1. Find index of no of 1 and no of 0Find max subarray with max sum - Kadan’s algorithm

int max\_so\_far = INT\_MIN, max\_ending\_here = 0;

for (int i = 0; i < size; i++)

{

max\_ending\_here = max\_ending\_here + a[i];

if (max\_so\_far < max\_ending\_here)

max\_so\_far = max\_ending\_here;

if (max\_ending\_here < 0)

max\_ending\_here = 0;

}

return max\_so\_far;

1. Binary XOR (CODEcheck long challange)

<https://www.codechef.com/DEC19B/problems/BINXOR>

Find the all possible hamming distance between two strings

Min\_hd = ones(A) + ones(B)

Max\_hd = ones(A) - ones(B)

For x = Min\_hd to Max\_hd ; x += 2

Ans += n! / (x! \* (n-x)!)

Eg : A = 001111 A = 11110

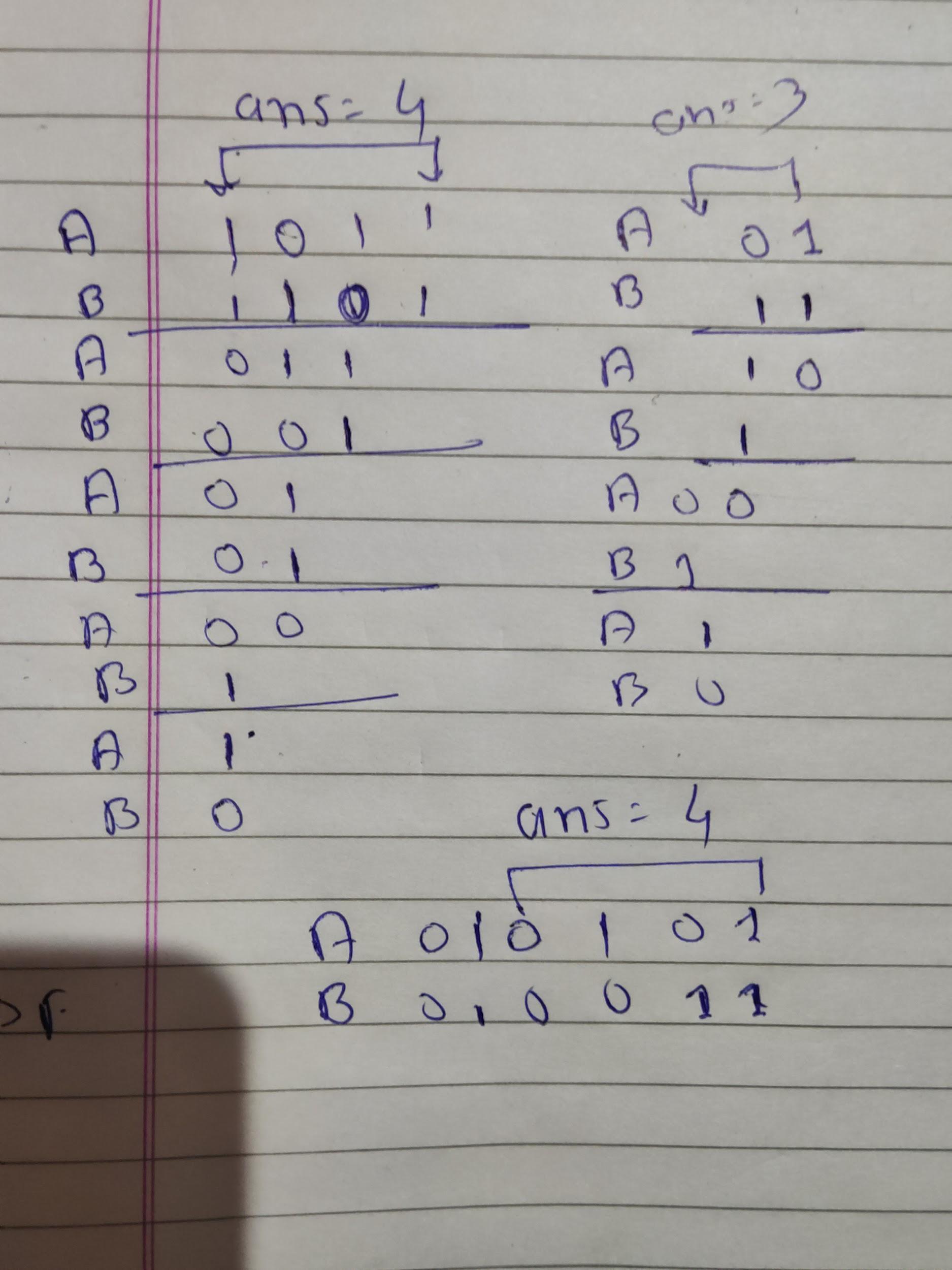
B = 000011 B = 11100

Possible hd = 6, 4, 2, 0 hd = 3, 1

1. Binary Addition ( codechef long challange)

<https://www.codechef.com/DEC19B/problems/BINADD/>

Find the max diff between index of two ones’ and nearest two ones’ or two zeros’



1. Trie
   1. <https://leetcode.com/problems/implement-trie-prefix-tree/>
   2. <https://leetcode.com/problems/add-and-search-word-data-structure-design/> (trie with regular expression .)
2. Union find
   1. <https://leetcode.com/problems/most-stones-removed-with-same-row-or-column/submissions/>
   2. code

for(int i=0;i<1201;i++)

parent[i] = i;

int parent[1201];

int find(int u)

{

if(parent[u] == u)

return u;

int x = find(parent[u]);

parent[u] = x;

return x;

}

void dounion(int u, int v)

{

int pu = find(u);

int pv = find(v);

if(pu == pv)

return;

parent[pu] = pv;

}

1. <https://leetcode.com/problems/trapping-rain-water/>
2. Intervals
   1. <https://leetcode.com/problems/merge-intervals/submissions/>
3. Division without using / \* mod

<https://leetcode.com/problems/divide-two-integers/>