## MOTIVATION

In the last unit, we saw that it's a lot easier and faster to search for an object if the list is sorted. But how do we make a sorted list from an unsorted list? Is it worth the effort?

## OBJECTIVES

Students will be able to:

* + describe the selection, bubble, insertion, and merge sorting algorithms
	+ demonstrate how these four algorithms work by sorting weights or cards
	+ decide when an unsorted list should be sorted, then searched, or when it should just be searched

## RESOURCES

Lesson Plan ([Source](http://www.muddx.com/c4x/HMC/MyCS/asset/Sorting_Algorithms_Lesson_Plan.docx) / [PDF](http://www.muddx.com/c4x/HMC/MyCS/asset/Sorting_Algorithms_Lesson_Plan.pdf))

Food and Color Cards ([Source](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pptx) / [PDF](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pdf))

These are the same cards you used for the search activity. If you still have them, you don't need to print them out again.

Sorting Cards Activity ([Source](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pptx) / [PDF](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pdf))

# Sorting Algorithms

## OBJECTIVES

Students will learn about the selection, bubble, insertion, and merge sorting algorithms and practice sorting objects with each of these algorithms. Through these activities, students will realize that sorting requires a lot of computation.

## SAMPLE AGENDA

* Motivate the need for sorting algorithms
* Selection Sort
* Bubble Sort
* Insertion Sort
* Merge Sort

## MOTIVATION: WHY DO WE NEED SORTING ALGORITHMS?

Lists of information are often more useful if they are sorted in some order: alphabetical, by price, by relevance, or by quantity. It's much easier to find the object we want if all the objects are sorted.

But how do we sort a list? It's something we don't usually think about, because people are very good at quickly sorting small lists. Put the following numbers in order: 97, 25, 13, 42, 81, and 7. You were probably able to sort that so quickly that it would be difficult to break the sorting process down into precise steps.

Sometimes the objects we want to sort are data in a computer. Instead of 6 numbers, there might be a hundred, or ten thousand, or a billion numbers. How do we tell the computer to sort this data?

## ACTIVITY PREPARATION: RANKING CARDS

Take out your color and food cards from the search activity; you'll need them again. Rank either the color cards or the food cards in order of preference, and write your rankings on the Sorting Cards handout. Don't let your partner see your rankings! 

 

**How did you come up with a ranking for your cards?**

When you ranked your cards, you *sorted* them in order of preference. What were the steps you took to sort your cards? Would these steps work for any kind of sorting problem?

## INTRODUCING SORTING ALGORITHMS

There are **lots** of different ways to sort a collection of objects! Computer scientists make sorting algorithms, which are just step-by-step processes for putting objects in order. Today we'll learn about three of them: selection sort, bubble sort, and insertion sort. These algorithms can be used to sort any unsorted list!

These sorting algorithms work slightly differently than human sorting does. The computer can only "look at" or "compare" two objects at a time, and it can't "see" the whole list. It can only find the "best" or "worst" thing if it compares all of them. Let's start with selection sort:

## WHAT IS SELECTION SORT?

The first sorting algorithm we'll look at is selection sort. We start with an unsorted list in white. The computer goes through and compares the numbers in the list, two at a time.

It starts with the first two numbers. The smallest number is in red, and the number it's being compared to is in blue. When the computer finds a new smallest number, that number becomes red. This algorithm is called "selection sort" because it keeps selecting the smallest number in the unsorted list.

When the computer has finished going through the list, the smallest number it found is in red, and it places that number at the beginning of the list and bumps everything else down. The smallest number is highlighted in yellow to indicate that it is forming the beginning of a new sorted list. The computer doesn't move things around in the sorted list, it just keeps adding numbers to the end of it, since it finds the next smallest number each time.

The computer repeats the process of finding the smallest number in the white unsorted list and putting it in the next spot in the sorted list until all of the numbers are yellow and sorted. The animation below shows how this sorting algorithm works:



 The video below also shows how selection sort works:

## VIDEO: SELECTION SORT

The video below shows the selection sort algorithm working. The computer is trying to sort the random list of bars in increasing height order. Each of the bars has a pitch associated with it: the higher the pitch, the taller the bar.

The red flashing bars are the bars that are being compared. You can see that the computer keeps track of the shortest bar *so far*, and it does not "remember" what the second, third, fourth... shortest bars are from the first time it ran through the list.

When the computer is at the end of the list, it moves the shortest bar it found to the correct position in the sorted list. The newest bar to be moved is highlighted in green. Again, note that the computer only adds to the sorted list--it never switches things around in the sorted list!

Note that the computer runs through all the bars at the end to double-check that everything is indeed sorted :)

This video does not run in real time: computers can actually sort lists *much* faster! Think about how your google search results return to you in sorted order in less than a second. There's a delay built in so that you can see each step; the delay gets shorter in the second half of the video.

