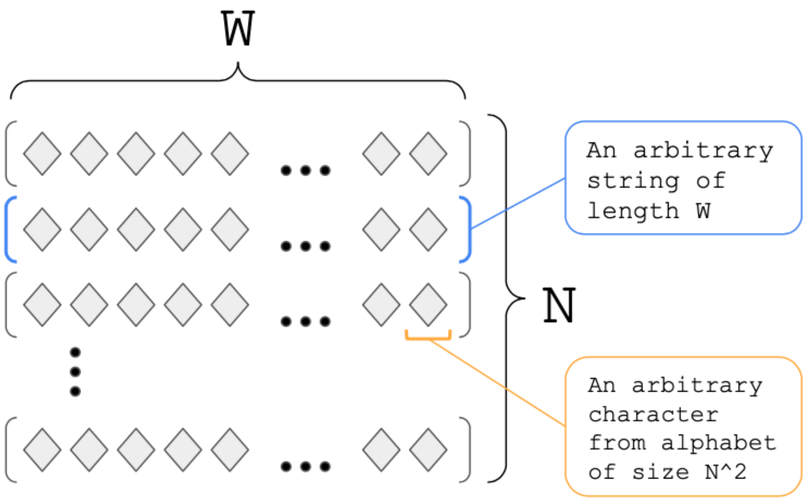
Aliens Arrive at Moon Base

a) Suppose the aliens have received a landing code as a series of strings, but it's all scrambled up! Help them sort the strings so they can land on the moon base! More specifically, suppose we want to **sort** N **distinct** Strings, each of length W, that are comprised of characters from an alien alphabet of size N^2. We can visualize the **set of strings** to sort as the image below.



What are the runtimes of the following sorting algorithms, in the best and worst case? If we use some sorting algorithm, say quicksort, as the subroutine in LSD/MSD sort, that means we use quicksort to sort each *character*. If two Strings are distinct, that means that **at least** one character differs. Finally, if the sorting algorithm provided **would not work, select "Would Not Work" for the best and worst case**. Your answers may involve both W and N.

Hint: When we compare two strings, it takes constant time in the best case, and W time in the worst case.

i) Merge sort

Best Case: ○ $\Theta(1)$ ○ $\Theta(logN)$ ○ $\Theta(N)$ ○ $\Theta(NlogN)$ ○ $\Theta(N^2)$ ○ $\Theta(N^2)logN)$ ○ $\Theta(logW)$ ○ $\Theta(W)$ ○ $\Theta(WlogW)$ ○ $\Theta(W^2)$ ○ $\Theta(W^2logW)$ ○ $\Theta(WlogN)$ ○ $\Theta(WN)$ ○ $\Theta(WNlogN)$ ○ $\Theta(WN^2)$ ○ $\Theta(WN^2logN)$ ○ $\Theta(NlogW)$ ○ $\Theta(NWlogW)$ ○ $\Theta(NW^2)$ ○ $\Theta(NW^2logW)$ ○ $\Theta(N^2W^2)$ ○ Never terminates (infinite loop) ○ None of the above Would Not Work

Worst Case:
○ $\Theta(1)$ ○ $\Theta(logN)$ ○ $\Theta(N)$ ○ $\Theta(NlogN)$ ○ $\Theta(N^2)$ ○ $\Theta(N^2)logN)$ ○ $\Theta(logW)$ ○ $\Theta(W)$ ○ $\Theta(WlogW)$ ○ $\Theta(W^2)$ ○ $\Theta(W^2logW)$ ○ $\Theta(WlogN)$ ○ $\Theta(WN)$ ○ $\Theta(WNlogN)$ ○ $\Theta(WN^2)$ ○ $\Theta(WN^2logN)$ ○ $\Theta(NlogW)$ ○ $\Theta(NWlogW)$ ○ $\Theta(NW^2)$ ○ $\Theta(NW^2logW)$ ○ $\Theta(N^2W^2)$ ○ Never terminates (infinite loop) ○ None of the above Would Not Work

ii) Insertion sort

Best Case:

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work

Worst Case:

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work

iii) MSD sort with heap sort as the subroutine.

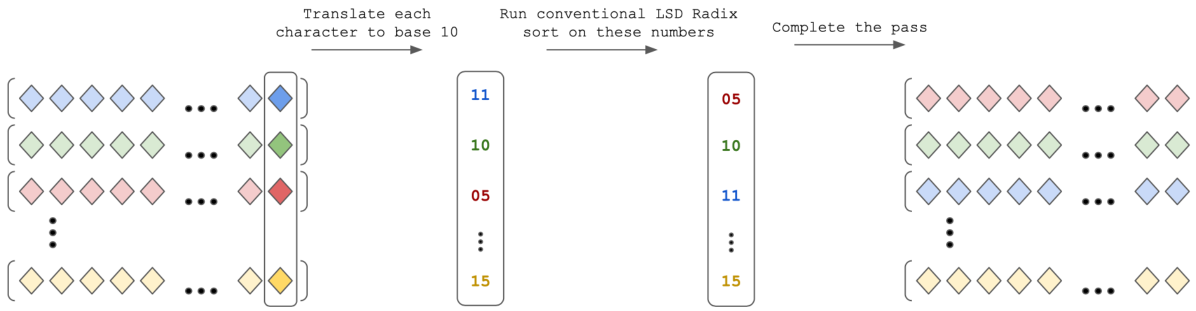
Best Case:

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work

Worst Case:

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work

b) For the following two parts, we will be calling **LSD sort with conventional LSD sort as the subroutine**! Conventional LSD sort uses counting sort as the subroutine. Since conventional LSD sort requires that the objects it's comparing are comprised of **digits of a fixed radix, or base**, we will decompose each character from the alien alphabet into a number of a certain radix. We can visualize the first pass of the sorting algorithm as shown in the image below.



For each part below, find *only* the **worst case** runtime of calling **LSD sort with LSD sort as the subroutine**, if we decompose each character into a number of the given radix.

Here are some helpful hints before you begin.

- If we decompose a character of size M into a number of radix B, the number of digits in the resulting number is $log_B M$.
- $log_B B = 1$.
- $log_B M^a = a \times log_B M$.

i) Radix 10.

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work

ii) Radix N.

◯ $\Theta(1)$ ◯ $\Theta(logN)$ ◯ $\Theta(N)$ ◯ $\Theta(NlogN)$ ◯ $\Theta(N^2)$ ◯ $\Theta(N^2)logN)$
◯ $\Theta(logW)$ ◯ $\Theta(W)$ ◯ $\Theta(WlogW)$ ◯ $\Theta(W^2)$ ◯ $\Theta(W^2logW)$
◯ $\Theta(WlogN)$ ◯ $\Theta(WN)$ ◯ $\Theta(WNlogN)$ ◯ $\Theta(WN^2)$ ◯ $\Theta(WN^2logN)$
◯ $\Theta(NlogW)$ ◯ $\Theta(NWlogW)$ ◯ $\Theta(NW^2)$ ◯ $\Theta(NW^2logW)$
◯ $\Theta(N^2W^2)$ ◯ Never terminates (infinite loop) ◯ None of the above
Would Not Work