***Saving Efficiency***

C371\_Coding\_November2022

**Topic**: Searching Algorithms

**Difficulty Level:** Medium

**Question / Problem Statement**:

There are **N** Schools in a straight line and positions of these schools are given in array **A** and **K** students who want to spread their knowledge to the entire road of school.

All the students have knowledge value, which denotes the range of knowledge which they can spread in both directions. Since, they do not want to waste their efficiency on trivial things of the world, they want to keep their range minimum.

Write a program to find the minimum knowledge value such that all the school can receive it.

**Note**

The **N** schools are in a straight line, we mean that all the schools lie on something like an X-axis in a graph.

**Function Description**

In the provided code snippet, implement the provided **savingEfficiency(...)** method using the variables to find the minimum knowledge value such that all the school can receive it. You can write your code in the space below the phrase **“WRITE YOUR LOGIC HERE”**.   
  
There will be multiple test cases running so the Input and Output should match exactly as provided.  
The base Output variable **result** is set to a default value of **-404** which can be modified. Additionally, you can add or remove these output variables.

**Input Format**

The first line contains two integers, **N** and **K** - denoting the number of schools and students.

The next line contains the position array - **A**.

**Sample Input**

3 2 – **N, K.**

1 5 20 – **A.**

**Constraints**

1 <= **N** <= 10^5

1 <= **K** < **N**

1 <= **A***i* <= 10^7

**Output Format**

Output should find the minimum knowledge value such that all the school can receive it.

**Sample Output**

2

**Explanation**

The optimal answer is 2. A student positioned at 3, can serve school at 1 and 5. The other student can be placed at 18, to serve the school at 20.

**Solution Steps**

1. First sort the input array in ascending order.

2. The main idea behind the question is to apply binary search to choose the minimum possible answer and then check whether it is possible or not. For this we will keep on decreasing the considered value until the check fails. We all know whatever the minimum answer is, cannot be other than in between 0 to 10^7. So this is the range in which we have to apply binary search. Every time we will calculate the mid value of the considered range [l, r] (taking l=0; r=10^7; initially) and will check whether this mid value (i.e .possible minimum knowledge value or answer) is possible or not and will also check at mid-1. If the check at mid returns that, this value is possible(returns 1) and at mid-1, this mid-1 is not possible. Then our answer will be mid. If both at mid and mid-1 check is valid(i.e. returns 1) then r=mid-1; in loop and if at mid, this mid value is not valid(i.e. check fails or say returns 0) then l=mid+1; This part will take logarithmic time. Here is the given solution. h is initialised as 10^8, you could make it 10^7. And there is no use of statements containing variable p.

3. Now how to check whether a considered mid value is valid or not : In the given input array, take the first school and keep a student at a position or at a distance of "mid" to the right of first school and decrement the number of students by 1( Here this is denoted by, prev=tmp[0] + num ;(i.e. "mid") k--; (i.e. student--) ). Now if the effect of considered student is up to a distance greater than the next school, then keep on going to right (i.e. continue; i++;) if in between travelling from first to last school, student decreases to zero, then check fails.otherwise keep on going right updating the position of recently put student and decrementing the number of student. If we arrive at the last school, then the check succeeds.

**Running Solution in C++** :

#include <bits/stdc++.h>

#define int long long int

using namespace std;

int check(int n,int a[],int num,int k){

int i,j,prev=a[0]+num;

k--;

for(i=1;i<n;i++)

{

if(prev+num>=a[i])

continue;

if(k==0)

return 0;

prev=a[i]+num;

k--;

}

return 1;

}

int32\_t main() {

int n,k;

cin>>n>>k;

int a[n];

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

cin>>a[i];

sort(a,a+n);

int l=0, h=100000000, mid,p=-1;

while(h>=l)

{

mid=(l+h)/2;

int vv=check(n,a,mid,k);

if(check(n,a,mid,k) && check(n,a,mid-1,k)==0)

break;

if(vv==0){

p=mid;

l=mid+1;

}

else

h=mid-1;

}

cout<<mid<<"\n";

return 0;

}

Input:

5 8

2 5 7 8 9

Output:

