



Email:

leo.zhao@vanderbilt.edu

Test Name:

[TikTok] AMS Intern Assessment 2025 Start - 7 - 11 Dec (Generic) - Practice Session

Taken On:

9 Dec 2024 04:57:45 IST

Time Taken:

6 min 28 sec/ 42000 min

Invited by:

Prepkrit

Skills Score:

Problem Solving (Intermediate)

225/225

Tags Score:

Adjacency Matrix

5/5

Algorithms

75/75

Arrays

75/75

Binary Search

150/150

Data Structures

80/80

Dijkstra's Algorithm

5/5

Easy

5/5

Graphs

5/5

Hard

5/5

Hashing

75/75

Interviewer Guidelines

75/75

Javascript

5/5

Medium

225/225

Problem Solving

75/75

Programming Fundamentals

5/5

Strings

75/75

Theme: Finance

75/75

100%

235/235

scored in [TikTok] AMS Intern Assessment 2025 Start - 7 - 11 Dec (Generic) - Practice Session in 6 min 28 sec on 9 Dec 2024 04:57:45 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Product Recommendations > Multiple Choice	3 sec	5/ 5	✔
Q2	Deque operations > Multiple Choice	43 sec	5/ 5	✔
Q3	User-Friendly Password System > Coding	3 min 19 sec	75/ 75	✔
Q4	Profit Targets > Coding	40 sec	75/ 75	✔
Q5	Maximizing Element With Constraints > Coding	1 min 33 sec	75/ 75	✔

QUESTION 1



Correct Answer

Score 5

Product Recommendations > Multiple Choice

Hard

Programming Fundamentals

Javascript

Dijkstra's Algorithm

Graphs

Adjacency Matrix

QUESTION DESCRIPTION

An e-commerce company has a platform that uses an algorithm to suggest products to users. It uses a graph-based approach, where each node represents a product, and the edges between them represent similarities. To avoid recommending too many similar items, they implement a modified version of Dijkstra's algorithm that takes into account the similarity score.

Here is the JavaScript code snippet:

```
function dijkstra(graph, startProduct) {
  let n = graph.length;
  let minDistances = new Array(n).fill(Infinity);
  let visited = new Array(n).fill(false);

  minDistances[startProduct] = 0;

  for (let i = 0; i < n; i++) {
    let minIndex = -1;
    for (let j = 0; j < n; j++) {
      if (!visited[j] && (minIndex === -1 || minDistances[j] <
minDistances[minIndex])) {
        minIndex = j;
      }
    }
    if (minDistances[minIndex] === Infinity) {
      break;
    }
    visited[minIndex] = true;

    for (let j = 0; j < n; j++) {
      if (graph[minIndex][j] !== 0) {
        let potentialDist = minDistances[minIndex] +
graph[minIndex][j];
        if (potentialDist < minDistances[j]) {
          minDistances[j] = potentialDist;
        }
      }
    }
  }
  return minDistances;
}
```

Given an adjacency matrix that represents the similarity scores between different products:

```
let graph = [
  [0, 2, 0, 1, 0],
  [2, 0, 3, 0, 0],
  [0, 3, 0, 4, 0],
  [1, 0, 4, 0, 5],
  [0, 0, 0, 5, 0]
];
```

What is the output when the recommendation system calls `dijkstra(graph, 0)`; to find the least similar products to the current product (*Product 0*)?

INTERVIEWER GUIDELINES

The adjacency matrix 'graph' represents the similarity scores between different products. Each row and column index corresponds to a product, and the value at `graph[i][j]` represents the similarity score between

product 'i' and product 'j'. A higher score indicates higher similarity. The Dijkstra's algorithm in this case works by finding the path of least similarity (i.e., the path with the lowest total score). So, starting from product 0, the algorithm calculates the least total similarity score to reach every other product. Hence, the output [0, 2, 5, 1, 6] represents the least total similarity scores from product 0 to every other product.

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ☒ [0, 2, 5, 1, 6]
- ☐ [0, 1, 4, 2, 7]
- ☐ [0, 2, 3, 1, 5]
- ☐ [0, 2, 3, 1, 6]

No Comments

QUESTION 2



Correct Answer

Score 5

Deque operations > Multiple Choice

Easy

Data Structures

QUESTION DESCRIPTION

What will be the output when these operations are performed on a Doubly Ended Queue?

