Take forward:https://takeuforward.org/data-structure/implement-k-stacks-in-a-single-array

See here we have 1 array and must implement n stacks. there can be multiple approaches.

Consider 1st approach we have 3 stack there we an allocate to each n/k space that is 2 above .

But the issue will be considered stack 1 has 12345 here 12 can go rest cannot go there. Here rest of space will go to waste of stack 2 and 3. also we have extra space in array but still we cannot fill 345 of stack 1 in array as allocated less.

2nd logic:

Here we use top to store index of top element of stack 0, stack 1 and stack 2.

This will store the index of next empty index like at 0 if we want to store here, we 1 is next empty we will store it.

Doing iterate in following example will explain everything.

Initially this will be -1 as empty.

Initially this will be filled like this as array empty hence next index is free.

Here we will also take these 2 variables as they will tell us which is free. Free will always store whichever is free. While I is to iterate and put the element in array.

See here we are filling arr 0 index with number hence change next. Then next store whatever was earlier on stack 1 currently top 0 index is -1 hence -1.

Here arr 0 index will be filled with element of stack 0 and hence top 0 index will change to 0 as it will store top of stack 0 which is stored in index 0 of arr.

Now store 2 in arr.

See free store next free index and I store current index in which we store data.

Now for storing 2 here we know current free index will show shw on free variable which is 1 this i=1. And now store next free index on free variable from next at 1 index which is 2.

Now at next index 1 will store 0 as this is the index which store earlier element of stack and this every time will be taken from top of index 0 like from stack 0 or top index 1 from stack 1. here top 0 index for stack 0.

See here next at index 1 we see 0 index means 0 store its earlier element.

see now top of 0 index store 1. where current top of stack 0 is stored.

Now insert stack 1 with value 3.

We know that currently at free stored 2 and i=1. now i=free and free is next free element at next index 2 which is 3, free=3.

At next index 2 will store earlier data stored for that stack here none hence -1 and top here at index 1 will be 2.

Now we know free=3 and i=2.hence i=free and free will store whatever it at next index 3 free=4.

Now here we want to store 4 at index 4 of arr.at next index 3 we will store top of index 0 which is 1 hence next at index 3 is 1 and change top at index 0 to 3 .

See here using next index we can track back stack 0.

Same logic.

Here at push first is element and second is the stack it represents.

Now pop:

Remove 4 from stack 0.

Here we will go to top of 0 as remove from stack 0.go to that index which is 3.now top of index 0 will be next[3] which is 1 and next[3] will be free =5 which will be stored and now free =3 as this will be closest stack saved.

Same logic to pop 2 from stack 0.