Project 5

SORTING

Problem 1: insertion Sort

- Download the file insertion_sort.py and try to understand the insertion sort algorithm which is implemented there
- Write an English description (or Hebrew if you must to …) which outlines the insertion sort algorithm
- Use the module sort_bench.py to conduct a run time benchmark of this algorithm like we did in class (use the same list sizes and other parameters)

Problem 2: Heap Sort

- A list L of integers is called a heap if it satisfies the heap property:
  For every k: L[k] <= L[*2k+1] and L[k] <= L[2*k+2]
  Read more: http://docs.python.org/2/library/heapq.html
- Download the file heap_sort.py and try to understand the heap sort algorithm which is implemented there
- Write an English description (or Hebrew if you must to …) which outlines the heap sort algorithm
- Use the module sort_bench.py to conduct a run time benchmark of this algorithm like we did in class (use the same list sizes and other parameters)

Problem 3: Remove Duplicates

- Describe and analyze an efficient method for removing all duplicates from a list *L* of *n* elements
- After removing duplicates, the remaining elements should retain the order they had before
- Exmaple:

```
L = [7,2,2,5,7,2,1,7,3]
remove_dups(L) => [7,2,5,1,3]

L = [5,0,1,0,9,2,1,0,5]
remove_dups(L) => [5,0,1,9,2]
```

Problem 4: has_dup

- Given a list L of *n* integers, write an efficient algorithm **has_dup** for determining whether there are two equal elements in *L*
- What is the running time of your method? Is it the best running time possible?
- Examples:

```
L = [5,1,0,4,2,9,7,4,3]
has_dup(L) => True
L = [1,2,3]
has_dup(L) => False
```

Problem 5: is_elements_sum

- Let *A* and *B* be two lists of *n* integers each. Given an integer *m*, write an *O*(*n*log *n*)-time algorithm **is_elements_sum** for determining if there is an integer *a* in *A* and an integer *b* in *B* such that *m* = *a*+*b*

```
A = [5,1,0,4,2,9]
B = [2,4,0,7,1,8]
m = 13
is_elements_sum(m, A, B)
=> True!  13 = 9 + 4  (or 5+8)
is_elements_sum(18, A, B)
=> False
```

Problem 6

- If L is a list of n integers smaller than n**3 then it can be sorted in O(n) time
- Find such an algorithm and prove that its time complexity is O(n)
- Hint: look again on Radix sort