Problem 2:

For this problem there are 3 main section that I have to consider

1. Finding the largest alternating subsequence
2. Find all the subsequences for this equivalent length
3. Adding the values of all the values of the subsequence and finding the largest sum

For this problem the immediate though I had was Stalin-Sort on how you just remove values that are outside of your intended sequence,

Similarly, I can remove the values in the given sequence that ruin the alternating behaviour.

In other words, if there are consecutive values of a sequence containing the same sign, then remove all of them until 1 is left. This will ensure the longest possible subsequence that is alternating in sign. This completes requiement 1.

Next I have to find all the subsequences, so immediately i thought this would be inefficient so instead of dealing with the values after collecting all of it in cin, I am going to sanitise my values upon collection.

So as i recieve values when i'm in a consecutive flow of the same sign i'm only going to remember the highest value, and once the sign changes i'm going to sum the largest seen value. So this sort of mitigates the necessaity to fulfil requirement 2 while simultaneously doing requirement 1.

Challenges:

During implementation everything seemed to be going smooth:

Examples

Input	copy	Output	copy
4 5 1 2 3 -1 -2 4 -1 -2 -1 -3 10 -2 8 3 8 -4 -15 5 -2 -3 1 6 1 -1000000000 1 -1000000000 1 -1000000		2 -1 6 -2999999997	

And i was indeed getting all of the correct output values except the last one, that's when i realised that number was too large to be represented as an int.

Even after changing my data type to a long long, when i printed it there was an issue that is because upon printing:

```
for (long long ans : solutions) {
    cout << ans << '\n';
}
```

Unlike python, you have to define the variable's type when looping through an iterable, so I had forgotten to consider this as a long long as well which took me a long time to debug.