*9.7. 6. Insertion sort goes sequentially through the array when making comparisons to find a proper place for an element currently processed. Consider using binary search instead and give a complexity of the resulting insertion sort.*

The time complexity would change from O(n^2) to O(n log n) because where in the original insertion sort the program must check each new value against all other values and the potentially move them N times. Though with the binary insertion sort the comparison is done through a binary search, reducing the number of comparisons because half of the possible selections are removed with each iteration of the binary search. Instead of O(N) for the inner search time we have O(log N) though for the exterior loop it is still O(N) regardless.

Example of Insertion Sort algorithm:

//Incorrect capital letters are the fault of Microsoft Word and it’s auto capitalization.

Void insertionSort (int array[]){

Int n = array.length();

For (int x = 1; x < n; ++x){

Int key = array[I];

Int j = I – 1;

//Interior loop to compare and move every greater element one position to the right

While (j >= 0 && array[j] > key)

{

array[j + 1] = array[j];

j = j – 1;

}

array[j + 1] = key; } }

Example of Binary Insertion Sort algorithm:

//Main insertion sort block

void insertionSort(int a[], int n) {

int i, loc, j, k, selected;

for (i = 1; i < n; ++i){

j = i - 1;

selected = a[i];

// find location where selected should be inserted

loc = binarySearch(a, selected, 0, j);

// Move all elements after location to create space

while (j >= loc){

a[j+1] = a[j];

j--;}

a[j+1] = selected; } }

//Binary search helper

int binarySearch(int a[], int item, int low, int high) {

if (high <= low)

return (item > a[low])? (low + 1): low;

int mid = (low + high)/2;

if(item == a[mid])

return mid+1;

if(item > a[mid])

return binarySearch(a, item, mid+1, high);

return binarySearch(a, item, low, mid-1);

}

References

<https://www.tutorialspoint.com/data_structures_algorithms/insertion_sort_algorithm.htm>

<http://www.geeksforgeeks.org/insertion-sort/>

<http://www.geeksforgeeks.org/binary-insertion-sort/>