From the Teacher: K. Evans

Class: AP Computer Science A

Periods: 1 and 3

Assignment: Week 1

If turning in paper packet and work, make sure to include this header information on all pages!

From the Student:
Student Name
Teacher Name
Name of class
Period #
Assignment #

Distance Learning 2020 Week 1

AP Exam Update, Binary Search, & Merge Sort

Assignments (not from Edhesive) are accessible in Microsoft Teams on Office 365. Work can also be submitted in Teams, which I highly encourage you to do if you are able to. You can contact Ms. Evans if you need help with Teams. You must write your name in pen on each page of your assignment.

The work in this packet is not officially due until 5/8/2020. However, I have broken down the work into daily chunks to help you manage your time. I encourage you to have the work from week 1 complete by 4/24/2020. New assignments for weeks 2 and 3 will be given that date.

My office hours are 1 pm – 3 pm, M–F. You can reach me through Remind (class code: @evans-csa), email (kevans@tusd.net) or chat on Teams. Please continue to check your email regularly.

Ms. Evans will be holding a half hour meeting on Microsoft Teams to talk about the algorithms in the notes for the week and answer questions on **Tuesday 4/21**. Check in Teams in the posts or the calendar to find the exact time.

Week 1: Day 1 (turn in by 5/8/2020): Review information about the updates to the AP Computer Science A exam.

1. Read over the AP Computer Science Exam Updates PowerPoint slides.
2. Write 1-2 thoughtful paragraphs about what this means for you, the rest of the year, and what questions you have. (**Assignment #1**)

*If turning in work on Teams, you can type up your answer and upload the file. Or, you can write your answer on binder paper and then upload a picture of it. Please write your name in pen on each page before you take a picture. Make sure your picture is clear and readable.

Week 1: Day 2 (turn in by 5/8/2020): Lesson 17, Binary Search

Online: Complete Lesson 17 on Edhesive. Make sure to watch the lesson video and read the lesson summary.

Offline: Read over notes on Lesson 17. Then complete **Assignment #2**, Lesson 17 problems.

Week 1: Day 3 (turn in by 5/8/2020): Lesson 18, Merge Sort

Online: Complete Lesson 18 on Edhesive. Make sure to watch the lesson video and read the lesson summary.

Offline: Read over notes on Lesson 18. Then complete **Assignment #3**, Lesson 18 problems.

Week 1: Day 4-5 (turn in by 5/8/2020): More Practice

Online: Complete Unit 7 Progress Check (all parts) on college board AP Classroom.

Offline: Write up a summary of all the algorithms for Arrays that we learned in Unit 7. (Lessons 13-18)

AP Computer Science Exam Updates PowerPoint

AP COMPUTER SCIENCE A

2020 AP Exam
Friday, May 15 @ 1:00PM PST

New Testing Schedule

Exam Start Times: Local times may vary depending on a student's geographic location.	Hawaii Time: 6:00 a.m. Alaska Time: 8:00 a.m. Pacific Time: 9:00 a.m. Mountain Time: 10:00 a.m. Central Time: 11:00 a.m. Eastern Time: 12:00 p.m. Greenwich Mean Time: 4:00 p.m.	Hawaii Time: 8:00 a.m. Alaska Time: 10:00 a.m. Pacific Time: 11:00 a.m. Mountain Time: 12:00 p.m. Central Time: 1:00 p.m. Eastern Time: 2:00 p.m. Greenwich Mean Time: 6:00 p.m.	Hawaii Time: 10:00 a.m. Alaska Time: 12:00 p.m. Pacific Time: 1:00 p.m. Mountain Time: 2:00 p.m. Central Time: 3:00 p.m. Eastern Time: 4:00 p.m. Greenwich Mean Time: 8:00 p.m.
Mon, May 11	Physics C: Mechanics	Physics C: Electricity and Magnetism	United States Government and Politics
Tues, May 12	Latin	Calculus AB Calculus BC	Human Geography
Wed, May 13	Physics 2: Algebra-Based	English Literature and Composition	European History
Thurs, May 14	Spanish Literature and Culture	Chemistry	Physics 1: Algebra-Based
Fri, May 15	Art History	United States History	Computer Science A
Mon, May 18	Chinese Language and Culture	Biology	Environmental Science
Tues, May 19	Music Theory	Psychology	Japanese Language and Culture Italian Language and Culture
Wed, May 20	German Language and Culture	English Language and Composition	Microeconomics
Thurs, May 21	French Language and Culture	World History: Modern	Macroeconomics
Fri, May 22	Comparative Government and Politics	Statistics	Spanish Language and Culture

New Test Format will be …

- 25 min FRQ
- 5 min submit Answer
- 15 min FRQ
- 5 min sumit answer

45 Minutes Total

What Technology may I use?

