**Competency-Based Guide: Computer Science**

1. **Overview**

**How to Use this Guide**

This guide is meant to help you understand the knowledge and competencies typically expected of someone who has a college-level understanding of Computer Science in the areas of Introduction to Programming, Computer Science I, and Computer Science II. This guide will help you to determine which of these topics best describes your learning. This guide is also meant to help you go through the processes of thinking about your learning and competencies in preparation for writing your Prior Learning Request.  Respond to the activity prompts by drawing from your own experience and using the short scenarios provided to discuss how you would approach each situation.

The activity prompts in this guide act as a worksheet to help you identify and describe your learning. The rubric at the end can help you both formulate your responses and assess your learning. Once you have completed all of the competency activities and assessed that you have college-level knowledge in this area, you will compile a more formal iPLA request, incorporating your responses to the competency activities.

**PLA Process**

When you are ready, compile your responses into a learning description to create a more formal iPLA request. Once you complete your learning description, submit it to your mentor for review before you submit your iPLA request to PLA Planner, which initiates the evaluation process.

The college will assign an evaluator to your iPLA request. The evaluator uses two sources to assess your learning: 1) your learning description/iPLA request, which you develop from your answers to this worksheet, with any appropriate supporting evidence, and 2) an interview.

The evaluator asks questions to help clarify and verify your knowledge and to gain a better understanding of the depth and breadth of your learning. The evaluator may ask about possible redundancies between a particular credit request and other studies that are listed on your degree program. The evaluator also may suggest changes in the title of your learning components, the number of credits, the level of learning, or liberal arts and sciences credit designations. Once your interview is complete, the evaluator will write a recommendation which is posted in PLA Planner.

To learn more about the evaluation process, please talk to your mentor and read the college’s iPLA guide at <https://www.esc.edu/degree-planning-academic-review/prior-learning-assessment/individualized-prior-learning-assessment/>

1. **Competency Worksheet for your iPLA Request**

|  |
| --- |
| Competency 1: Programming Concepts |
| **Activity 1: Explain programming concepts (including data structures, recursion, sorting, object-oriented programming, and event-driven programming).**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.  Draft your responses to the following questions.   1. What are data structures? 2. How and why are data structures used? 3. What is recursion? 4. How is recursion used? 5. What is sorting and how is it done? 6. What are the differences among the major programming paradigms? 7. What are the features of object-oriented programming? Please explain them. 8. What are the phases of language translation? 9. What error conditions might be associated with the phases of language translation? 10. Explain the difference between the main representations of memory in the system. 11. What are the subprograms, functions, and methods and why are they valuable in programming? 12. When do you use a variable versus a constant? 13. What is information hiding? |
| As you prepare your request for evaluation, consider your prior knowledge and experience in these areas:   * Reflect on the programming concepts you have used and how you have used each. * What are the differences among the major programming paradigms? What types of problems are best suited to each of these paradigms? * What are the phases of language translation? * What is the difference between the main representations of memory in the system? * What are the subprograms, functions, and methods and why are they valuable in programming? * When do you use a variable versus a constant? * What error conditions might be associated with the phases of language translation? |

|  |
| --- |
| Competency 2: Programming Syntax |
| **Activity 2: Describe programming syntax.**  Please respond to the questions. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore, you may not be able to respond to all questions. That is okay. Answer the best that you can, and you and your mentor will figure out the best title for your learning.   1. What syntax do you use for counting loops versus conditional loops? 2. What are arithmetic and comparison operators, and how are they used? 3. What is the syntax for input and output? 4. What is the syntax for comments (in this language)? 5. What is the assignment operator (in this language)? And how is it different from a comparison operator? |

