// C++ code to implement the approach

#include <bits/stdc++.h>

using namespace std;

// Function to check if

// all the task can be

// completed by 'per\_day'

// number of task per day

bool valid(int per\_day,

vector<int> task, int d)

{

// Variable to store days required

// to done all tasks

int cur\_day = 0;

for (int index = 0; index < task.size();

index++) {

int day\_req

= ceil((double)(task[index])

/ (double)(per\_day));

cur\_day += day\_req;

// If more days required

// than 'd' days so invalid

if (cur\_day > d) {

return false;

}

}

// Valid if days are less

// than or equal to 'd'

return cur\_day <= d;

}

// Function to find minimum

// task done each day

int minimumTask(vector<int> task, int d)

{

int left = 1;

int right = INT\_MAX;

for (int index = 0;

index < task.size();

index++) {

right = max(right, task[index]);

}

// Variable to store answer

int per\_day\_task = 0;

while (left <= right) {

int mid = left

+ (right - left) / 2;

// If 'mid' number of task per day

// is valid so store as answer and

// more to first half

if (valid(mid, task, d)) {

per\_day\_task = mid;

right = mid - 1;

}

else {

left = mid + 1;

}

}

// Print answer

return per\_day\_task;

}

// Driver Code

int main()

{

// Input taken

vector<int> task{ 3, 4, 7, 15 };

int D = 10;

cout << minimumTask(task, D) << endl;

return 0;

}