Saturday, January 1, 2022 12:23 PM

Allocate minimum number of pages
Hard Accuracy: 48.87% Submissions: 35670 Points: 8

You are given **N** number of books. Every **ith** book has **Ai** number of pages and are arranged in **sorted order**.

You have to allocate contagious books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is minimum of those in all the other permutations and print this minimum value.

Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation: Allocation can be done in
following ways:{12} and {34, 67, 90}
Maximum Pages = 191{12, 34} and {67, 90}
Maximum Pages = 157{12, 34, 67} and {90}
Maximum Pages =113. Therefore, the minimum
of these cases is 113, which is selected
as the output.

Example 2:

Input:
N = 3
A[] = {15,17,20}
M = 2

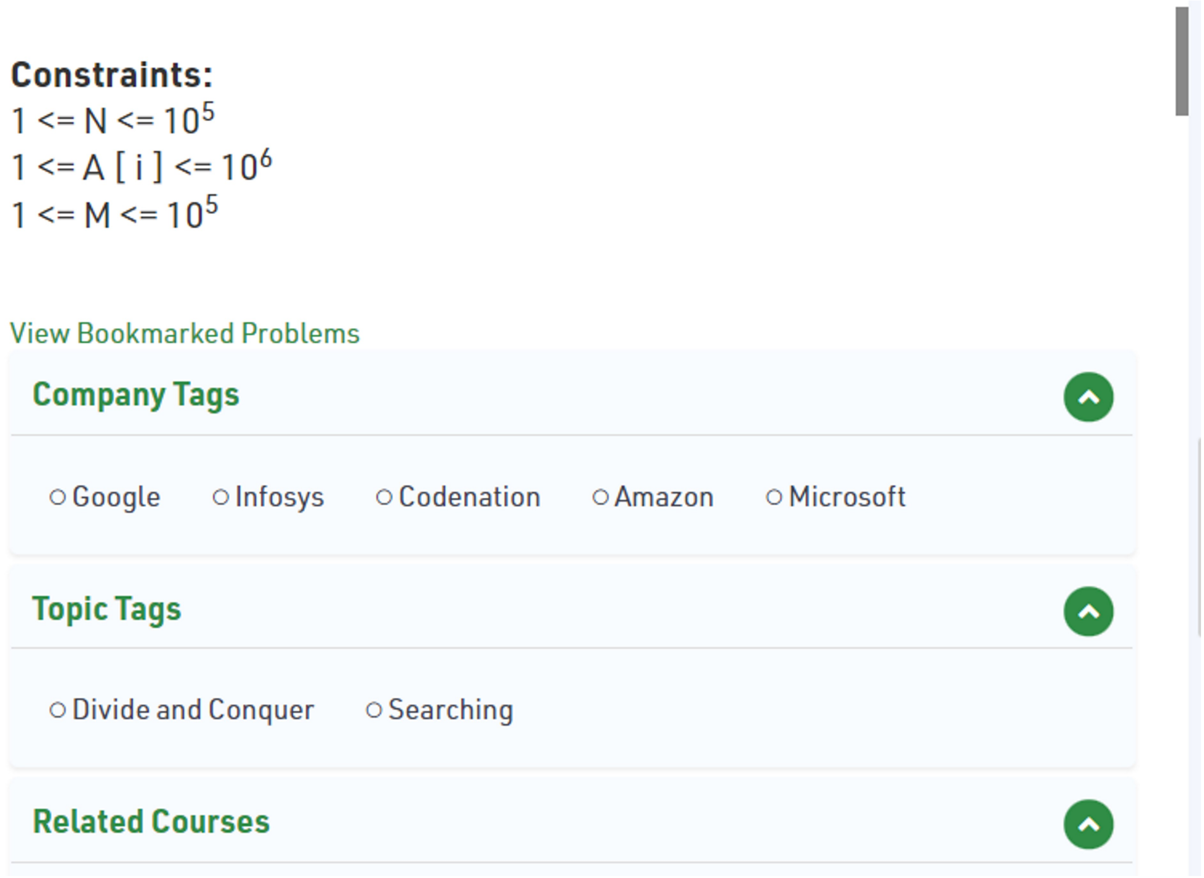
Output:32
Explanation: Allocation is done as
{15,17} and {20}
Your Task:

You don't need to read input or print anything. Your task is to complete the function findPages() which takes 2 Integers **N**, and m and an array **A[]** of length **N** as input and returns the expected answer.