1. Insertfront(1);
2. Insertfront(2);
3. Insertrear(3);
4. Insertrear(4);
5. Deletefront();
6. Insertfront(5);
7. Deleterear();
8. Display();

INTERVIEWER GUIDELINES

Here is what dequeue will look like after each operation:

1

2 1

2 1 3

2 1 3 4

1 3 4

5 1 3 4

5 1 3

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ☐ 1, 2, 3
- ☒ 5, 1, 3
- ☐ 4, 2, 3
- ☐ None of the above

No Comments

QUESTION 3



Correct Answer

Score 75

User-Friendly Password System > Coding Medium Strings

QUESTION DESCRIPTION

A website is programming an authentication system that will accept a password either if it's the correct password *or* if it's the correct password with a single character appended to it. In this challenge, your task is to implement such a system, specifically using a hashing function. Given a list of events in which either a password is set or authorization is attempted, determine if each authorization attempt will be successful or not.

The hashing function that will be used in this problem is as follows. Let $f(x)$ be a function that takes a character and returns its decimal character code in the ASCII table. For instance $f('a') = 97$, $f('B') = 66$, and $f('9') = 57$. (You can find all ASCII character codes here: [ASCII table](#).) Then, let $h(s)$ be the hashing function that takes a string and hashes it in the following way, where $p = 131$ and $M = 10^9 + 7$:

$$h(s) := (s[0] \cdot P^{(n-1)} + s[1] \cdot P^{(n-2)} + s[2] \cdot P^{(n-3)} + \dots + s[n-2] \cdot P + s[n-1]) \bmod M$$

For instance, if $s = \text{"cAr1"}$, then the formula would be as follows:

$$h(s) = (f('c') \cdot 131^3 + f('A') \cdot 131^2 + f('r') \cdot 131 + f('1')) \bmod 10^9 + 7 = 223691457$$

Your system will be tested on q event types, each of which will be one of the following:

1. `setPassword(s)` := sets the password to s
2. `authorize(x)` := tries to sign in with integer x . This event must return 1 if x is either the hash of the current password *or* the hash of the current password with a single character appended to it. Otherwise, this event must return 0.

Consider the following example. There are 6 events to be handled:

1. `setPassword("cAr1")`
2. `authorize(223691457)`
3. `authorize(303580761)`
4. `authorize(100)`
5. `setPassword("d")`
6. `authorize(100)`

As we know from the above example, $h(\text{"cAr1"}) = 223691457$, so the second event will return 1. The third event will also return 1 because 303580761 is the hash value of the string "cAr1a", which is equal to the current password with the character 'a' appended to it. The fourth event will return 0 because 100 is not a hash of the current password or of the current password with a single character appended to it. In the fifth event, the current password is set to "d", and the sixth event will return 1 because $h(\text{"d"}) = 100$. Therefore, the array you would return is [1, 1 0, 1], corresponding to the success or failure of the authorization events.

Function Description

Complete the function `authEvents` in the editor below.

`authEvents` has the following parameter(s):

string `events[q][2]`: a 2-dimensional array of strings denoting the event types and event parameters

Returns:

int[number of authorize events]: an array of integers, either 1 or 0, corresponding to the success (1) or failure (0) of each authorization attempt

Constraints

- $2 \leq q \leq 10^5$
- $1 \leq \text{length of } s \leq 9$, where s is a parameter of the `setPassword` event
- $0 \leq x < 10^9 + 7$, where x is the integer value of the parameter of the `authorize` event
- The first event will always be a `setPassword` event.

- There will be at least one authorize event.
- `s` contains only lowercase and uppercase English letters and digits.

▼ Input Format Format for Custom Testing

In the first line, there is a single integer, q , denoting the number of rows in *events*.

In the second line, there is a single integer, 2, denoting the number of columns in *events*.

Each line i of the q subsequent lines (where $0 \leq i < q$) contains two space-separated strings—*events* $[i][0]$ denoting the event type ("setPassword" or "authorize") and *events* $[i][1]$ denoting the event parameter (s or x.)

