4: Type conversion is the conversion of data types without explicit user casting. Type casting is also when the user explicitly forces a conversion, may or may not be compatible. When its a smaller to bigger data type, the conversion is automatic. Explicit conversion can reverse order from big to small data types. In case of promotions, java automatically promotes data types into ints when evaluating. If any data type is of a float, long, or double data type, java automatically automatically promotes all the data types into it. Attempting to store a larger value in a smaller data type can result in data/precision loss.

6: Memory access with pointers are usually more efficient than subscripting through an array. This is because in compile time, sometimes at large values the compiler takes a while to convert the subscripts to exact memory locations. Having it already in pointer form gives the compiler one less step to access memory.

7: From the first test, it tested adding a variable with a function that added 3 to it, which resulted in 23. The next test was running the function first, which resulted in the same answer, as there are no pointers, it kept the original value of 10 for the rest of the statement. Third test was having x++ + method, which resulted in 24, which is the same one as the first test but adds 1 after. The 4th test, is method first and ++x, which resulted in 24 as well, beacuse method ran first and then added 1 to x. The last test is ++x and then the method, which results in 25, because it does the ++x before the method.