HOMEWORK 2

100 points

DUE DATE: February 2nd by 9:00pm.

Homework 2 Deliverables:

- Create a folder named as **hw2_yournetid**
- Place the following files under **hw2_yournetid** folder into a zip and upload the zip file through dropbox on D2L. The zip file should include:
    1. KthSmallest.h
    2. Problem2.pdf

This is not a team work, do not copy somebody else's work.

Problem 1

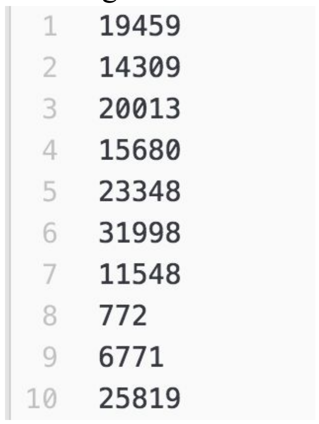
Suppose you have a group of N numbers and would like to determine the kth smallest in an array (or a vector they are both referred as data collections).

You will use 2 different approaches to solve "**finding the kth smallest element**" in a collection.

1. **Approach 1:** Read all N numbers into an array (or a vector), sort the array in an increasing order using **insertion sort**. (You can use the following pseudocode that we already covered in class and Homework 1 and convert that into a C++ code or write your own insertion sort)

```
INSERTION-SORT(A)                                          cost   times
for $j \leftarrow 2$ to $n$                                $c_1$  $n$
    do $key \leftarrow A[j]$                               $c_2$  $n-1$
       ▷ Insert $A[j]$ into the sorted sequence $A[1..j-1]$.  0   $n-1$
       $i \leftarrow j-1$                                  $c_4$  $n-1$
       while $i > 0$ and $A[i] > key$                      $c_5$  $\sum_{j=2}^{n} t_j$
           do $A[i+1] \leftarrow A[i]$                     $c_6$  $\sum_{j=2}^{n}(t_j-1)$
              $i \leftarrow i-1$                           $c_7$  $\sum_{j=2}^{n}(t_j-1)$
       $A[i+1] \leftarrow key$                             $c_8$  $n-1$
```

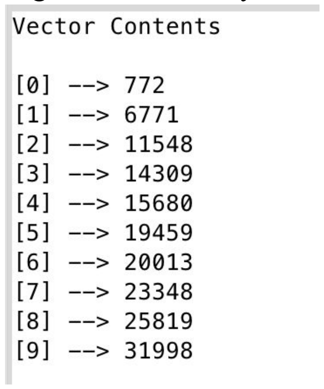
For example assume a given data file:



```
19459
14309
20013
15680
23348
31998
11548
772
6771
25819
```

Then find its kth smallest element (let’s say 4th smallest element).

Also please note that: In this assignment, you must understand the user does not know that indexing starts from 0. So when the user requests for you to find the 4th smallest value as a programmer you need to retrieve the element at the 3rd index.
Once sorted, using insertion sort, your array(or your vector) will look like the following.



```
Vector Contents

[0] --> 772
[1] --> 6771
[2] --> 11548
[3] --> 14309
[4] --> 15680
[5] --> 19459
[6] --> 20013
[7] --> 23348
[8] --> 25819
[9] --> 31998
```

In this case 4th smallest values is : 14309 (Value located at 3rd index).

2. **Approach 2:** This approach may be a lot better than Approach 1.
   - Use of a temporary array (or a vector).
   - Using a temporary array (or a vector), store the first k elements into a temp array (or a vector) and find its max_element.
   - Remaining element from the array[k] to array[n-1] will be checked against this max.
   - When the program finds that array[i]th element is less than the current max , then it must override this value in that temparray where the max value was found.
   - A new max value in the temp array must be calculated each time the max value is overridden.
   - Once every element has been checked, you should have the k smallest elements in the temporary array, but they will not be sorted.
   - Now, Sort only the **temporary array** in increasing order and the kth smallest element will be at index k-1(reminder: indexing starts at 0).
   - This method is better when data size (n) is large and k is small, because it only involves sorting k elements instead of all n elements.

Below is a sample run using the same input file shown for Approach 1:

```
Approach 1: Using Insertion Sort for finding kth smallest element
4 th smallest value in vector -->  14309


Size of the vector : 10
Insertion Sort Search method  took 122263 nanoseconds



Approach 2: Using Temp Array for finding kth smallest element
Vector Contents

[0] --> 772
[1] --> 6771
[2] --> 11548
[3] --> 14309
4 th smallest value in vector -->  14309


Size of the vector : 10
50964 nanoseconds
```

An incomplete KthSmallest.h is provided for you along with

- main.cpp
- inputfiles --> f100, f1000, f10000, f100000
- Makefile

- Hint: Start with a real small data file (like the one shown above) first to test your logic.
- Test your code on our servers (black or arctic) upon submission and make sure that your code compiles on our servers to avoid penalty deductions.
- Remember we will need your KthSmallest.h when you are ready to upload your work on D2L.

Problem 2

Your job is to test your insertion sort and kth smallest performance for different input sizes(n) of random values.

1. Using a text editor (Word, Google Doc, ...etc) Create a table and enter the the running time of your program for insertionSort and KthSmallest methods for n values provided in given input files

2. Using this table you, please provide a **graph** of run time(in nano or milliseconds) vs n values (You can use Excel or any other software to draw a graph) . Copy your chart and paste it on a text document and convert this document to PDF for submission. We will only accept PDF format. Make sure to convert your document to PDF prior to submission.