▼ Sample Case 0

Sample Input

```
4
2
setPassword 000A
authorize 108738450
authorize 108738449
authorize 244736787
```

Sample Output

```
0
1
1
```

Explanation

There are 4 events to process:

1. The first one sets the password to "000A".
2. The second one tries to authorize with the hash value 108738450. This value (which is the hash of the string "000B") doesn't correspond to the current password, nor to the current password with a single character appended to it. Therefore, this event returns 0.
3. The third event tries to authorize with the hash value 108738449. This is indeed the hash value of the current password, so this event returns 1.
4. Finally, the last event tries to authorize with hash value 244736787. This is the hash value of string "000AB", which is valid because it is equal to the current password with a single character appended to it. Therefore, this event returns 1.

▼ Sample Case 1

Sample Input

```
5
2
setPassword 1
setPassword 2
setPassword 3
authorize 49
authorize 50
```

Sample Output

```
0
0
```

Explanation

There are 5 events to process:

1. The first one sets the password to "1".
2. The second one sets the password to "2".
3. The third one sets the password to "3".
4. The fourth event tries to authorize with the hash value 49, which corresponds to "1". Because this is invalid for the current password of "3", this event returns 0.

5. The fifth event tries to authorize with the hash value 50, which corresponds to "2". Because this is invalid for the current password of "3", this event returns 0.

INTERVIEWER GUIDELINES

Editorial (pawel):

For each setPassword event, compute 62 different hashes. 10 for any digit appended to its end, 26 for any lowercase letter appended to its end, and other 26 for any uppercase letter appended to its end. Then, for authorize event, convert the given hash to int and compare it to these 62 hashes. Return 1 if any matches and 0 otherwise.

Setters' solution (pawel):

```
import string

P = 131
MOD = 10**9+7
VALID_CHARS = string.ascii_lowercase + string.ascii_uppercase +
string.digits
SET = "setPassword"
AUTH = "authorize"

def h(s):
    y = 0
    for c in s:
        y = (P*y + ord(c)) % MOD
    return y

def get_hashes(p):
    hashes = set([h(p)])
    for c in VALID_CHARS:
        hashes.add(h(p+c))
    return hashes

def authEvents(events):
    hashes = None
    res = []
    for event_type, param in events:
        if event_type == SET:
            hashes = get_hashes(param)
        else:
            param = int(param)
            res.append(int(param in hashes))
    return res
```

Tester's code:

```
def go_hash(word):
    res = 0
    for i in range(len(word)):
        res *= 131
        res += ord(word[i])
        res %= 1000000000 + 7
    return res

def authEvents(events):
    q = len(events)
    assert(q >= 2 and q <= 100000)
    password = ""
    res = []
    for i in range(q):
        if (events[i][0] == "setPassword"):
            password = events[i][1]
```

```

        assert(len(password) <= 9)
    else:
        words = set()
        words.add(go_hash(password))
        for j in range(48, 58):
            words.add(go_hash(password + chr(j)))
        for j in range(65, 91):
            words.add(go_hash(password + chr(j)))
        for j in range(97, 123):
            words.add(go_hash(password + chr(j)))
        assert(int(events[i][1]) < 1000000000 + 7 and int(events[i]
[1]) >= 0)
        if (int(events[i][1]) in words):
            res.append(1)
        else:
            res.append(0)
    return res

