ECT Lesson Plan: Sorting Data

Lesson plan at a glance...

Core subject(s)	Mathematics
Subject area(s)	Statistics & Probability
Suggested age	14 to 18 years old
Prerequisites	Familiarity with central tendency
Time	**Preparation:** 10 to 22 minutes **Instruction:** 40 minutes
Standards	**Core Subject:** CCSS Math **CS:** Australia, CSTA, UK

In this lesson plan…

Lesson Overview

Students gather data from classmates and **analyze** categorical and quantitative **data** using spreadsheet functions and/or Python**. Decomposition** is used to break the sorting process into individual steps and students learn how to **design algorithms.** Upon completion of this lesson, students will be able to write algorithms to calculate the mean, median, mode, and range of data**.** Additionally, students will be able to use spreadsheet functions or programming instructions to sort data based on different criteria, leading to the identification of patterns and trends**.**

For more advanced activities on sorting data, view the ECT Lesson Plan "Sorting the World's Cities**.**"

Materials and Equipment

- ☐ For the teacher:
  - *Required:* Presentation set-up
    - Internet-connected computer
      - Chrome browser (https://www.google.com/chrome/browser/desktop) recommended
    - Projector and projection screen or other flat projection surface
  - *Required:* Access to YouTube (http://www.youtube.com) for optional extension activity
  - *Recommended:* Whiteboard and markers or equivalent
- ☐ For the student:
  - *Required:* Software Development Environment
    - Internet-connected computers (one (1) computer per student recommended)
    - Python 2.x (https://www.python.org/) OR a web-based Integrated Development Environment (IDE) such as Trinket (https://trinket.io/)
  - *Required:* Google Sheets (http://docs.google.com/spreadsheets) or other spreadsheet app
  - *Recommended:* Paper and pen/pencil

Preparation Tasks

	Confirm that your computer is on and logged-in	1 to 3 minutes
	Confirm that your projector is turned on and is projecting properly	1 to 4 minutes
	Confirm that all students' computers are turned on and logged-in	3 to 5 minutes
	Download and install Python 2.x (https://www.python.org/downloads) or navigate to Trinket (https://trinket.io/)	5 to 10 minutes

The Lesson

Warm-up Activity: Collecting data	10 minutes
Activity 1: Sorting data	10 minutes
Activity 2: Exploring algorithms	15 minutes
Wrap-up Activity: Understanding key concepts	5 minutes

Warm-up Activity: Collecting data (10 minutes)

Activity Overview: In this activity, students will gather meaningful and relevant content from each other using data collection. They will use the collected data in the following activities.

Activity:
Collect class data, such as shoe size vs. height in inches, and enter it into a spreadsheet.
Sample Class Data:

	A	B	C
1	**Name**	**Height**	**Shoe Size**
2	Sasha	63	6
3	Hector	68	9
4	Kayla	67	8
5	Adriana	65	7
6	Sean	64	7
7	Kay	61	3
8	Hillary	69	9
9	Thomas	68	9
10	Luis	64	8
11	Mayra	66	6
12	Michael	68	10
13	Tonya	64	8
14	Philip	65	8
15	Raymond	66	9
16	Jalen	70	10
17	Samantha	62	4

Activity 1: Sorting data (10 minutes)

Activity Overview: In this activity, students will recognize patterns within a data set through the process of sorting the data based on various criteria (height and shoe size). They will use a spreadsheet or Python to organize a set of data. Students recognize that by sorting data, we can more easily generalize the pattern between height and shoe size.

Notes to the Teacher:
Instructions for how to organize a set of data using Google Sheets are:
Sort by shoe size:

1. Highlight all of the data
2. Click on the **Data** menu
3. Click on **Sort**
4. Sort by **Shoe Size**

Sort by height:

1. Highlight all of the data
2. Click on the **Data** menu
3. Click on **Sort**
4. Sort by **Height**

Sort by name:

1. Click on the **Name** column
2. Click **Sort A-Z** at the top of the column

Instructions for how to use Python to organize a set of data are:

1. Open Python and type **Ctrl-N** (PC) or **Command-N** (Mac) to create a blank page.
2. Save the data set above as `StudentData.txt` in the same directory as where the code below is saved. After entering the code, press **F5** or from the menu **Run** → **Run Module** to run the code.

```
import re

#extracts the first column from the data
def GetName(list):
    return list[0]

#extracts the second column from the data
def GetHeight(list):
    return float(list[1])

#extracts the third column from the data
def GetShoe(list):
    return float(list[2])

#reads in the data from StudentData.txt
data = open('StudentData.txt','r')
data_search = re.findall('(\w+)\s(\d+)\s(\w+)', data.read())

#sorts the data by name
ordered_names = sorted(data_search, key=GetName)

#sorts the data by height
ordered_heights = sorted(data_search, key=GetHeight)

#sorts the data by shoe size
ordered_shoes = sorted(data_search, key=GetShoe)

#print out the results
print "Alphabetical Order\n"
for column in ordered_names:
    print column + "\n"

print "Sorted by height\n"
for column in ordered_heights:
    print column + "\n"

print "Sorted by shoe size\n"
for column in ordered_shoes:
    print column
```

Activity:
Have students sort the data by height and list the the five tallest students and the five shortest students. They can do this either by hand, electronic spreadsheet, or Python. Ask them the following questions:

Q1: Can you identify the tallest and shortest students faster before or after sorting the data? Repeat the process for the data on shoe size.

Q2: Name the people who are on both "top five" lists. Name any people on the list of the five tallest, but not on the list of students with the largest shoe size.