|  |
| --- |
| Competency 3: Programming Code |

|  |
| --- |
| **Activity 3: Implement programming concepts into code.**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.   1. What are examples in your sample code of the used of variables being defined as primitive data types? What other data types might have been used and what would the implications have been of using these other data types? 2. What are some examples in your sample code of the declaration of variables, assignment of values to the variables, and the use of expressions? Explain an expression and what assignment is made to the variable after evaluation of the expression. 3. Point out a sequence of operations in your sample code and explain why it has to be in that order. 4. Identify a conditional operation, the conditions that might hold going into the operation, and the results based on these various conditions. 5. Identify an example of iteration in your sample code. Indicate the conditions at the start of the loop, how the condition changes, and when it ends. 6. Parameter passing - Identify an example of parameter passing in your sample code. Explain how and why it is checked to be sure it is of the correct type and range and what happens if it is not. 7. Identify the type of file that you submitted and discuss the major file types of at least one programming language. 8. Able to implement significant data structures. 9. Identify the use of recursion in the file that you submitted and explain the condition at the start, how the condition changes, and when it ends. 10. Provide an example of input and output. 11. Provide an example of the use of file I/O. |

|  |
| --- |
| Competency 4: Encourage Creativity, Problem Solving |
| **Activity 4a: Analyze a problem, identify a solution, and implement the solution.**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.   1. Present a problem that requires interpretation and a multi-step solution. 2. Define the scope of the problem you presented. 3. Identify acceptable input and how to handle inappropriate input [including secure coding techniques: criteria for selection of a specific type and use, input data validation] 4. Step-by-step solution: Show how you've broken the problem into the major tasks and subtasks. 5. Sequential: Explain the sequence of the tasks and subtasks. 6. Communicate solution: Provide a structured form of communication for the problem solution such as pseudocode and/or a flowchart |
| **Activity 4b: Planning your work.**  Demonstrate how you go about planning your work. Evidence may include a flowchart or pseudo-code. Demonstrate that you can perform an analysis by explaining how you moved from the problem to the flowchart. Consider the following questions as you put together your response.   * What role do algorithms play in the development of your solution? * Were there any ambiguous points in the problem statement requiring clarification to fully analyze the problem? * How did you implement your flowchart? * What decisions did you make to ensure the appropriate use of secure coding techniques? * Was this a team project? If so, what portion of the flowchart/project were you responsible for creating? |

|  |
| --- |
| Competency 5: Reading Programs, Test, and Debug |
| **Activity 5:** **Read, test, and debug programs.**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.   1. What tools or techniques do you use for tracing execution? What does this tool/technique do that is helpful? 2. Demonstrate the use of execution tracing. 3. What are different types of errors that might occur during programming? What approaches might you use to identify and resolve each type? 4. Provide an example in which you recognized that a program was not doing what it should be doing.  How did you realize there was a problem? 5. Provide an example in which you encountered a syntax error.  Explain how you interpreted the error message, and identified and resolved the problem. 6. Provide an example of how you created a simple test plan that you used to test your code for a variety of input. 7. Provide an example of how you resolved errors identified in testing and your next steps. 8. Describe an IDE (integrated development environment) and explain the features of the code editor and debugger. |

|  |
| --- |
| Competency 6: Comment Code and Document Work |
| **Activity 6: Comment code and document work.**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.   1. What information do you put in a header (and why/what purpose do these elements serve)? 2. Identify the types of comments that should be included in code. 3. What are approaches that help make code readable?  Please discuss things such as variable names and indentation. |

|  |
| --- |
| Competency 7: Professional Behavior |
| **Activity 7: Demonstrate professionalism in response to ethical issues inherent in computing.**  Please provide a response to the questions below. The questions span the topic areas of Introduction to Programming, Computer Science I and Computer Science II. Therefore you may not be able to respond to all questions. That is okay. Answer the best that you can and you and your mentor will figure out the best title for your learning.   1. Identify at least one professional organization relevant to technology that has a code of ethics. 2. What types of guidance to professional codes provide? 3. What specific professional responsibilities are required by such codes? 4. What sorts of societal impacts are of concern to a programmer? How might privacy, security, and reliability be issues in programming? |

|  |
| --- |
| Describe Additional Learning |
| This section is intended to provide you with an opportunity to discuss other learning you may have accomplished that is not already reflected in the Competency prompts.  Please make sure you discuss your learning with your mentor for guidance on how to articulate what you have learned. |

1. **Next Steps**

**Step 1: Compile and Write Your iPLA Request, Based on Responses to Worksheet Activities**

Review and compile all your responses to the activities from the various competency areas into a learning description. Review your learning description with your mentor. From the questions in the guide and the feedback you receive from your mentor, complete your iPLA request.

