Radix Sort

Use radix sort to sort the following elements, show each pass.

\[ ['a', 'get', 'zen', 'ah', 'row', 'rat', 'tie'] \]

\[
\begin{array}{cccccccc}
\text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} \\
\text{g} \rightarrow & \text{a} & \text{g} \rightarrow & \text{a} & \text{g} \rightarrow & \text{a} & \text{g} \rightarrow & \text{a} \\
\text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} \\
\text{n} & \text{n} & \text{n} & \text{n} & \text{n} & \text{n} & \text{n} & \text{n} \\
\text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} \\
\text{i} & \text{i} & \text{i} & \text{i} & \text{i} & \text{i} & \text{i} & \text{i} \\
\text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} & \text{e} \\
\text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} & \text{r} \\
\text{w} & \text{w} & \text{w} & \text{w} & \text{w} & \text{w} & \text{w} & \text{w} \\
\text{t} & \text{t} & \text{t} & \text{t} & \text{t} & \text{t} & \text{t} & \text{t} \\
\text{z} & \text{z} & \text{z} & \text{z} & \text{z} & \text{z} & \text{z} & \text{z} \\
\end{array}
\]
Interrupting sorts

The following arrays have been interrupted in the middle of a sorting algorithm. Use your knowledge of comparison based sorting to determine which algorithm was being used on each array. Each of the following will be present exactly once: Heap Sort, Insertion Sort, Selection Sort, Merge Sort

Array:

-5  2  19  53  44  91  87  35
14  42  17  72  12  10  5  1
29  35  44  114  37  30  28  46
6  10  3  50  15  60  1  34

Sort Used:

Selection
starts w/ global sort
Heap Sort
Insertion
Merge Sort
sorted subsequence
Short answer

1. Suppose we are trying to perform a radix sort on a set of positive `java int`. Recall that the runtime of Radix sort is $O(P(B + n))$ where $B$ is the Radix and $P$ is the number of passes. To what extent are $P$ and $B$ selectable and how might differing values of $n$ impact the selection process.

   If $n$ is very large, we can let $B$ be larger.

   This increases the radix, but decreases the number of passes.

2. Provide an example for each of the following or indicate why no sort exists:
   - A stable comparison sort with worst-case $O(n^2)$ runtime
     
     **Insertion**
   - A stable comparison sort with worst-case $O(n)$ runtime
     
     **not possible, comparison sorts are $O(n \log n)$**
   - A $O(n \log n)$ stable comparison sort
     
     **merge sort ($O(n)$ memory usage, though)**
   - A stable, in-place, comparison sort that runs in $O(n \log n)$ time
     
     **not possible**
     
     quick sort is not stable, 
     merge is not in-place
     
     comparison $\log n$ an only the above 3
Graph Introduction

Use this graph and help from the TA to introduce our next topic: graphs.

- Identify Vertices and Edges. \( G(V,E) \)
- Discuss directed vs. undirected graphs
- Discuss weights
- What sort of information could this graph represent?

\[ V = \{ A, B, C, D, E, F \} \]
\[ E = \{ (A,B), (A,D), (A,C), (B,F), (C,D), (C,E), (C,G), (D,F), (E,F) \} \]

Maps, networks, traffic, genealogy, social connections.

Symmetry: if \( (A,B) \) then \( (B,A) \)