LESSON 6:

Insertion Sort

AP COMPUTER SCIENCE

UNIT 7

edhesive

Learning Objectives

- Understand how the insertion sort algorithm orders an array of data
- Understand the advantages and disadvantages of using the insertion sort algorithm to order the elements of an array

Goal:

Put the elements of the array in **order**

In this sort we move through the array and insert each element in the correct position in the array

Lesson 6: Insertion Sort

56	34	55	12	88	37	45	12

Steps:

Start with a pointer variable `next = 1;`

Going backwards, move through the elements until you find the position of an element less than the value at next.

Insert `array[next]` at this location.

Increment `next`.

Repeat until `next =` ____________.

After the first 3 repetitions of the insertion sort what would the array hold?

21	17	60	20	56	12
0	1	2	3	4	5

Insertion Sort Implementation

```
for (int j = 1; j < elements.length; j++)
{
   int temp = elements[j];
   int possibleIndex = j;
   while (possibleIndex > 0 && temp < elements[possibleIndex - 1])
   {
      elements[possibleIndex] = elements[possibleIndex - 1];
      possibleIndex--;
   }
   elements[possibleIndex] = temp;
}
```

Advantages:

Easy to code and understand

Disadvantages:

Slow for large datasets

Comparing Sort Times

Insertion Sort and **Selection Sort** perform differently on different data sets.

To do a thorough comparison, add an **execution count** variable, and run the two algorithms on a variety of different arrays of different sizes.