b'

Final Exam | Google Kickstart 2021 Round D

- Difficulty Level :\nHard
- Last Updated :\n21 Sep, 2021

It\xe2\x80\x99s time for the final exam in algorithms and data structures!

Edsger prepared **N** sets of problems. Each set consists of problems in an increasing difficulty sequence; the i-th set can be described by two integers Ai and Bi (Ai\xe2\x89\xa4Bi), which denotes that this set contains problems with difficulties Ai, Ai+1\xe2\x80\xa6, Bi. Among all problems from all sets, it is guaranteed that no two problems have the same difficulty.

This semester Edsger has to test **M** students. He wants to test each student with exactly one problem from one of his sets. No two students can get the exact same problem, so when Edsger tests a student with some problem, he cannot use this problem anymore. Through countless lectures, exercises, and projects, Edsger has gauged student number j to have skill level Sj and wants to give that student a problem with difficulty Sj. Unfortunately, this is not always possible, as Edsger may have not prepared a problem of this difficulty, or he may have already asked this problem to some other student earlier. Therefore, Edsger will choose for the j-th student a problem of difficulty Pj, in a way that |Pj\xe2\x88\x92Sj| is minimal and a question of difficulty Pj was not already given to any of the students before the j-th student. In case of ties, Edsger will always choose the easier problem. Note that the problem chosen for the j-th student may affect problems chosen for all the students tested later, so you have to process students in the same order as they appear in the input.

As keeping track of all the problems can be fairly complicated, can you help Edsger and determine which problems he should give to all of his students?

\xc2\xa0 \xc2\xa0OR

Given an array **problemRange** of **N\xc2\xa0**pairs having starting and ending values as a range of difficulty levels, and an array **arr** of size **M** indicating the difficulty level every student can attempt. The task is to assign a unique integer **X** from **problemRange** to every integer in array **arr\xc2\xa0**such that **| arr[i] \xe2\x80\x93 X |** is minimized. In case of a tie between two values closest to **arr[i]**, a lesser difficulty value must be chosen. **X** values must be assigned to students in their order since the same value of **X** cannot be assigned to more than one student. Print the **X** value assigned to every student.\xc2\xa0

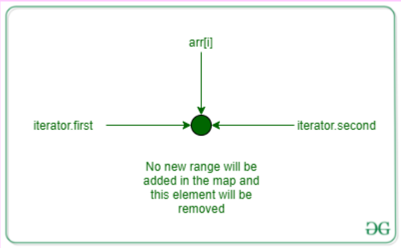
Example:

> **Input:** N = 5, M = 4, arr = [14, 24, 24, 4], problemRange = [[1, 2], [6, 7], [9, 12], [24, 24], [41, 50]]\xc2\xa0
> **Output:** 12 24 11 2
> **Explanation:** values which can be assigned to the students are {1, 2}, {6, 7}, {9, 10, 11, 12}, {24}, {41, 42, 43, 44, 45, 46, 47, 48, 49, 50} 12 is assigned to first student who can attempt questions of difficulty level 14 as it is the closest to 14. 24 is closest to 24. Next student can also attempt question of 24 difficulty but 24 from the range is already chosen and the next closest is 11. 2 and 6 is closest to last student of difficulty 4, since 2 and 6 both are similarly close to 4, easier questions of difficulty level 2 is assigned.\xc2\xa0
>
> **Input:** N = 1, M = 1, arr = [24], problemRange = [[42, 42]]
> **Output:** 42

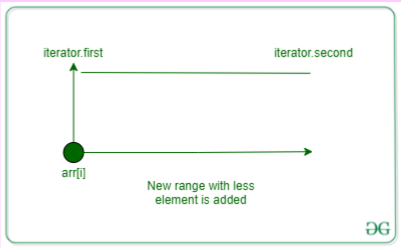
\xc2\xa0

Approach: Given problem can be solved by following the steps below:

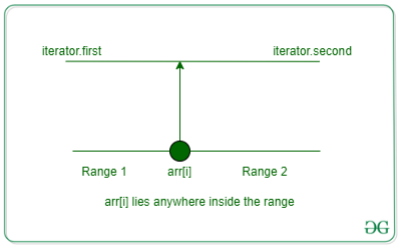
- Use a map to store the start of ranges as keys and end of ranges as values
- Iterate the array and for every element in it find its **lower_bound** in the map
- Two cases are possible: **Lower_bound** will return the iterator pointing to the key which will be equal to **arr[i]** or the key which is just greater than **arr[i]**
  - let\xe2\x80\x99s say the iterator provided by **lower_bound** be **it** and **pre** be an iterator just before **it** (**pre** will be equal to **it** when **it**=**mp.begin()**)
  - Either **pre.first<=arr[i]<=pre.second** or **it.first<=arr[i]<=it.second** will be true. **arr[i]** will lie in either the forward section or in the backward section of this range
  - every time value is assigned to the **arr[i]** previous range is deleted from the map and a new range is added as shown in the image
  - Either two new ranges are added or one new range is added as cases shown in the images below:



Only one element is present in the range so it is removed and no new range is added\xc2\xa0



arr[i] is equal to either one of the range



The previous range is removed and two new ranges are added in the map

Below is the implementation of the above approach:

C++

```
// C++ implementation for the above approach
\xc2\xa0
#include <bits/stdc++.h>
\xc2\xa0
using namespace std;
\xc2\xa0
void solve(long long int N, long long int M,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0vector<pair<long long int,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0problemRange,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0vector<long long int> arr)
{
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0// Store the problem range in a map
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0map<long long int, long long int> mp;
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0for (long long int i = 0; i < N; i++) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int a, b;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0a = problemRange[i].first;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0b = problemRange[i].second;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp[a] = b;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0vector<long long int> ans(M);
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0for (long long int i = 0; i < M; i++) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0auto it = mp.lower_bound(arr[i]);
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0auto pre = it;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (it != mp.begin())
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0pre--;
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0// If answer lies in a valid range
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (pre->first <= arr[i]
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0&& arr[i] <= pre->second) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0ans[i] = arr[i];
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int st = pre->first,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp.erase(pre);
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int left = arr[i] - 1,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (st <= left) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp[st] = left;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (end >= right) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp[right] = end;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0// If answer is not in a valid range
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0else {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int op1 = pre->second,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (abs(arr[i] - op1) <= abs(arr[i] - op2)) {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0ans[i] = op1;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int st = pre->first,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
= op1 - 1;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp.erase(pre);
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (st <= end)
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp[st] = end;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0else {
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0ans[i] = op2;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int st = it->first + 1,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
= it->second;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp.erase(it);
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0if (st <= end)
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0mp[st] = end;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0}
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0for (auto it : ans)
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0cout << it << " ";
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0cout << endl;
}
\xc2\xa0
// Driver code
int main()
{
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int N, M;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0N = 5;
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0M = 4;
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0// Student difficulty level
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0vector<long long int> arr = { 14, 24, 24, 4 };
\xc2\xa0
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0vector<pair<long long int,
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0long long int> >
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0problemRange = { { 1, 2 },
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0\xc
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0solve(N, M, problemRange, arr);
\xc2\xa0\xc2\xa0\xc2\xa0\xc2\xa0return 0;
}
```

Output

```
12 24 11 2
```

Time complexity: MLog(N)
Auxiliary Space: O(N)

My Personal Notes\n*arrow_drop_up*

Add your personal notes her

Save

'