.com/library/mac/documentation/Swift/Conceptual/Swift_Programming_Language/AutomaticReferenceCounting.html
- http://stackoverflow.com/questions/8760431/to-arc-or-not-to-arc-what-are-the-pros-and-cons
- http://rypress.com/tutorials/objective-c/memory-management