```

CANDIDATE ANSWER

Language used: Python 3

```

1
2 #
3 # Complete the 'authEvents' function below.
4 #
5 # The function is expected to return an INTEGER_ARRAY.
6 # The function accepts 2D_STRING_ARRAY events as parameter.
7 #
8
9 def authEvents(events):
10     # Write your code here
11     result = []
12
13     def calcHash(password):
14         n = len(password)
15         x=0
16         for i in range(n):
17             x += ord(password[i]) * (131 ** (n - i - 1))
18         return x % (10 ** 9 + 7)
19
20     password = ""
21     for event in events:
22         if event[0] == 'setPassword':
23             password = event[1]
24             hash_value = calcHash(password)
25         elif event[0] == 'authorize':
26             x = int(event[1])
27             if x == hash_value:
28                 result.append(1)
29             else:
30                 for c in range(128):
31                     if calcHash(password + chr(c)) == x:
32                         result.append(1)
33                         break
34                 else:
35                     result.append(0)
36
37     return result
38

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	✔ Success	1	0.0235 sec	10.9 KB
TestCase 1	Easy	Sample case	✔ Success	1	0.0266 sec	10.9 KB
TestCase 2	Easy	Sample case	✔ Success	1	0.0246 sec	10.9 KB
TestCase 3	Easy	Hidden case	✔ Success	6	0.0233 sec	10.9 KB
TestCase 4	Easy	Hidden case	✔ Success	6	0.0252 sec	10.8 KB
TestCase 5	Easy	Hidden case	✔ Success	6	5.0351 sec	39.6 KB
TestCase 6	Easy	Hidden case	✔ Success	6	5.1574 sec	39.6 KB
TestCase 7	Easy	Hidden case	✔ Success	6	4.8289 sec	39.6 KB
TestCase 8	Easy	Hidden case	✔ Success	6	5.0134 sec	39.6 KB
TestCase 9	Easy	Hidden case	✔ Success	6	5.1461 sec	39.7 KB
TestCase 10	Easy	Hidden case	✔ Success	6	5.139 sec	39.6 KB
TestCase 11	Easy	Hidden case	✔ Success	6	5.293 sec	39.6 KB
TestCase 12	Easy	Hidden case	✔ Success	6	4.7418 sec	39.7 KB
TestCase 13	Easy	Hidden case	✔ Success	6	5.0784 sec	39.6 KB
TestCase 14	Easy	Hidden case	✔ Success	6	5.0045 sec	39.6 KB

No Comments

QUESTION 4



Correct Answer

Score 75

Profit Targets

> Coding

Binary Search

Data Structures

Medium

Algorithms

Arrays

Problem Solving

Theme: Finance

Interviewer Guidelines

Hashing

QUESTION DESCRIPTION

A financial analyst is responsible for a portfolio of profitable stocks represented in an array. Each item in the array represents the yearly profit of a corresponding stock. The analyst gathers all distinct pairs of stocks that reached the target profit. Distinct pairs are pairs that differ in at least one element.

Given the array *stocksProfit*, find the number of distinct pairs of stocks where the sum of each pair's profits is exactly equal to the target profit.

Example

stocksProfit = [5, 7, 9, 13, 11, 6, 6, 3, 3]

target = 12 profit's target

- There are 4 pairs of stocks that have the sum of their profits equals to the target 12 . Note that because there are two instances of 3 in *stocksProfit* there are two pairs matching (9, 3): *stocksProfits* indices 2 and 7, and indices 2 and 8, but only one can be included.
- There are 3 distinct pairs of stocks: (5, 7), (3, 9), and (6, 6) and the return value is 3.

Function Description

Complete the function *stockPairs* in the editor below.

stockPairs has the following parameter(s):

int stocksProfit[n]: an array of integers representing the stocks profits

long target: a value representing the yearly target profit

Returns:

int: the total number of pairs determined

Constraints

- $1 \leq n \leq 5 \times 10^5$
- $0 \leq \text{stocksProfit}[i] \leq 10^9$
- $0 \leq \text{target} \leq 5 \times 10^9$

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer n , the size of the array *stocksProfit*.

The next n lines each contain an element *stocksProfit*[i] where $0 \leq i < n$.

The next line contains *target*, the target value.

▼ Sample Case 0

Sample Input 0

STDIN	Function
-----	-----
6	→ <i>stocksProfit</i> [] size n = 6
1	→ <i>stocksProfit</i> = [1, 3, 46, 1, 3, 9]
3	
46	
1	
3	
9	
47	→ target = 47

Sample Output 0

1

Explanation 0

There are 4 pairs where *stocksProfit*[i] + *stocksProfit*[j] = 47

1. (*stocksProfit*[0] = 1, *stocksProfit*[2] = 46)
2. (*stocksProfit*[2] = 46, *stocksProfit*[0] = 1)
3. (*stocksProfit*[2] = 46, *stocksProfit*[3] = 1)
4. (*stocksProfit*[3] = 1, *stocksProfit*[2] = 46)

Since all four pairs contain the same values, there is only 1 *distinct* pair of stocks : (1, 46).

▼ Sample Case 1

Sample Input 1

STDIN	Function
-----	-----
7	→ <i>stocksProfit</i> [] size n = 7
6	→ <i>stocksProfit</i> = [6, 6, 3, 9, 3, 5, 1]
6	
3	
9	
3	
5	
1	
12	→ target = 12

Sample Output 1

2

Explanation 1

There are 5 pairs where *stocksProfit*[i] + *stocksProfit*[j] = 12:

1. $(stocksProfit[0] = 6, stocksProfit[1] = 6)$
2. $(stocksProfit[1] = 6, stocksProfit[0] = 6)$
3. $(stocksProfit[2] = 3, stocksProfit[3] = 9)$
4. $(stocksProfit[3] = 9, stocksProfit[2] = 3)$
5. $(stocksProfit[3] = 9, stocksProfit[4] = 3)$
6. $(stocksProfit[4] = 3, stocksProfit[3] = 9)$

The first 2 pairs are the same, as are the last 4. There are only 2 *distinct* pairs of stocks: (3, 9) and (6, 6).

INTERVIEWER GUIDELINES

▼ Hint 1

Is there an efficient way you can find out whether $target - stocksProfit[i]$ exists in the array for every i ?

▼ Hint 2

Multiple occurrences of the same value don't contribute to the final answer except in one special case, $target/2$ when $target$ is even. Try using hash tables.

▼ Solution

Concepts covered: Hash Table

Optimal Solution:

Suppose that we already know the value of the first stock, call it *value*. We can say that the value of the second stock must be $target - value$. Then we just need to find out whether $target - value$ exists in the array. We can do this efficiently using a hash table. One point to notice here is that if $target$ is divisible by 2, then there must be at least two occurrences of $target/2$ in the array for it to contribute in the final answer.

