Lesson Plan

# Lesson 05: Unordered Data Structures

|  |  |  |  |
| --- | --- | --- | --- |
| **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: Unordered Data Structures**  **3.1. Objectives**  **3.2. Overview**  *(Facilitator Notes:*   * *Have students load the U\_CSCI2011\_L05\_Data\_*   *Structures\_2\_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**   * 5.3.1. Lists * 5.3.2. Tuples * 5.3.3. Boolean Expressions * 5.3.4. If Statements |  | |
| Lesson: Data Structures | **Informal Lecture/Guided Discussion/Demonstration (60 minutes)**  **5.4 Lesson: Data Structures**   * 5.4.1. Unordered Collections * 5.4.2. Sets   + 5.4.2.1. Creating Sets   + 5.4.2.2. Set Methods and Operations * 5.4.3. Dictionaries   + 5.4.3.1. Creating Dictionaries   + 5.4.3.2. Dict Methods and Operations * 5.4.4. Common Operations Across Data Structures |  | |
| 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.)*  **5.5. Guided Exercise: Analyzing Crime Data**  *(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.)*   * We've received some data about criminal activity, and have been tasked with doing some analysis of the data. The dataset is rather small, but we expect to be receiving similar reports on a recurring basis and will always want to perform the same analysis. This is a perfect case to write some Python code to do our analysis for us. The data is in the code cell below, and we want to answer the following questions:   + What dates do we have data for?   + How many crimes were reported on the first and last dates in the data set?   + How many unique coordinates are there? Are there coordinates that appear more than once?   + Did 10-11-2016 have more or less crimes than average for that day on record? * Follow-up Questions   + 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.   **5.6. 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.*   **5.6.1. Practical Exercise 1: Knowledge Check: Sets**   * What are the differences between sets and lists? When would you use a set rather than a list? * Jot down all the actions that you can do using sets.   **5.6.2. Practical Exercise 2: Practice Sets**   * Create an empty set and save it to a variable called fruits. * Add three different fruits to the fruits set using .add(). * Create a set with everyday groceries and save it to a variable called groceries. * Find the length of the grocery list. * Find all the items in your fruits set that are in the groceries set. That is, find the intersection of the two sets. * Create the set of all the groceries that aren't in the fruits set. That is, find the difference between the groceries and the fruits.   **5.6.3. Practical Exercise 3: Knowledge Check: Dictionaries**   * In the broadest terms possible, what are dictionaries made up of? * In one dictionary, can we have multiple different keys with the same values? Why or why not? * In one dictionary, can there be duplicate keys linked to different values? Why or why not? * What kind of data types and data structures can be used as values in a key-value pair? * When applying common functions on a dictionary (such as the sum() function), is the function applied to the dictionary's keys or values?   **5.6.4. Practical Exercise 4: Practice Dictionaries**   * Create a dictionary with 6 people and their ages, where the age is the key and the name is the value. Account for a few people having the same age. * Add the following key-value pair to your dictionary: {43: 'Laura'} * Print the name of the youngest person (or people) in the dictionary.   HINT: Try using the min() function on the dictionary and see what that gives you.   * Print the name of the oldest person (or people) in the dictionary.   **5.6.5. Practical Exercise 5: Deduplicated List**   * Create a list of data, and be sure to include duplicates. Store your list in a variable. Then, write a script to create a list with all and only the unique values from your list.   **5.6.6. Practical Exercise 6: Employee Availability**   * Your employees have provided you a list of days that they are available to work, you've complied this data into the dictionary seen in the code block below. * Problem 1: Write a script that takes a name and a day as inputs, and outputs True if that employee can work on that day and False otherwise. * Problem 2: One of your employees calls in sick on Wednesday and you need to find an employee that is available to come in as soon as possible. Your order of preference is Tara, Lucy, Tom, Alex, Rachel, Sara, Zach, then after that no preference. Who is the most preferred employee available?   **5.6.7. Practical Exercise 7: Symmetric Difference**   * Write a script that does the same thing as .symmetric\_difference() using only the other set methods from this lesson: .union(), .intersection(), and .difference().   **5.7. 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) | | |