## VIDEO: SELECTION SORT ILLUSTRATEDREVIEW: SELECTION SORT

**What is selection sort? How does the algorithm work?**

Now shuffle your [cards](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pdf), and have your partner put them in order from your least favorite to your most favorite using the selection sort algorithm. Remember that you can only have two face up cards at a time! Your partner can ask you if you prefer one over the other, but you can only answer with "yes" or "no." Keep track of the number of questions you asked on your [handout](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pdf)!

## WHAT IS BUBBLE SORT?

Our next sorting algorithm is bubble sort. The computer "marches through" the unsorted list and compares the numbers two at a time. The numbers being compared are boxed in red.

It does not care what the numbers are, and it doesn't keep track of what the values are. It just looks at the numbers two at a time, and if they're in the wrong order, it swaps them. Then the computer shifts one item down and keeps on comparing.

Notice that the largest number ends up on the right hand side, where the sorted list is forming and getting boxed in black. This algorithm is called bubble sort because it "bubbles" the largest number from the unsorted list into the sorted list.

The computer goes back to the beginning again and checks each consecutive pair of numbers, swapping them if necessary. It does this until it goes through the list and finds no more swaps to make. At that point, the whole list is sorted.



The video below demonstrates how the Bubble Sort algorithm works.

## VIDEO: BUBBLE SORT

The video below is a stop-motion animation that uses bubble sort to put LEGO towers in order of increasing height. The LEGO man stands on top of the tower, checks to see if it's taller than the tower next to him. If it is, the towers get swapped. Then, the LEGO man moves to the next spot and compares his tower with the next tower.

Notice again that the tallest unsorted tower gets "bubbled" to the right-hand side, where it joins the sorted towers.

## VIDEO: BUBBLE SORT WITH LEGOS

The video below shows a series of bars being sorted in increasing height order using the bubble sort algorithm. The computer compares two consecutive bars at a time. While bars are being compared, they are highlighted in red. If the bar on the left is taller than the bar on the right, the bars get swapped.

Notice how the red highlight "bubbles" through the list of bars and carries the tallest unsorted bar to the right-hand side. Each time the computer runs through the unsorted list, it finds the next biggest bar and drops it off where the sorted list begins.

Each bar has a pitch associated with it: the taller the bar, the higher the pitch. Remember that the animation has a delay built into it, and that the computer can actually sort much faster!

## VIDEO: BUBBLE SORT ILLUSTRATED

## REVIEW: BUBBLE SORT

**What is bubble sort? How does the algorithm work?**

Now shuffle your [cards](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pdf), and have your partner put them in order from your least favorite to your most favorite using the bubble sort algorithm. Remember that you can only have two face up cards at a time! Your partner can ask you if you prefer one over the other, but you can only answer with "yes" or "no." Keep track of the number of questions you asked on your [handout](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pdf)!

## INSERTION SORT

Insertion sort works by taking the first number out of the white unsorted list and inserting it into the black sorted list. The number you pulled out, which is boxed in red, is compared to each number in the sorted list one by one until it is compared to a number that is smaller.

We can think of an insertion sort also as a "pull-and-bubble" procedure. We pull a number out of the unsorted list and "bubble" it through the sorted list: we compare the red number with the black numbers one by one and swap them if they're in the wrong spots. We stop bubbling when the numbers we compare are in the right positions relative to each other.

This algorithm is called *insertion* sort because it takes a number and inserts it into the correct position in the sorted list.



Fun fact: many card players use insertion sort to put a deck into the correct order! They consider the cards one at a time, then places the card in its proper place among the cards they have already looked at. This splits the cards up between the ones that have already been sorted and the ones that haven't. Each time the player looks at a new card, he or she adds it to the ones that have already been sorted. When every card has been considered and placed, the deck is ordered!

Watch the Inserton Sort demonstration below:

## VIDEO: INSERTION SORT

The video below sorts the bars in order of increasing height. It grabs the first bar in the unsorted list, and inserts it into the correct spot into the sorted list.Each of the bars has a pitch associated with it: the higher the pitch, the taller the bar.

In this case, you can hear the unsorted bar being moved downwards through the sorted list and getting compared to every other bar, because the pitch gets lower and lower. When it finds a sorted bar that's smaller than the bar you grabbed, it'll insert the bar into the sorted list. At the end, the computer double-checks its sorting to make sure that it actually did put everything in the right place.

Remember that the animation is delayed, and that computers can actually sort much faster!

