Q1:
Convex Hull
In this question our task was to find a polygon which wraps all the points such that all other polygons which wrap these points are bigger than the polygon which we just found out.
Approach:
To get intuition about the problem, at first I read the overleaf notes of the topic Convex hull to get some idea about the problem and how to solve it , then I referred about the same on internet and came to know about two Algorithms, Jarvis March and Graham Scan.
Since Jarvis March method was based on brute force search for angles, it would make the time complexity of our code O(n^2).
So I decided to go with Graham Scan through which I can solve the problem in O(nlogn) time.
Implementation:
1st step: I calculated the angle of my minimum point (left-most) with every other point and stored it in a vector, it took O(n) time.
2nd step: I then sorted the points according to angles, which took O(nlogn) time.
3rd step: I iterated through the vector of points and used the formula of parallelogram's area calculation to find out if the next point is making a clockwise or an anticlockwise turn.
Formula: (bx-ax) * (cy-ay) – (cx-bx) * (by-ay)
If the formula gave positive answer then the angle was clockwise ,anticlockwise when negative and zero when collinear.
Clockwise: pass
Anticlockwise: remove 2 top elements and do it for other element by skipping the older element
Collinear: remove the older element
This took -> O(n) time
So Overall time Complexity of the Algorithm : O(nlogn)
Q2:
Radix Sort
I implemented Radix Sort by applying Counting Sort on each place of given numbers and to sort negative numbers I'm doing that by applying radix sort separately on those numbers and later appending in the list.
Implementation:
1st step: find max element: O(n) time.
2nd step: apply counting sort on each place of digit: O(d+CountSort) time.
3rd step: counting Sort: O(n+k) time.
Overall Time Complexity : O(n(d+k))
n = No of integers
d= Max number of digits
k= Max number of digits possible in a number
Q3:
Median of Medians
In this question our task was to find a kth largest element using median of medians technique in O(n) time.
Approach:
To get intuition about the problem, at first I read the overleaf notes of the topic Median of Medians to get some idea about the problem and how to solve it , then I referred about the same on internet and came to know about the algorithm.
Implementation:
1st step: base case : return sorted array's kth largest element if size of array is less than or equal to 5 : O(5).
2nd step: find min and max element: O(n).
3nd step: find medians by dividing into groups of 5: O(n/5)
4rd step: s-> elements smaller than median of median x
l-> elements greater than median of median x
4th step: recursive calls : O(7n/5)
Overall Time complexity is O(n) as for 5 chunks : T(n) = T(n/5) + T(7n/10) + O(n);
Discrepancy: In test case one, the algorithm appears to calculate the rank in increasing order
Of elements. Conversely, in test case two, it calculates the rank in decreasing order.
Clarification: So I took in increasing order!