0

**Test Cases [ Qty: 12 ]**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case No** | **Input** | **Output** | **Score** |
| 1 | 3 2  1 5 20 | 2 | 0 |
| 2 | 5 8  2 5 7 8 9 | 0 | 0 |
| 3 | 42 18  1001 3357 1125 8369 7226 9388 852 6291 9573 8165 7007 5545 1437 9611 1523 7799 5529 6434 4291 7134 227 6873 2252 1605 1269 9135 5784 744 7563 9061 3981 529 5681 1073 2031 3811 9501 1607 6365 6868 2137 1041 | 118 | 1 |
| 4 | 967 615  6749 6417 199 5907 7677 8590 8796 3463 721 963 1779 8646 4529 7831 8222 1665 3941 4397 6286 2971 1613 2501 6285 3381 1438 5360 160 9400 5006 5423 9949 3213 3181 4385 1901 1111 191 4089 5685 1396 8701 3277 1481 517 6253 1313 2255 5948 6085 7201 5310 1105 985 748 9271 2883 7345 4686 9401 3565 7795 3793 6997 279 9576 2900 2 943 9601 5045 1681 8239 2023 414 9585 9097 2803 831 7979 5721 4717 6069 4961 7889 4733 2876 1399 3105 8239 5623 9680 9133 7989 3401 6001 9608 2369 2390 2866 6171 1871 7489 4070 3425 27 8270 8105 2657 876 9267 8793 8701 7655 4798 8885 1424 6914 4835 6636 1553 6031 3617 6971 1915 3037 3719 3116 2806 6897 1241 1629 2119 2373 3265 9513 5770 751 6619 997 4189 4073 4541 9506 5219 429 1025 5236 7579 2251 9479 4737 7917 5149 2311 9640 4687 4711 3685 121 4961 7552 3399 3376 8733 1357 8042 4297 5466 9476 5865 7493 5601 3321 263 5901 5756 5361 7633 6751 7161 6681 376 9745 717 6887 55 8885 6353 7089 3601 1561 571 3551 3753 4595 877 1853 3156 7499 876 2713 3961 7725 5309 6358 2376 641 5433 9781 7482 1818 2561 5961 1445 3941 6799 1701 3725 3556 2401 7463 6281 5669 4517 9232 2851 8881 313 8701 145 9261 3201 1156 8097 4483 1783 1328 7710 1713 7105 7033 4986 2568 5386 4989 7081 2987 6657 1689 5634 2317 3182 9486 817 5151 1667 2295 9676 6676 9271 4685 516 2301 5580 5663 6159 3109 4081 5177 4301 6734 6026 3605 4848 5473 2137 5209 2521 3001 8981 5845 7841 440 9521 6651 7734 8551 3843 7741 1353 4266 5161 785 7988 5201 4125 8245 7748 1648 9961 6463 2278 6846 6297 225 4529 2761 2269 8951 5061 6090 9813 1524 3279 711 9361 2889 6148 65 3777 3751 6526 867 9207 4433 3945 1476 41 4189 8479 8216 9817 7101 3461 2661 9124 2305 2225 8911 6804 9922 401 3476 3205 4537 692 9368 4370 1097 2481 6519 9153 741 9008 6601 6117 1689 5911 2031 2538 5553 1358 6863 7521 4869 6276 5356 5305 6761 5689 7476 5833 257 2157 218 8821 2313 1876 6341 1777 9117 8913 8369 3111 7966 1431 876 8483 8501 1709 5230 8033 6097 8783 4103 5569 6353 1522 5053 6001 8845 7751 4305 5429 3488 6023 3573 3147 6281 8047 2640 4403 1953 1914 4046 7569 5406 5123 3237 7171 7875 2189 5788 2536 6254 2601 4345 2156 7326 4871 9057 2229 351 6617 9701 5492 5086 4781 1525 965 7685 8905 8391 8450 5153 1601 8836 4155 1712 7136 8601 6205 2701 9138 4881 885 7177 3657 3506 8017 8617 7943 3485 9241 6903 8145 3236 6447 5685 6691 3614 9 9437 3450 2773 1681 2227 2705 971 3393 3631 7585 6593 4935 1563 2289 8741 6852 2675 6989 2183 9257 5447 3477 7233 226 9327 7815 7341 3201 5123 1697 6281 1126 6055 5393 3947 3629 1681 9326 683 8801 5321 3111 4601 1195 8109 7445 7973 2995 3626 2476 4721 1381 4921 9835 365 8482 4591 551 1751 3633 4687 2949 9097 8860 3176 3349 1952 81 5345 5886 9697 4489 5577 4513 5361 5507 2174 950 1736 926 3201 3536 2625 8083 2701 236 6830 8497 6099 2169 