Lesson Plan

# Lesson 03: Conditional Statements

|  |  |  |  |
| --- | --- | --- | --- |
| **Lesson Overview** | | | |
| **Lesson Length** | 3 hours (180 minutes) | | |
| **Overview** | This session will familiarize students with data structures used within the Python scripting language. | | |
| **Objectives**   * **Domain:** Cognitive * **Level:** Applying | Using conditionals, loops, Python dictionaries, local data, the CSV Library, and the Glob Library, students will be able to:   * TLO 1: Examine the implications of using computation to solve a problem   + ELO 1.1: Discuss best practices for using computation to solve a problem   + ELO 1.2: Suggest types of problems that can be solved through computation   + ELO 1.3: Show how computation can solve a problem * TLO 2: Recognize key computer science concepts   + ELO 2.2: Identify data structures used in Python scripting | | |
| **Instructional Methods** | Informal lecture, demonstration, guided discussion, practical exercises | | |
| **Assessment Strategies** | Informal: Guided and practical exercises  Formal: N/A | | |
| **Materials and Equipment Needed** | Required:   * SBU * Jupyter Notebook * Python   Optional:   * N/A | | |
| **Background Resources** | Resource:   * NGA SME * Technical facilitators   Subject matter/content questions may be referred to:   * Jeremy DeBrow, Course Manager   [Jeremy.R.Debrow@nga.mil](mailto:Jeremy.R.Debrow@nga.mil)  [Jeremy.R.Debrow@coe.ic.gov](mailto:Jeremy.R.Debrow@coe.ic.gov)  National Geospatial-Intelligence College (NGC) HDNPE Branch  Unclassified: 571-557-7583 | | |
| **Comments** | ELOs 1.1, 1.2, and 1.3 are meant to be covered throughout the entire lesson (informal lecture and assessment). Instructors will be expected to facilitate classroom discussion that identifies problems best suited to be solved computationally, best practices for solving those problems, and potential solutions. | | |
| **Lesson Sequence** | | | |
| **Lesson Topic** | **Instructional Method** | | **Time**  **(mins)** |
| Introduction | Informal lecture, guided discussion | | 05 |
| Review | Informal lecture, guided discussion, demonstration | | 10 |
| Lesson: Data Structures | Informal lecture, guided discussion, demonstration | | 60 |
| Assessment | Guided exercise, practical exercises | | 100 |
| Conclusion | Informal lecture | | 05 |
| **Lesson Outline** | | | |
| **Introduction** | | | |
| Introduction | * **Attention** (to be personalized by instructor) * **Motivation** (to be personalized by instructor) * **Overview** (to be personalized by instructor)   + Learning objectives   + Lesson topics/main points * **Rules of Engagement** (to be personalized by instructor) | | |
| **Body** | | | |
| ***Lesson Topic*** | ***Main Points/Notes*** | ***Personalization*** | |
| Review | **Informal Lecture/Guided Discussion/Demonstration (10 minutes)**  **Lesson 3: Conditional Statements**  **3.1. Objectives**  **3.2. Overview**  *(Facilitator Notes:*   * *Have students load the U\_CSCI2011\_L03\_Conditional\_Statements\_SG\_V3.0.ipynb file to begin the lesson.)* * *See instructor’s notebook for instructor guidance for discussion points and guided exercises.* * *Refer back to Lesson 1 and relate the four steps of problem-solving using Computational Thinking (Decomposition, Pattern Recognition, Abstraction, & Algorithm Design) to lessons, exercises, examples, student questions/comments, etc., as appropriate throughout this lesson.)*   **3.3. Review**   * 3.3.1. String * 3.3.2. Numbers * 3.3.3. Casting |  | |
| Lesson: Data Structures | **Informal Lecture/Guided Discussion/Demonstration (60 minutes)**  **3.4 Lesson: Conditional Statements**   * 3.4.1. Boolean Values and Expressions   *(Facilitator Note: Discuss NameError associated with type(true) and reiterate creation of variables, proper variable naming convention, and what happens when the kernel restarts.)*   * + 3.4.1.1. Comparison Operators   + 3.4.1.2. Membership Operators   + 3.4.1.3. Logical Operators * 3.4.2. If Statements |  | |