Phone
iPad
Tablet
Laptop
Desktop

If you know of students who have a need for technology or connectivity in order to take their AP Exam, please
contact Mrs. Rotondi ASAP
College Board has to know before April 24.

TWO OPTIONS

1. Type your answers on your electronic device
2. Handwrite your answers, take a photo with your phone, and submit using your electronic device

25 Minute FRQ

Worth 65% of score

Free-response question 3: Array / ArrayList the question will asses your ability to do the following:

- Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.
- Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects

15 Minute Question

Worth 35% of score

Free-response question 1: Methods and Control Structures the question will asses your ability to do the following:

- Write program code to create objects of a class and call methods.
- Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.

Other Information

- An Interactive Development Environment (IDE) or compiler (for example Dr. Java or repl.it) is not required for the 2020 AP Computer Science A Exam, and students will not have an advantage if they use one. However, students may use an IDE or compiler if they choose.

General Exam Features

- Like many college-level exams, this year's AP Exams will be open book/open note. Get tips for taking open book/open note exams.
- Students will be able to take exams on any device they have access to—computer, tablet, or smartphone. They will be able to either type and upload their responses or write responses by hand and submit a photo via their cell phone.
- In late April, information on how to access the testing system on test day, and video demonstrations so that students can familiarize themselves with the system will be released.
- College Board has free online classes/review videos on YouTube, organized by subject and Topic: https://www.youtube.com/user/advancedplacement

AP Computer Science A Testing Dates

Friday, May 15, 2020 @ 1:00 PM
From your home on your device

Makeup Date
Friday, June 3, 2020 @ 11:00 AM
ONLY if you have a conflict with May 18
Must be validated by your school

We Have Four WEEKS
(From Mon 4/20)

- Learn the remaining lessons needed for the exam (Lessons 17 & 18)
- Review the key concepts we have studied
- Take advantage of the Personal Progress Checks in AP Classroom
- Utilize multiple practice exams that mimic the current exam structure over the next six weeks

Week 1: Lesson 17 Notes

Lesson 17: Binary Search

This lesson covers a search algorithm called the binary search.

If we were to search for a particular word in the dictionary using the linear search we covered previously, we might be reading words for a very long time--especially if our target word is towards the end of the alphabet.

However, a binary search can find our target word in a much shorter amount of time. A binary search finds the element in the exact middle of our dataset and compares that value to our target value. If the midpoint value is too low, we know we don't need to look at any values from the lower half of the set because they will also be too low. Just like that, we've eliminated a full half of our dataset. With the other half, we can again split that part in half and run another comparison. Each time we do this, we eliminate half of the remaining possibilities, closing in on the target with exponential speed until we find it.

There is one rule for using a binary search: the dataset we're searching through *must already be sorted*. If it's not, we can't be sure that we haven't eliminated the entry we're looking for when we cut out half the results.

To execute the binary search, we first need to find the midpoint of our dataset. If we call the upper limit of the set "high" and the lower limit "low", then the middle point mid = (low + high)/2. Grabbing the value at this index, we can compare it to our target value. If our target value is too low compared to the midpoint value, we repeat the process with our new high set to mid-1. If our target value is too high, we repeat with low set to mid+1. To summarize:

1. Find the value at midpoint mid = (low + high)/2
2. If our target value is:
    1. Too low: repeat with high = mid-1
    2. Too high: repeat with low = mid+1
3. Repeat until we find our match, or high < low (which means our target does not exist in the set)

In terms of code, there are several ways you can code a binary search. You can do this *iteratively*, using a for loop or a while loop, or *recursively*, having your method call itself.

Compared to a linear search, the binary search has some distinct advantages and disadvantages. Especially over large sets of data, the binary search can be very fast. However, it can be a little tricky to code, and the dataset you're working on must already be sorted (otherwise the binary search won't work at all!).

Advantage	**Disadvantage**
• Very fast	• More difficult to code • Array must be sorted

For the AP Exam, it is important to understand the differences between our two types of search: the linear search and the binary search.

Week 1: Assignment #2 (Lesson 17)

Fast Start

Question 1 1 pts

For the following array, what is the maximum number of comparisons a linear search will need to do?

```
int[] a= {2, 8, 59, 20, 128, 3, 6, 3, 7, 1};
```

- 55
- 10
- 9
- 45

Review

Question 1 1 pts

Consider the array declaration that follows.

```
int[] a = {1, 3, 5, 6, 7, 8, 9, 11, 13, 14, 15};
```

How many comparisons would it take if you were to search the array for 3 using a binary search?