Assessment:
A1: It is faster to identify the information that we wish to find after the data has been organized. By sorting data, we can more easily identify a pattern between height and shoe size. We can generalize the pattern if we notice that as height increases, shoe size also increases.

A2: Answers will vary. Students see how different **patterns** overlap, as well as the limits of some generalizations.

Activity 2: Exploring algorithms (15 minutes)

Activity Overview: In this activity, students **design algorithms** by **decomposing** a process into individual steps.

Activity:
Have students write an algorithm to calculate each of the following:

1. the average height of your class
2. the median height of your class
3. the mode of the shoe sizes in your class
4. the range of shoe sizes in your class

Teaching Tips:

- Divide students into small groups of 4-5 and have each group write a different algorithm. Have them test their algorithm on another group.

Assessment:
Writing an algorithm helps students to decompose a process into individual steps.

1. average: add up the values and divide the sum by the number of values that you added.
2. median: sort the data from least to greatest, fold your data set in half and circle the one or two pieces of data that lie on the fold.
3. mode: tally up the number of times each data entry occurs to identify the piece of data that appears most frequently.
4. range: sort the data from least to greatest and circle the two pieces of data at either end of the data set.

Activity:
Ask your students the following questions:

Q1: What step of the spreadsheet algorithm corresponds to the function "`sorted( )`" in Python?

Q2: In the line of code "`orderednames = sorted(datasearch, key=GetName)`", which argument tells Python the name of the data that we want to analyze?

Q3: What part of the program instructs Python to sort by Height?

Q4: What part of the Python program most closely corresponds to hitting the [Return] or [Enter] key and seeing the data sorted in the spreadsheet?

Assessment:
A1: Clicking on **Sort A-Z** from the data menu corresponds to the `sorted` function in Python. Students see that there is some overlap in the **algorithm** used to sort data in Excel and the built-in functions used in Python.

A2: The data from our spreadsheet is stored in a variable called `datasearch`, so when we type `datasearch` into the sorted function we are telling Python what data to sort. By focusing on a single line of code within a relatively lengthy program, students begin to understand the individual parts of a multi-step process **(decomposition).**

A3: The line `orderedheights = sorted(datasearch, key=GetHeight)` gets the data from `datasearch`, and runs it through the function `GetHeight`. Identifying the part of an **algorithm** that performs a specific function encourages students to focus on decomposing a multi-step process.

A4: The `print` statements instruct Python to print the answer to the screen similarly to how hitting [Return] or [Enter] instructs the spreadsheet to display the result. Students see how an **algorithm** in Python corresponds to an action in a spreadsheet.

Wrap-up Activity: Understanding key concepts (5 minutes)

Activity Highlights: In this activity, students will finish the lesson by demonstrating their understanding of the key concepts covered during this lesson.

Student Activity:
Journaling: Students respond to the following prompt in their journal or word processor:
In what other situations might you need to sort data and how would it help with data analysis?

Learning Objectives and Standards

Learning Objectives	Standards
LO1: Students will be able to collect data from classmates and enter it into a spreadsheet.	*Core Subject* CCSS.MATH.CONTENT.HSS.ID: Interpreting Categorical and Quantitative Data. *Computer Science* CSTA L2.CT.14: Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions.
LO2: Students will be able to sort data, using a spreadsheet or Python.	*Computer Science* CSTA L3A.CT.4: Compare techniques for analyzing massive data collections.

	UK 4.2: Develop and apply their analytic, problem-solving, design, and computational thinking skills.
LO3: Students will be able to recognize and generalize patterns in sorted data.	*Computer Science* AUSTRALIA 10.4 (Collecting, managing and analyzing data): Analyse and visualise data to create information and address complex problems; and model processes, entities and their relationships using structured data. CSTA L3B.CT.9: Analyze data and identify patterns through modeling and simulation.
LO4: Students will be able to write algorithms to calculate mean, median, mode, and range of a set of data	*Computer Science* CSTA L2.CT.1: Use the basic steps in algorithmic problem-solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing and evaluation). UK 3.2: Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem.

Additional Information and Resources

Lesson Vocabulary

Term	Definition	For Additional Information
Algorithm	A series of instructions that can be repeated over and over with the same result for a given input (e.g. recipe, computer software, sheet of music notes).	http://en.wikipedia.org/wiki/Algorithm
Pattern	A discernible regularity in the world or in a man-made design (i.e. data)	http://en.wikipedia.org/wiki/Pattern
Sorting	Any process of arranging items according to a certain sequence or in different sets, by either ordering or categorizing	https://en.wikipedia.org/wiki/Sorting

Computational Thinking Concepts

Concept	Definition
Algorithm Design	Creating an ordered series of instructions for solving similar problems
Data Analysis	Making sense of data by finding patterns or developing insights
Data Collection	Gathering information
Decomposition	Breaking down data, processes or problems into smaller, manageable parts
Pattern Generalization	Creating models of observed patterns to test predicted outcomes
Pattern Recognition	Observing patterns and regularity in data

Extension Activities for Student Enrichment

- Have students watch this animated video (http://www.youtube.com/watch?v=vxENKIcs2Tw) explaining two types of sort - bubble sort and quicksort - and comparing their performance. Discuss with the students.

Administrative Details

Contact info	For more info about Exploring Computational Thinking (ECT), visit the ECT website (g.co/exploringCT)
Credits	Developed by the Exploring Computational Thinking team at Google and reviewed by K-12 educators from around the world.
Last updated on	06/30/2015
Copyright info	