| Assessments | **Guided Exercise/Practical Exercise (100 minutes)**  *(Facilitator Note: All assessments should incorporate a facilitator directed discussion on computational thinking techniques as they relate to the assigned problem. Utilize student handouts, performance support tools, or projected code cells to capture facilitator/student discussion.)*  **3.5. Guided Exercise: Code a Basic Calculator**  *(Facilitator Notes: Refer back to Lesson 1 and relate the four steps of problem-solving using Computational Thinking (Decomposition, Pattern Recognition, Abstraction, & Algorithm Design) as appropriate throughout these exercises.)*   * Our task is to write a script that will perform basic subtraction and addition. The inputs we get are strings of numbers and operator characters. Assume that all inputs will include two positive numbers (ints or floats) separated by either a plus (+) or minus (-) sign. Our script should output the answer to any valid input string, and the data type of the final output should be a float.   **3.6. Practical Exercises**  *(Facilitator Note:*   * *Refer back to Lesson 1 and relate the four steps of problem-solving using Computational Thinking (Decomposition, Pattern Recognition, Abstraction, & Algorithm Design) as appropriate throughout these exercises.* * *The practical exercises deemed most important due to content and/or a cumulative result, which should be completed first in the interest of maximum training value in relation to time are Practical Exercises 1, 3, and 5. Ensure you go over the exercise solutions and (as necessary) the processes to arrive at the solutions with the students.* * *Follow-up questions are designed to be asked by the facilitators individually as each student completes the task and has it looked at by a facilitator.*   **3.6.1. Practical Exercise 1: Practice Boolean Expressions**   * Problem 1: Type out what each boolean operator does. * Problem 2: Create two different boolean expressions that evaluate to True using different comparison operators. * Problem 3: Create a different boolean expression that evaluates to True using membership operator. * Problem 4: Create a different boolean expression that evaluates to False using membership operator. * Problem 5: Create two unique boolean expressions using logical operators and any other operators.   **3.6.2. Practical Exercise 2: Practice If Statements**   * Problem 1:   + When do you use if statements?   + What does the 'elif' keyword allow you to do and how many times can you use it? * Problem 2: Create a variable that hold's a person's age. Use that variable to create an if statement printing whether that person can vote!   **3.6.3. Practical Exercise 3: Guess the Output**   * Without running the code below, what is the final value of number?   **3.6.4. Practical Exercise 4: Odd or Even?**   * Write a Python script that will take an integer and print whether it is even or odd.   **3.6.5. Practical Exercise 5: Valid Question?**   * Write a Python script that will take a string and check if it is a valid question.   **3.6.6. Practical Exercise 6: Digit in Number**   * Write a script that, given a number (int or float) and a digit (int), will output True if the number contains the digit and False otherwise.   **3.6.7. Practical Exercise 7: Century From Year**   * Write a script that, given a year (int) outputs the century as defined in the table below.   **3.6.8. Practical Exercise 8: Date Formatting**   * Problem 1: Write a script that, given a month (str) and a date (int), will output a string that combines both date and month like so: the <date> of <month>. * Problem 2: [Challenge] Extend your script so that the output string contains the appropriate suffix for the date ('st', 'nd', 'rd', 'th'). * Problem 3: [Challenge] Extend your script to output a helpful error message if the date is not between 1 and 31 (inclusive). * Bonus: Check for valid date values based on the month (e.g. February dates can only be between 1 and 29).   **3.6.9. Practical Exercise 9: Valid Emails?**   * Given an email address, check that it is a valid email.   **3.7. Appendix** |  | |
| Administrative Notes | N/A |  | |
| **Assessment** | | | |
| ***Assessment Type*** | ***Instructions/Prompts/Notes*** | | |
| Guided Exercise | See the facilitator notes located above for additional guidance. All exercises will be conducted inside the Jupyter Notebook lesson file. | | |
| Practical Exercise | See the facilitator notes located above for additional guidance. All exercises will be conducted inside the Jupyter Notebook lesson file. | | |
| **Conclusion** | | | |
| Conclusion | * **Final Summary** (to be personalized by instructor)   + Review learning objectives   + Review lesson topics/main points * **Remotivation** (to be personalized by instructor) * **Closure** (to be personalized by instructor) * **Next Lesson Introduction** (to be personalized by instructor) | | |