1221 4947 2321 5163 3181 2137 6011 4041 6013 8735 7254 2976 2301 2623 773 9 6337 9855 1321 1485 7407 5145 7603 1671 4201 4965 6685 1545 7388 6827 457 4685 4987 7969 3421 3793 8251 5465 2014 8363 6237 6389 1469 9957 47 5565 7982 7361 4981 8407 9660 3851 5209 3851 3169 6790 3156 3410 5249 761 4243 1230 541 6685 3571 7452 506 3264 5021 6557 3963 1761 9959 181 6939 8621 6417 2995 5155 7956 4379 6751 3173 6739 9629 4347 8716 7376 4197 3881 9841 4577 2721 9011 6191 5605 4097 2889 1737 7217 505 8281 4209 5708 4616 2901 8177 1074 4043 5201 1461 3544 6841 6465 6581 9976 8855 7957 773 1041 4481 3266 9701 9073 2291 2863 7968 4090 5141 1977 7781 6903 2403 8392 6529 3041 7137 3932 9591 2505 8143 5460 1841 2330 7313 3933 1603 3353 7901 9497 6481 337 6009 3081 2021 563 2228 591 996 7137 685 2583 6751 2104 2596 2329 9948 7961 9545 1797 6507 9241 3844 5657 1887 225 7310 1165 6335 5729 5179 8166 9294 3281 8037 1063 6711 8103 7461 4226 2894 9085 4921 7618 5651 2776 7751 2046 2349 6025 833 4405 211 4198 1696 6699 2898 369 7485 9759 3677 6741 4041 2582 4045 3092 6311 8883 5393 6177 3556 6429 4931 1025 5283 723 5471 2985 497 3961 9829 6451 1240 366 5269 5446 7603 7151 2948 8891 9996 7989 9873 8357 1831 8156 5905 7481 9116 7795 1069 7289 8387 3013 7881 2853 257 7683 2256 185 2209 1017 8021 216 6715 8201 9324 276 8441 2378 4829 9303 5721 3895 8193 7725 1246 8845 6863 2897 5001 5055 2745 596 9108 4313 1108 982 6483 7256 4313 8981 9026 9885 2433 2009 8441 7441 9044 6969 2065 6721 424 5478 9128 5921 11 6201 3681 4876 3369 6205 4865 8201 9751 371 2881 7995 641 5841 3595 6041 2403 1361 5121 3801 8031 7909 3809 7741 1026 9633 8711 1907 6363 6977 625 6939 8501 1266 5256 9921 9822 2056 6142 5957 2958 455 9000 5881 9193 4428 9329 1431 236 279 1977 2421 9211 9481 8096 1871 6788 7991 3675 371 2453 905 1366 6471 4331 4106 2570 4647 1648 7911 2147 1273 6437 3393 7124 7961 7428 9325 6874 8264 5366 5346 1951 3303 1613 5826 8035 7079 7633 6155 9811 9761 3207 4293 3551 5245 7891 4463 3981 | 3 | 1 |
| 5 | 42 18  1 357 125 369 226 388 852 291 573 165 7 545 437 611 523 799 529 434 291 134 227 873 252 605 269 135 784 744 563 61 981 529 681 73 31 811 501 607 365 868 137 41 | 14 | 1 |
| 6 | 67 51  749 417 199 907 677 590 796 463 721 963 779 646 529 831 222 665 941 397 286 971 613 501 285 381 438 360 160 400 6 423 949 213 181 385 901 111 191 89 685 396 701 277 481 517 253 313 255 948 85 201 310 105 985 748 271 883 345 686 401 565 795 793 997 279 576 900 2 | 2 | 1 |
| 7 | 2 1  1 57 | 28 | 1 |
| 8 | 9 7  69 26 88 52 91 73 65 7 45 | 2 | 1 |
| 9 | 1 4  7 | 0 | 1 |
| 10 | 6 4  7 434 64 23 54 54 | 5 | 1 |
| 11 | 11 5  434 64 23 54 54234 24435 24444 55666 65656 6556 56665 | 1216 | 1 |
| 12 | 5 4  434 64 54 24435 65656 | 5 | 1 |

Plagiarism found – No

Clarity of the problem statement - Yes

Clarity of the example in the problem statement - Yes

Clarity of sample test cases - Yes

Clarity of test cases (Dual output) – Yes

Clarity of explanations - Yes

Provided Solution running – Yes

EEOC complaint (using abusive words/Indian Names/) - No

Similar Question in System - No

Difficulty Level – Medium

Question w.r.t Searching algorithms concepts- Yes

Final Comment: **Accepted**