## VIDEO: INSERTION SORT ILLUSTRATED

## REVIEW: INSERTION SORT

**What is insertion sort? How does the algorithm work?**

Now shuffle your [cards](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pdf), and have your partner put them in order from your least favorite to your most favorite using the insertion sort algorithm. Remember that you can only have two face up cards at a time! Your partner can ask you if you prefer one over the other, but you can only answer with "yes" or "no." Keep track of the number of questions you asked on your [handout](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pdf)!

## MERGE SORT

The last sorting algorithm we will cover is called Merge Sort. Merge Sort is a bit more complicated than previous sorting algorithims. Merge Sort works by splitting the list you want to sort in half and sorting each half. How does it sort each half? By splitting them into quarters and sorting the quarters. We keep repeating this sorting until the lists only have one element, since one element lists are always sorted. We then combine the two sorted lists into a larger sorted list. We repeat this until we build up all the way into the entire list. This is called Merge Sort because we sort the list by *merging* two independently sorted lists.



Merge Sort seems like it is needlessly complicated relative to the other sorting algorithims we covered. However, this complexity comes with extreme benefits. Because we halve the size of the list repeatedly, we don't need to make as many comparisons, meaning this is a much faster algorithim. While the other algorithims might be easy to do with a small amount of things to sort, they are much slower with larger lists. The following table shows the amount of comparisons and time taken to sort a list with five thousand random numbers in it.

|  |  |  |
| --- | --- | --- |
| **Sorting 5000 random numbers** | **Comparisons** | **Time (seconds)** |
| **Selection Sort** | 12,497,500 | 2.1 |
| **Bubble Sort** | 12,497,500 | 3.79 |
| **Insertion Sort** | 2,261,683 | 0.489 |
| **Merge Sort** | 55,237 | 0.047 |

As you can see, Merge Sort is extremely fast compared to the other sorting algorithims.

Watch the video below on Merge Sort.

## VIDEO: MERGE SORT

The video below sorts the bars in order of increasing height. It grabs the first bar in the unsorted list, and inserts it into the correct spot into the sorted list.Each of the bars has a pitch associated with it: the higher the pitch, the taller the bar.

In this case, you can see the list being split into groups. Once it has the group sorted, it merges it with the group next to it and repeats until the entire list is sorted. At the end, the computer double-checks its sorting to make sure that it actually did put everything in the right place.

Remember that the animation is delayed, and that computers can actually sort much faster!

## VIDEO: MERGE SORT ILLUSTRATED

## REVIEW: MERGE SORT

**What is merge sort? How does the algorithm work?**

Now shuffle your [cards](http://www.muddx.com/c4x/HMC/MyCS/asset/Search_and_Sort_Food_and_Color_Cards.pdf), and have your partner put them in order from your least favorite to your most favorite using the merge sort algorithm. Remember that you can only have two face up cards at a time! Your partner can ask you if you prefer one over the other, but you can only answer with "yes" or "no." Keep track of the number of questions you asked on your [handout](http://www.muddx.com/c4x/HMC/MyCS/asset/Card_Sorting_Activity_Handout.pdf)!

## BONUS VIDEO: OBAMA ANSWERS A SORTING QUESTION!

## SEARCHING & SORTING WRAP-UP

We've learned about four different ways to sort things. We also know that sorting a lot of things takes a long time (much longer than a single linear search). We learned about two ways to search: linear searching and binary searching. Binary searching is much faster than linear searching, but the list has to be sorted first. Consider the follow scenarios, and discuss whether it would be better to have the computer sort the list and then use binary searches, or skip sorting and just do a linear search.

You have a list of 100 names, and need to search for 1 name.

You are making a database of all students in the state. Should you sort the list so that whenever someone tries to look up a name, they can use a fast binary search, or just leave the list unsorted and make everyone do a linear search every time they want to find a name?

## BONUS: SEARCHING & SORTING IN SCRATCH

Want to do more with algorithms? You can create Scratch software programs that use the searching and sorting algorithms we learned about. Here's a couple of Scratch programs that other people have made. Click on the images to go to the Scratch page. Please let us know if any of these projects have been taken down!

The Linear and Binary Search number-guessing games are a good way to compare the algorithms. In the linear search game, you can only know if the number you picked is the correct number, while the binary search game tells you if the number you picked is too high or too low. Try it out!

 

This Binary Search game lets you look up a word in its list of almost 75,000 words. Linear searching here would be really really tedious!!



The Bubble Sort Guy walks through the Bubble Sort algorithm step-by step to sort a list of numbers.



 The Sorting Algorithm Comparison program compares the Bubble Sort, Selection Sort, and Insertion Sort algorithms to see how quickly each of them can sort the list. The author also created two sorting algorithms: Loop Sort and Thread Sort. Computer scientists are always looking for new algorithms to complete tasks more quickly!



You can also try searching for more cool Scratch programs. There are lots!