Expected Time Complexity: O(NlogN)

Expected Auxilliary Space: O(1)

From <https://practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1>



Constraints:
1 <= N <= 10^5
1 <= A [i] <= 10^6
1 <= M <= 10^5

View Bookmarked Problems

Company Tags

○ Google ○ Infosys ○ Codenation ○ Amazon ○ Microsoft

Topic Tags

○ Divide and Conquer ○ Searching

Related Courses

○ DSA-Self Paced with Doubt Assistance ○ Data Structures and Algorithms

○ Placement 100 ○ Complete Interview Preparation

○ Complete Interview Preparation With Doubt Assistance

○ Microsoft SDE Test Series ○ Google Test Series

○ Must Do Coding Questions ○ Placement Guide 2021-2022

○ Competitive Programming - Live ○ DSA Live for Working Professionals

○ Data Structures with C++ Live ○ Advance DSA Training program

○ Advance DSA Training Course

Related Interview Experiences

○ Amazon interview experience on campus for sde 1 5

Hint 1 {

Use Binary search

}

Hint 2 {

We need to think whether we can find how many number of students we need if we fix that one student can read at most V number of pages. So, our problem statement reduces to : Given fixed number of pages (V), how many number of students we need?

From <https://practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1>

}

Hint 3 {

Check below pseudocode for better understanding.

```
intially Sum := 0
cnt_of_student = 0
iterate over all books:
    If Sum + number_of_pages_in_current_book > V :
            increment cnt_of_student
            update Sum
    Else:
            update Sum
  EndLoop;

  fix range LOW, HIGH
  Loop until LOW < HIGH:
        find MID_point
        Is number of students required to keep max number of pages below MID < M ?
        IF Yes:
           update HIGH
        Else
           update LOW
   EndLoop;
```

Close

From <https://practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1>

}

The idea is to use [Binary Search](). We fix a value for the number of pages as mid of current minimum and maximum. We initialize minimum and maximum as 0 and sum-of-all-pages respectively. If a current mid can be a solution, then we search on the lower half, else we search in higher half.

Now the question arises, how to check if a mid value is feasible or not? Basically, we need to check if we can assign pages to all students in a way that the maximum number doesn't exceed current value. To do this, we sequentially assign pages to every student while the current number of assigned pages doesn't exceed the value. In this process, if the number of students becomes more than m, then the solution is not feasible. Else feasible.
Below is an implementation of above idea.

```
 static boolean isPossible(int arr[], int n, int m, int curr_min)
    {
        int studentsRequired = 1;
        int curr_sum = 0;

        // iterate over all books
        for (int i = 0; i < n; i++)
        {
            // check if current number of pages are greater
            // than curr_min that means we will get the result
            // after mid no. of pages
```

```
      if (arr[i] > curr_min)
         return false;

      // count how many students are required
      // to distribute curr_min pages
      if (curr_sum + arr[i] > curr_min)
      {
         // increment student count
         studentsRequired++;

         // update curr_sum
         curr_sum = arr[i];

         // if students required becomes greater
         // than given no. of students,return false
         if (studentsRequired > m)
            return false;
      }

      // else update curr_sum
      else
         curr_sum += arr[i];
   }
   return true;
}

// method to find minimum pages
static int findPages(int arr[], int n, int m)
{
   long sum = 0;

   // return -1 if no. of books is less than
   // no. of students
   if (n < m)
      return -1;

   // Count total number of pages
   for (int i = 0; i < n; i++)
      sum += arr[i];

   // initialize start as 0 pages and end as
   // total pages
   int start = 0, end = (int) sum;
   int result = Integer.MAX_VALUE;

   // traverse until start <= end
   while (start <= end)
   {
      // check if it is possible to distribute
      // books by using mid is current minimum
      int mid = (start + end) / 2;
      if (isPossible(arr, n, m, mid))
      {
```

```
                // update result to current distribution
                // as it's the best we have found till now.
                result = mid;

                // as we are finding minimum and books
                // are sorted so reduce end = mid -1
                // that means
                end = mid - 1;
            }

            else
                // if not possible means pages should be
                // increased so update start = mid + 1
                start = mid + 1;
        }

        // at-last return minimum no. of  pages
        return result;
    }
```

From <https://practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1>