**1. Imagine asking a friend to meet you at a restaurant. What instructions would a computer program need that your friend would not in order to understand how to get there?**

While both my friend and a computer program would need referential instructions, the computer program would need explicit instructions that define driving, turning, driving laws, and a number of other things.  Both would need instructions about street names, which way to turn on the streets, and possibly some other items.  With my friend, he would be able to find the restaurant even if only given the street names and the direction to turn.

On a high level, the program would need definition on what turning means - turning left and right would have to be defined as angles.  In addition, driving would have to be defined as movement forward for a certain amount of time at a certain speed for the program to understand where it is going.  On a lower level, there are far more things for the program to understand, such as traffic laws - stopping at red lights, merging with traffic, etc - that a human would not need because they already have the understanding.  To say "the 4th right" implies having a counter of streets which is an organizational hierarchy of information that humans understand but a program would have to be given a definition to account for some items.

**2. What is the difference between source code and machine code? What does the CPU do? Where are instructions and data stored?**

Source code is statements that programmers write in different programming languages such as Java, C++, or Python.  None of these languages are readable by a CPU, so they are either compiled or interpreted by the CPU into machine code where it is then executed.  Machine code is the code that CPUs can understand and execute directly, but is next to unreadable to humans.  The instructions and data are stored on the memory - RAM - as it is executed by the CPU.

**3. What is an IDE and why is it useful?**

IDE stands for integrated development environment, which is a user friendly way to write source code.  Though source code can be written on a plain text editor, it is suspect to errors, typos, and other issues.  IDE's are great because they format code by color to help the user easily distinguish between different types of source code - methods, functions, etc.  In addition, they often come with the SDK of a specific programming platform and can be used to compile and run programs, which includes logging bugs.  They can be used to debug, run tests, and a number of other developer helpful items.

**4. What kind of language is Objective-C? Why do you think we use it for making device specific applications? Why wouldn't we want to use an interpreted language for making iPhone and iPad apps?**

Objective-C is a lower-level language than most other common programming languages.  Though not as close to assembly as C, Objective-C is still a lower level language than C++, Java, Ruby, and Python.  Higher level languages are good for sharing across platforms and hardware configurations.  However, since iOS is only run on Apple devices, we do not have to worry about cross-platform or hardware compatibility since it's all streamlined.  At the same time because it is closer to assembly, it is faster to run.