- 4
- 2
- 11
- 3

Coding Activity

For this activity, write a class called Lesson_17_Activity which includes the following methods:

- `public static boolean isSorted(int [] a)` - returns true if the array a is in ascending order and false otherwise.
- `public static int binarySearch(int [] a, int b)` - performs a binary search to determine if the value b is stored in a. If b is found, return the index of b. Otherwise, return -1.

Include both methods in a **class called Lesson_17_Activity** and write the entire class on your paper.

AP MC Review

Consider the following instance variable and method.

```
private List vals;
       // vals is sorted into increasing order

public void binaryInsert(int num)
{
  int low = 0;
  int high = vals.size();
  while(high > low)
  {
    int mid = (low + high) / 2;         /* calculate midpoint */
    if(vals.get(mid).intValue() < num)
    {
      low = mid + 1;
    }
    else
    {
      high = mid;
    }
          }
          vals.insert(low,new Integer(num)); /* insert new number */
}
```

The binaryInsert method creates inserts an Integer element into the correct position of the list vals which is sorted into increasing order. Suppose vals contains the following Integer values: [2, 3, 4, 4, 4, 8], and that the following command is executed:

```
binaryInsert(4);
```

What will be the value of low when the line marked /* insert new number */ in the binaryInsert method is executed?

Week 1: Lesson 18 Notes

Lesson 18: Merge Sort

This lesson covers the last of the three sorts you will need to know for the AP exam: the merge sort.

So far, our two other sorts--the selection sort and the insertion sort--have been relatively easy to code and understand, but are very slow when dealing with large datasets. The merge sort, on the other hand, is often written recursively, which has the advantage of being very fast on large datasets.

In general, the merge sort works by breaking down an array into smaller and smaller pieces until each piece consists of one element, and then rebuilding the array by comparing the pieces to each other.

It is easiest to view the merge sort as two processes: the breaking down into smaller pieces, and then the remerging.

Each step of the breaking down phase chops our array in half. Let’s say we have an array of 8 numbers:

```
[56, 34, 55, 12, 88, 37, 45, 21]
```

The first step will chop the array in half at the middle to make 2 arrays of 4 elements each:

```
[56, 34, 55, 12]  [88, 37, 45, 21]
```

Next, each 4-element array will be split in half to 2 elements, and then each 2-element array split down into arrays of single elements:

```
[56, 34]  [55, 12]  [88, 37]  [45, 21]
[56] [34] [55] [12] [88] [37] [45] [21]
```

Now that the array has been split into its single pieces, the real magic can begin: the merging.

The important part to remember when merging is that each step of the merge takes only two of the smaller arrays and merges them to completion, then moves on to just the next two arrays. So, the first stage of merging above starts with [56] and [34]. The merge takes the smallest one first and adds it to a bigger array first. Out of these two arrays, [34] is the smallest, so it gets added to the new array first.

```
[34, 56]
```

Then, we repeat with the next few pair of arrays:

```
[34, 56]  [12, 55]  [37, 88]  [21, 21]
```

At this point, we again merge only two arrays at one time to make a single bigger array. The trick at this point is to remember that once you add an element from one of the smaller arrays, the “pointer” moves one spot forward in that array only, and you compare the new number with the same number from the other array. To understand this, let’s just look at the first two arrays in our current set: [34, 56] and [12, 55].

The “pointer” starts off at the first spot in each, so our first comparison is between 34 and 12. Since 12 is smaller, it gets added to the new array first. Now, the “pointer” in our second smaller array moves to the 55, but our other pointer is still on the 34. Thus, the next comparison is between 34 and 55. Since 34 is smaller, it gets added to the new array, and that “pointer” moves to the 56. We repeat for 55 and 56, adding the 55, and then adding the 56 because it’s the only one left.

```
Old arrays: [34, 56]  [12, 55]
New array: [12, 34, 55, 56]
```

Repeating this process with the other half of our origin array, we have the following:

```
[88] [37] [45] [21]

[37, 88]  [21, 45]

[21, 37, 45, 88]
```

Now, our two working arrays are `[12, 34, 55, 56]` and `[21, 37, 45, 88]`. Note that each of these arrays is already in proper sorted order. Performing the merge again, we end up with our final sorted array:

```
[12, 21, 34, 37, 45, 55, 56, 88]
```

When tracing the behavior of the merging phases, it may be easiest to cross out each element from the smaller arrays as you add it to a bigger array. This can help ensure that you know where your pointer is, and that the elements are merged in the proper order.

As you have probably gathered, the processes involved in the merge sort can be a little complicated. Like the other sorts, it is highly unlikely you'll be asked to actually code the merge sort on the AP exam from scratch; however, it is important to understand how it works for multiple choice questions.

Remember: the merge sort, while complicated, works incredibly fast on large datasets because of its recursive nature.

