

## COMP151 (CS1) Course Competencies

1. (***Understanding and Evaluating Programs***) Alone or as part of a team, be able to evaluate a given (multi-function, Python) program to determine its overall purpose, if unknown, and evaluate its overall correctness.

<u>Knowledge Area</u>	<u>Skill Level</u>
1 - 6 (all)	Analyzing
9a - Analytical and Critical Thinking	Applying
9b - Collaboration and Teamwork	Applying
9d - Mathematics and Statistics	Applying
9k - Research and Self-Starter/Learner	Applying

<u>Dispositions</u>			
Meticulous	Self-Directed	Collaborative	Adaptable

2. **(Debugging and Correcting Programs)** In the event that a program contains bugs (syntax or runtime. does not function correctly) be able to present, to a team of programmers, the bugs found and recommend one or more potential fixes.

<u>Knowledge Area</u>	<u>Skill Level</u>
1 - 6 (all)	Creating
8 (a,b,d)	Applying
9a - Analytical and Critical Thinking	Applying
9d - Mathematics and Statistics	Applying
9k - Research and Self-Starter/Learner	Applying
9f - Oral Communication and Presentation	Applying
9g - Problem Solving and Troubleshooting	Evaluating
9i -Quality Assurance / Control	Applying

<u>Dispositions</u>			
Meticulous	Self-Directed	Inventive	Adaptable
Self-directed			

3. **(Presenting Programs)** Be able to present (critiques and justifications of choices) a completed program, its overall design, and its function to a technically minded third party, i.e. other programmers that were not involved in the evaluation of the program.

<u>Knowledge Area</u>	<u>Skill Level</u>
1 - 7 (all)	Evaluating
9a - Analytical and Critical Thinking	Applying
9d - Mathematics and Statistics	Applying
9k - Research and Self-Starter/Learner	Applying
9f - Oral Communication and Presentation	Applying

<u>Dispositions</u>			
Meticulous	Self-Directed	Inventive	Adaptable
Self-directed	Proactive	Purpose-Driven	

4. **(Designing & Writing Programs)** Given a high-level problem statement, work alone or with a team to design and develop a program to address the given problem.

<u>Knowledge Area</u>	<u>Skill Level</u>
1 - 7 (all)	Creating
8 (all)	Applying
9a - Analytical and Critical Thinking	Applying
9c - Ethical and Intercultural Perspectives	Applying
9d - Mathematics and Statistics	Applying
9k - Research and Self-Starter/Learner	Applying
9b - Collaboration and Teamwork	Applying
9d - Multi-Task Prioritization and Management	Applying
9h - Project and Task Organization and Planning	Applying
9i - Quality Assurance / Control	Applying
9l - Time Management	Applying

<u>Dispositions</u>			
Meticulous	Self-Directed	Inventive	Adaptable
Self-directed	Proactive	Purpose-Driven	

5. **(Sharing Programs)** Given a completed program, be able to explain, to a non-technical user, how to use the program to address a given problem.

<u>Knowledge Area</u>	<u>Skill Level</u>
1 - 7 (all)	Understanding
9d - Mathematics and Statistics	Applying
9b - Collaboration and Teamwork	Applying
9f - Oral Communication and Presentation	Applying
9j - Relationship Management	Applying
9m - Written Communication	Applying
9c - Ethical and Intercultural Perspectives	Applying

<u>Dispositions</u>			
Collaborative	Passionate	Responsible	Adaptable
Self-directed	Proactive	Responsive	Professional

## CS1 Knowledge Areas

1. Data Types (Objects, Values, and Operations):
  - a. Integers
  - b. Floating Point Numbers
  - c. Strings/Characters
  - d. Booleans
2. Expressions and Substitution Semantics
  - a. Variable Evaluation - Identifiers
3. Statements and Side-Effect/State-based Semantics (I/O, Mutation, & Control)
  - a. I/O: Read & Write from terminal and files
  - b. Variables (mutation): Declaration, Initialization, Assignment
  - c. Conditionals (control): if, if..else, if..else if... else
  - d. Loops (control): for, while
  - e. Exceptions (control): throw, Try..catch, with
4. Functions/Procedures
  - a. Definition: return value, side-effects
  - b. Application: For value (functional) , for effect (stateful)
5. Data Structures, Classes & Compound Data
  - a. Structs/Named-Tuples
  - b. Tuples
  - c. Dynamic Array-List & Sequences
  - d. Dictionary
6. Design & Problem Solving
  - a. Iteration and Accumulation
  - b. Recursive Functions
  - c. Nested Expressions & Statements
  - d. Functional Composition (helper/auxiliary functions)
  - e. Functional Abstraction
  - f. Data Abstraction and Objects
7. Debugging and Troubleshooting
  - a. Print-statement based debugging
  - b. Debuggers and Steppers
  - c. Unit Tests & Testing Generally
  - d. Syntax, Run-time Errors, and Compiler/Interpreter Messages
8. *Professional Knowledge (Table 4.2 from CC2020 (pg 50))*
  - a. Analytical and Critical Thinking
  - b. Collaboration and Teamwork
  - c. Ethical and Intercultural Perspectives
  - d. Mathematics and Statistics
  - e. Multi-Task Prioritization and Management
  - f. Oral Communication and Presentation
  - g. Problem Solving and Troubleshooting
  - h. Project and Task Organization and Planning
  - i. Quality Assurance / Control

- j. Relationship Management
- k. Research and Self-Starter/Learner
- l. Time Management
- m. Written Communication

## CS1 Skills Hierarchy (Bloom's Taxonomy, CC2020 pg 50)

1. *Remembering* - Recall facts, terms, concepts, answers, etc.
2. *Understanding* - Be able to organize, compare, translate, interpret, and give descriptions of facts and ideas
3. *Applying* - Use knowledge, ideas, facts in different ways to solve problems in new situations.
4. *Analyzing* - Make inferences and find evidence to support solutions
5. *Evaluating* - Make judgements about information, validity of ideas, or quality of material
6. *Creating* - Combine elements of information in a new pattern or propose alternative solutions.

## CS1 Dispositions (From CC2020, pg 51.)

1. Adaptable
2. Collaborative
3. Inventive
4. Meticulous
5. Passionate
6. Proactive
7. Professional
8. Purpose-Driven
9. Responsible
10. Responsive
11. Self-directed

## CS1 Task Environments

1. Read Code (what will this do, expected behavior/outcome unknown): From technical/computational description to high-level purpose.
2. Evaluate Code for Correctness & Simplicity (expected behavior/outcome known). Again, behavior could be given in very technical terms (do this to variable state...) to high-level (find max of ...)
3. Write Code and Modify existing Code
4. Test and Debug Code
5. Explain Code to Technical & Non-Technical Audience
6. Compare and Contrast multiple code solutions to a given problem (correctness and simplicity, not necessarily efficiency)
7. Design a Code-based solution to a real-world problem