* Review what you have written and be sure that your learning description conveys your knowledge.
* Re-review the rubric, consider the evaluation criteria, and be sure that your learning description meets or exceeds expectations.
* Consider the feedback you have received from your mentor.
* The final iPLA request should reflect college writing; be sure to proofread what you write before submitting it to PLA Planner.

**Step 2: Prepare for your interview**

After the evaluator has reviewed your request, he or she will conduct an interview with you to gain a more in-depth understanding of your knowledge. You are expected to participate actively in the evaluation process. The evaluator may ask you to demonstrate or elaborate further on your learning. The evaluator needs to be confident that your learning is college level and not redundant with other learning in your degree plan.

Think about questions that the evaluator may want to ask you and think about possible responses that you can provide. The interview process can be interesting and informative. The interview is an opportunity for you to discuss ways in which you have used your knowledge and applied that knowledge in different situations. It can result in an increased understanding of your learning and provide you with new or differing perspectives.

**Rubric for Computer Science**

|  |  |  |
| --- | --- | --- |
| **Competency 1: Explain programming concepts (including data structures, recursion, sorting, object-oriented programming, and event-driven programming).** | | |
| **What are data structures?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies that data structures organize data. Provides examples of data structures beyond primitive data types. (Does not need to be able to demonstrate use)  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **How and why are data structures used?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Describe common applications for some of the following data structures: stack, queue, priority queue, set, and map.  ***Applicable to:*** Computer Science II | N/A |
| **What is recursion?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies that a recursive function calls itself; contrasts recursion with iteration.  ***Applicable to:***   * Intro to Programming * Computer Science II (prerequisite) | N/A |
| **How is recursion used?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides an example of the use of recursion.  ***Applicable to:*** Computer Science II | N/A |
| **What is sorting and how is it done?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Recognizes sorting and identifies at least basic sorting approaches (bubble, insertion, selection).  Describes at least one sorting technique.  ***Applicable to:*** Computer Science II | N/A |
| **What are the differences among the major programming paradigms?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Identifies languages in each paradigm, but does not discuss specifics (or does not generalize to how the paradigms differ) | Explain some of the key features of the major paradigms or contrast their differences. Identifies sample languages in each.  ***Applicable to:***   * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What are the features of object-oriented programming? Please explain them** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Lists major features but does not explain how they work OR explains some of the major features, but does not explain all of the three major features (encapsulation, inheritance, and polymorphism). | Explains the major features including object, class, inheritance, polymorphism, encapsulation, and message events.  At a minimum, addresses encapsulation, inheritance, and polymorphism.  ***Applicable to:***   * Intro to Programming (for credit in object-oriented language) * Computer Science I (meet minimum requirements) * Computer Science II (must address all features) | N/A |
| **What are the phases of language translation?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Lists the phases, but does not explain what happens in them | Explains the phases of language translation (must include compiling, interpreting, linking, and executing) making clear the differences among the phases.  ***Applicable to:***   * Intro to Programming * Computer Science I | N/A |
| **What error conditions might be associated with the phases of language translation?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies error conditions and explains what triggers them.  ***Applicable to:***   * Intro to Programming * Computer Science I | N/A |
| **Explain the difference between the main representation of memory in the system** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | How is data stored in computers? Specifically addresses different types of data structures and how we retrieve values from memory.  Addresses floating point and integers and pointers.  ***Applicable to:***Computer Science I | N/A |
| **What are subprograms, functions, and methods and why are they valuable in programming?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Explains the benefits of a modular approach. Explains the appropriate handling of input and output.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **When do you use a variable versus use a constant?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Contrasts constant and variables and explains why a constant is appropriate at times.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What is information hiding?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Explains information hiding including the mechanism for it and reason for it.  ***Applicable to:*** Computer Science II | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 2: Describe programming syntax.** | | |