Advantage	**Disadvantage**
Very fast, especially for large datasets (recursive)	Multiple parts can be difficult to code

Week 1: Assignment #3 (Lesson 18)

Fast Start

Question 1 — 1 pts

To use a binary search with an array, the array element values must be _____.

- instantiated
- sorted
- class
- object
- primitive

Question 2 — 1 pts

Consider the following method:

```
public static void mystery(int a) {
    if (a % 10 != 0) {
        mystery (a + 1);
    }
    System.out.print(a + " ");
}
```

What is output by mystery(8)

- 10 9
- 10 9 8 7
- 9 8
- 9 8 7
- 10 9 8

Review

Question 1 — 1 pts

Which of the following is the first step of merge sort?

- Sort the subarray
- Merge all the subarrays together
- Split the array into smaller subarrays

Question 2 — 1 pts

You need to sort a database of 35 million bank records. Which sort would be most effective?

- merge sort
- selection sort
- insertion sort
- linear search

AP MC Review

The following sort method correctly sorts the integers in elements into ascending order.

```
Line 1:  public static void sort(int[] elements)
Line 2:  {
Line 3:    for (int j = 1; j < elements.length; j++)
Line 4:    {
Line 5:      int temp = elements[j];
Line 6:      int possibleIndex = j;
Line 7:      while (possibleIndex > 0 && temp <
elements[possibleIndex - 1])
Line 8:      {
Line 9:        elements[possibleIndex] = elements[possible
Index - 1];
Line 10:       possibleIndex--;
Line 11:     }
Line 12:     elements[possibleIndex] = temp;
Line 13:   }
Line 14: }
```

Consider the following three proposed changes to the code:

```
Change 1
Replace line 3 with:
Line 3:    for (int j = elements.length - 2; j >= 0; j--)
```

```
Change 2
Replace line 7 with:
Line 7:      while (possibleIndex > 0 && temp >
elements[possibleIndex - 1])
```

```
Change 3
Replace line 7 with:
Line 7:      while (possibleIndex < elements.length - 1 &&
temp <
elements[possibleIndex + 1])
and replace lines 9-10 with:
Line 9:        elements[possibleIndex] = elements[possible
Index + 1];
Line 10:       possibleIndex++;
```

Suppose that you wish to change the code so that it correctly sorts the integers in elements into descending order rather than ascending order. Which of the following best describes which combinations of the proposed changes would achieve this goal?

- Enacting changes 1 and 3 together, or enacting change 2 by itself
- Enacting any of the three changes individually
- Enacting changes 1 and 2 together, or enacting change 3 by itself
- ONLY enacting changes 1 and 2 together
- ONLY enacting change 2 by itself

Unit 7 Review – AP MC Review

Question 1

1 pts

Consider the following sort method. This method correctly sorts the elements of array `arr` into increasing order.

```
public static void sort(int[] data)
{
    for (int j = arr.length - 2; j >= 0; j--)
        {
            int move = arr[j];
            int k = j + 1;

            while (k < arr.length && move > arr[k])
            {
                arr[k - 1] = arr[k];            /* Shuffle elements upwards */
                k++;
            }

            arr[k - 1] = move;                  /* Insert  value into position */
        /* end of for loop */
    }
}
```

Assume that `sort` is called with the array `{5, 4, 3, 2, 1}`. How many times will the expression indicated by `/* Shuffle elements upwards */` and the statement indicated by `/* Insert value into position */` execute?

- Shuffle elements upwards: 14, Insert value into position: 4
- Shuffle elements upwards: 15, Insert value into position: 5
- Shuffle elements upwards: 10, Insert value into position: 5
- Shuffle elements upwards: 10, Insert value into position: 4
- Shuffle elements upwards: 15, Insert value into position: 4

Question 2

1 pts

Consider the following instance variable and method.

```
private List<Integer> vals;
// vals is sorted into increasing order

public void binaryInsert(int num)
{
    int low = 0;
    int high = vals.size();
    while(high > low)
    {
        int mid = (low + high) / 2;            /* calculate midpoint */
        if(vals.get(mid).intValue() < num)
        {
            low = mid + 1;
        }
        else
        {
            high = mid;
        }
    }
    vals.insert(low,new Integer(num)); /* insert new number */
}
```

The `binaryInsert` method creates inserts an `Integer` element into the correct position of the list `vals` which is sorted into increasing order.

Suppose `vals` contains 50 `Integer` values representing the numbers from 0 to 49 inclusive once each, and that the following command is executed:

```
binaryInsert(50);
```

How many times will the line marked `/* calculate midpoint */` in the `binaryInsert` method be executed?

- 6
- 50
- 51
- 5
- 25