Maximum Absolute Difference

https://www.interviewbit.com/problems/maximum-absolute-difference/

https://www.geeksforgeeks.org/maximum-absolute-difference-value-index-sums/

keywords: Arrays, properties of absolute values.

You are given an array of N integers, A_1, A_2 ,…, A_N. Return maximum value of `f(i, j)` for all $1 \le i, j \le N$.
`f(i, j)` is defined as `|A[i] - A[j]| + |i - j|`, where |x| denotes absolute value of x.
For example,

```
A=[1, 3, -1]

f(1, 1) = f(2, 2) = f(3, 3) = 0
f(1, 2) = f(2, 1) = |1 - 3| + |1 - 2| = 3
f(1, 3) = f(3, 1) = |1 - (-1)| + |1 - 3| = 4
f(2, 3) = f(3, 2) = |3 - (-1)| + |2 - 3| = 5

So, we return 5.
```

An **efficient** solution in O(n) time complexity can be worked out using the properties of absolute values.

f(i, j) = |A[i] – A[j]| + |i – j| can be written in 4 ways (Since we are looking at max value, we don't even care if the value becomes negative as long as we are also covering the max value in some way).

```
Case 1: A[i] > A[j] and i > j

|A[i] - A[j]| = A[i] - A[j]

|i -j| = i - j

hence, f(i, j) = (A[i] + i) - (A[j] + j)


Case 2: A[i] < A[j] and i < j

|A[i] - A[j]| = -(A[i]) + A[j]

|i -j| = -(i) + j

hence, f(i, j) = -(A[i] + i) + (A[j] + j)
```

```
Case 3: A[i] > A[j] and i < j
|A[i] - A[j]| = A[i] - A[j]
|i -j| = -(i) + j
hence, f(i, j) = (A[i] - i) - (A[j] - j)

Case 4: A[i] < A[j] and i > j
|A[i] - A[j]| = -(A[i]) + A[j]
|i -j| = i - j
hence, f(i, j) = -(A[i] - i) + (A[j] - j)
```

Note that case 1 and 2 are equivalent and so are case 3 and 4 and hence we can design our algorithm only for two cases as it will cover all the possible cases.

1. Calculate the value of A[i] + i and A[i] – i for every element of the array while traversing through the array.

2. Then for the two equivalent cases, we find the maximum possible value. For that, we have to store minimum and maximum values of expressions A[i] + i and A[i] – i for all i.

3. Hence the required maximum absolute difference is maximum of two values i.e. max((A[i] + i) – (A[j] + j)) and max((A[i] – i) – (A[j] – j)). These values can be found easily in linear time.

a. For max((A[i] + i) – (A[j] + j)) Maintain two variables max1 and min1 which will store maximum and minimum values of A[i] + i respectively. max((A[i] + i) – (A[j] + j)) = max1 – min1

b. For max((A[i] – i) – (A[j] – j)). Maintain two variables max2 and min2 which will store maximum and minimum values of A[i] – i respectively. max((A[i] – i) – (A[j] – j)) = max2 – min2

```
1.  public class Solution {
2.      public int maxArr(ArrayList<Integer> A) {
3.
4.          int max1,max2,min1,min2;
5.          max1=max2=Integer.MIN_VALUE;
6.          min1=min2=Integer.MAX_VALUE;
7.
8.          int i;
9.
10.         for(i=0;i<A.size();i++)
11.         {
12.             max1=Math.max(max1,A.get(i)+i);
13.             min1=Math.min(min1,A.get(i)+i);
14.             max2=Math.max(max2,A.get(i)-i);
```

```
15.             min2=Math.min(min2,A.get(i)-i);
16.         }
17.
18.         return Math.max(max1-min1,max2-min2);
19.     }
20. }
```