Preparing People to Lead Extraordinary Lives
LOYOLA UNIVERSITY CHICAGO
DEPARTMENT OF Computer Science

COMP 170 Top Ten Things to Know

General / overall topics

Turning problems into programmed solutions → algorithms; using an Integrated Development Environment (IDE) to create and manage solutions. Being able to write, compile and debug programs on your own. Correct compile and run time errors successfully. Being able to read code written by others.

1. Be able to distinguish variable / type and class / instance

Understand types, how to name a variable and give it appropriate type. Be able to define and use classes (both built in and programmer defined.)

A class is an abstract representation or model (class aka type, abstract type, factory for objects). A class has field definitions (or attribute definitions) and operations (or methods) to implement its behaviors.

An instance (aka class instance, object) is an actual, specific, concrete thing and has specific field values (or attribute values). All instances of a class share the same operations and attribute definitions.

2. Know syntax rules and style with careful precision

Learn all the picky rules of C# syntax and use them correctly. Key items: placement of ";", use of case ("ClassSchedule" vs. "classSchedule"), distinguishing declarations and executable statements, use of "(" and ")".

Briefly, "End each statement with ; and always match up () and {} properly." Understand what happens when statements are executed and when that takes place (compared to declarations for the compiler).

3. Use conditional control structures fully and correctly

Construct programs with "if…", and "if…else". Understand nesting of control statements and blocks ("{" and "}").

Briefly, "Nesting, blocks, scope of variable names, no dangles."

4. Construct loops with control structures

Use "while", "do… while", "for", and "foreach" control structures to implement algorithms successfully. Understand how to use "break" and "continue" with loops. Understand when to use different kinds of loops.

Briefly, "Avoid never-ending loops, avoid off-by-one problems."

5. Use the primitive types and operators appropriately

Understand the use and limitations of the C# value types and “string”. Know the most useful arithmetic, relational, and logical operators and use them in expressions.

Briefly, “Understand integer vs. decimal division, casting, and precedence.” Understand how to use the + operator with strings as well as numeric values. Understand the “sizes” of various numeric types and when casting is required to assign one to another.

6. Understand and create functions and methods

Define and use static functions and instance methods correctly, including formal parameters and return values. Know when and how to use “static”. Understand the return statement. Distinguish between formal parameters (aka parameters) and arguments of methods. Understand using local variables. Be able to look up, understand, and use built in functions and methods.

Briefly, be able to construct a program with a good separation of concerns – dividing it into subprograms appropriately.

7. Learn how to create OO Classes

Define and use classes and demonstrate a full understanding of encapsulation and access modifiers “private”, and “public”.

Learn about method overloading and overriding. Learn how to pass object references as arguments to methods. Be able to implement multiple classes that interact or build upon each other. Understand what is happening in a nested stack of method calls.

Briefly: be able to design well-structured classes using the techniques of Object Oriented design.

8. Understand the OO interface concept and usage

Know how to define classes and methods that implement interfaces. Be able to define your own classes that work with existing system interfaces.

9. Use arrays and Collections to organize information

Be able to declare and use “array”, possibly two dimensional arrays, and arrays of complex objects to implement algorithms. Understand how to find things in arrays. Learn how to pass arrays as arguments to methods and what that means.

Be able to declare and use List<T> as a “growable array” and Dictionary<TKey, TValue> to associate keys with values. Understand how to “index” into both of these Collections and learn about some of their key methods (like Add) and Properties (like Count).

10. Perform simple Input / Output processing

Learn how to do basic input and output statements including formatted I/O. Learn how to interface with a user at a “Console” and with files in the OS file system.

Note: This Top Ten list is intended to be a useful guide to the course and to help you study and review appropriately. It is not a complete list of everything that you may need to know. It is not a complete list of what will be on exams.

Version 5.0; Clean up for use with assessment activities. Changes to match late objects approach including separating functions and methods. General section enhanced and moved to top. Inheritance, Switch statement, enum type removed. Fall 2014 by Dr. W L Honig and Dr. R H Yacobellis.
Version 4.0: Major revision by Dr. Yacobellis, 2013. Revised to conform with online text using C# vs. Java.
Version 3.0: Updated by Dr. Yacobellis, 2010. Created for COMP 170 / ISOM 370, Summer 2010.
Version 2.0 & 2.1: Updated by Dr. Honig (additional material and minor changes)