Jalil Farid
Programming Assignment 1

1. For search, let n be the size of the large array, and $\sqrt{n}$ be the size of the small array. The program first searches the $\sqrt{n}$ array in $O(log(\sqrt{n})) = O(\frac{1}{2}log(n)) = O(log(n))$ time, then the search continues in the large array which contains at most n values, searching in $O(log(n))$ time, giving a total time complexity of $O(log(n))$.

2. For insert, let n be the size of the large array, and $\sqrt{n}$ be the size of the small array. This proof will show by potential method that the amoritized cost of each insert is $O(\sqrt{n})$. Let $\Phi(s_i) = (\sqrt{i})^2 =\{$ number of values in of the small array squared $\}$. For simplicity, we will note the difference between potentials as $\Delta\Phi(s_i) = \Phi(s_i) - \Phi(s_{i-1})$. For a basic insert which does not overflow the small array, assume there are i-1 values in the small array. At this point the total insertion cost using insertion sort is $\sqrt{i}$ on the ith step. For

$$\Delta\Phi(s_i) = (\sqrt{i} + 1)^2 - (\sqrt{i})^2 = i^2 + 2(\sqrt{i}) + 1 - i^2 = 2(\sqrt{i}) + 1$$

Note also that for any i < n, $\sqrt{i} < \sqrt{n}$. So the potential cost c_i on the ith step is then

$$c_i = \sqrt{i} + \Delta\Phi(s_i) = 2\sqrt{i} + 1 = O(\sqrt{n})$$

For an insert step that overflows, the total number of values in the large array at this is i. Now from the program, we first insert the element into the small array for at most $\sqrt{i}$ operations, then merge with i. So the time complexity of this operation then be $i + \sqrt{i}$. Note also $\Delta\Phi(s_i) = -(i)^2 = i$. So the potential cost c_i on the ith step which is a merge insert operation is then

$$c_i = i + \sqrt{i} + \Delta\Phi(s_i) = i + \sqrt{i} - i = \sqrt{i} = O(\sqrt{n})$$

So this proof has shown each operation has an amoritized cost of $O(\sqrt{n})$.