| **What syntax do you use for counting loops versus conditional loops?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Writes a simple counting loop and a simple conditional loop.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What are arithmetic and comparison operators and how are they used?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Lists comparison operators and lists arithmetic operators. Describes how they are used in an expression and provides an example.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What is the syntax for input and output?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides the syntax for input and for output (in the language they are demonstrating).  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What is the syntax for comments (in this language)?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides the syntax for comments (in the language they are demonstrating).  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **What are the primitive data types (in this language)?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides a list of the primitive data types (in the language they are demonstrating).  At a minimum, identifies at least one character type, at least two numeric types, and boolean.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What is the assignment operator (in this language)?  (And how is it different from a comparison operator)** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies an assignment operator (and recognizes the difference between that and a comparison operator).  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 3: Implement programming concepts into code.** | | |
| **What are examples in your sample code of the use of variables being defined as primitive data types? What other data types might have been used and what would the implications have been of using these other data types?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should define variables as primitive data types.  Identifies this section and explains why these are the appropriate data types.   Explains whether there is another data type that might have been used and the implications for using this other data type.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What are examples in your sample code of the declaration of variables, assignment of values to the variables, and the use of expressions? Explain an expression and what assignment is made to the variable after evaluation of the expression** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should declare variables and assign values to those variables. At least one operation should include an expression.  Identifies this section and explains how the expression works and what assignment is made to the variable.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Point out a sequence of operations in your sample code and explain why it has to be in that order** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should include a sequence of operations.   Identifies the sequence and explains the order in which the operations are done.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Identify a conditional operation, the conditions that might hold going into the operation, and the results based on these various conditions** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should include at least one conditional operation.  Identifies the conditions that might come into this operation and explains what happens based on the various possible conditions.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Identify an example of iteration in your sample code. Indicate the conditions at the start of the loop, how the condition changes, and when it ends** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should include some form of iteration (loop).  Explains the condition at the start of the loop, how the condition changes, and when the loop ends.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Identify an example of parameter passing in your sample code. Explain how and why this is done including what is passed in and what is returned** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should include some parameter passing.  Explains how and why this is done including what is passed in and what is returned.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Identify an example of secure coding techniques in your sample code. Explain what input is being requested and how and why it is checked to be sure it is of the correct type and range and what happens if it is not** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should accept and validate input and gracefully handle inappropriate input.  Explains what input is being requested and how and why it is checked to be sure it is of the correct type and range and what happens if it is not.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Identify the type of file that you submitted and discuss the major file types of at least one programming language** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies the major file types used by at least one programming language.  ***Applicable to:***   * Intro to Programming * Computer Science II | N/A |
| **Implement significant data structures** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Designs and implements commonly used data structures including a class, array.  ***Applicable to:***   * Intro to Programming * Computer Science II | N/A |
| **Identify the use of recursion in the file that you submitted and explain the condition at the start, how the condition changes, and when it ends** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | The code should include some form of recursion.  Explains the condition at the start, how the condition changes, and when it ends (breaks out).  ***Applicable to:***Computer Science II | N/A |
| **Provide an example of input and output** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides sample code where input is accepted and output is generated.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Provide an example of the use of file I/O** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provide sample code where file I/O is done.  ***Applicable to:*** Computer Science II | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 4: Analyze a problem, identify a solution, and implement the solution** | | |
| **Present a problem that requires [interpretation and] a multi-step solution** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | An appropriate problem is not already in pseudocode, requires interpretation, and involves a multi-step solution.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Define the scope of the problem you presented** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Delineate the problem.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Identify acceptable input and how to handle inappropriate input  [including secure coding techniques: criteria for selection of a specific type and use, input data validation]** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies acceptable input, identifies appropriate type for that input, validates the input, and addresses how to handle inappropriate input.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Show how you’ve broken the problem into the major tasks and subtasks** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| The problem is broken into steps but some steps are obviously missing and/or some steps are too large to be clear | Breaks the problem into the major tasks and sub-tasks.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Explain the sequence of the tasks and sub-tasks** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Sequences the tasks and sub-tasks.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Provide a structured form of communication for the problem solution such as pseudocode and/or a flowchart** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Problem solution is translated into a structured communication such as pseudocode and/or flowchart.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 5: Read, test, and debug programs** | | |
| **What tools or techniques do you use for tracing execution? What does this tool/technique do that is helpful?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies a tool or technique for tracing execution.   Explains what this tool/technique does that is helpful, including how looking at the states of all of the variables at each point is helpful.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **Demonstrate the use of execution tracing** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Traces the execution of code segments and explains what is going on.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II (prerequisite) | N/A |
| **What are different types of errors that might occur during programming?   What approaches might you use to identify and resolve each type?**  **Alternate prompt: What is the difference between logic errors and syntax errors? What approaches might you use to identifying and resolving each type?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Distinguishes between logic errors and syntax errors and discusses the different approaches to identifying and resolving each.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Give an example in which you recognized that a program wasn't doing what it should be doing. How did you realize there was a problem?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Reads the task description and recognizes when the program is not doing what is expected. The intent is to get at logic errors.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Give an example in which you encountered a syntax error, explain how you interpreted the error message, and how you identified and resolved the problem** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Read error messages and locate and correct the problem being identified.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Provide an example of how you created a simple test plan that you used to test your code for a variety of input** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Creates a simple test plan consisting of a variety of input. Reviewing the output determines whether the program is functioning as expected. The input must include data out of the expected range and/or type of input, and data that test all paths within the program.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Provide an example of how you resolved errors identified in testing and your next steps** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Pinpoints errors identified in testing and steps for resolving/re-testing.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **Describe an IDE (integrated development environment) and explain the features of the code editor and debugger** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Discusses the features of integrated development environments (IDE) including the code editor and debugger.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 6: Comment code and document work** | | |
| **What information do you put in a header (and why/what purpose do these elements serve)?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies the appropriate information for a header.  ***Applicable to:***Computer Science II | N/A |
| **Identify the types of comments that should be included in code** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies key areas that need to be addressed with comments including changes to vanilla, section headers, and algorithm explanations.  Recognizes that not every line should be commented.  ***Applicable to:***   * Intro to Programming * Computer Science II | N/A |
| **Readable code: What are approaches that help make code readable? Please discuss things such as variable names and indentation.** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | * Useful variable names * Standard indentation * Appropriately commented (per above)   ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |

|  |  |  |
| --- | --- | --- |
| **Competency 7: Demonstrate professionalism in response to ethical issues inherent in computing** | | |
| **Identify at least one professional organization relevant to technology that has a code of ethics** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies at least one professional organization relevant to technology and/or their professional field that has a code of ethics.  ***Applicable to:***   * Intro to Programming * Computer Science I * Computer Science II | N/A |
| **What types of guidance do professional codes provide** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Identifies the types of guidelines within at least one professional code.  ***Applicable to:***Computer Science I | N/A |
| **What specific professional responsibilities are required by such codes?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides examples of some specific professional responsibilities in such codes.  ***Applicable to:***Computer Science I | N/A |
| **What sorts of societal impacts are of concern to a programmer? How might privacy, security, and reliability be issues in programming?** | | |
| **Approaches Expectations** | **Meets Expectations** | **Exceeds Expectations** |
| Does not meet expectations as outlined | Provides examples of privacy, security, and reliability, and discusses them in terms of societal impacts.  ***Applicable to:***Computer Science II | N/A |