```
def stockPairs(stocksProfit, target):
    stock_values = set(stocksProfit)
    ans = 0
    for value in stock_values:
        if target - value in stock_values and target != 2 * value:
            ans += 1
    if target % 2 == 0 and stocksProfit.count(target // 2) > 1:
        ans += 2
    return ans // 2
```

Brute Force Approach: Passes 13 of 15 test cases

```
def stockPairs(stocksProfit, target):
    values_taken = set()
    ans = 0
    n = len(stocksProfit)
    for i in range(n):
        for j in range(i+1, n):
            if stocksProfit[i] + stocksProfit[j] == target and
            (min(stocksProfit[i], stocksProfit[j]), max(stocksProfit[i],
            stocksProfit[j])) not in values_taken:
                ans += 1
                values_taken.add((min(stocksProfit[i], stocksProfit[j]),
            max(stocksProfit[i], stocksProfit[j])))
    return ans
```

Error Handling: The edge case which candidates must take care is when $target$ is divisible by 2 and the number of occurrences of $target/2$ is equal to 1.

▼ Complexity Analysis

Time Complexity - $O(n)$.

Since we are iterating over each element exactly once and for each element we are doing a lookup in the hash table ($O(1)$ time complexity), each pass costs $O(1)$ time. The overall time complexity is $O(n)$.

Space Complexity - $O(n)$.

The hash table takes $O(n)$ space.

▼ Follow up Question








What if the task is to find out the number of distinct pair of stocks such that their sum is \geq target?

Now, for each element value we need to query the number of integers which are \geq target - value. This can be done using a binary search tree.

CANDIDATE ANSWER

Language used: **Python 3**

```
1 #
2 # Complete the 'stockPairs' function below.
3 #
4 # The function is expected to return an INTEGER.
5 # The function accepts following parameters:
6 # 1. INTEGER_ARRAY stocksProfit
7 # 2. LONG_INTEGER target
8 #
9
10 def stockPairs(stocksProfit, target):
11     # Write your code here
12     visited = set()
13     pairs = set()
14
15     for n in stocksProfit:
16         difference = target - n
17         if difference in visited:
18             pairs.add((min(n, difference), max(n, difference)))
19         visited.add(n)
20
21     return len(pairs)
22
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0: $O(n^2)$	Easy	Sample case	 Success	1	0.0218 sec	10.9 KB
Testcase 1: $O(n^2)$ Edge Case : 2 occurrences of element with value target/2	Easy	Sample case	 Success	1	0.0231 sec	10.8 KB
Testcase 2: $O(n^2)$ Edge Case : 1 occurrence of element with value target/2	Easy	Sample case	 Success	1	0.0222 sec	10.8 KB
Testcase 3: $O(n^2)$	Easy	Hidden case	 Success	2	0.0256 sec	10.8 KB
Testcase 4: $O(n^2)$	Easy	Hidden case	 Success	2	0.0254 sec	10.8 KB
Testcase 5: $O(n^2)$	Easy	Hidden case	 Success	2	0.0235 sec	10.9 KB
Testcase 6: $O(n^2)$	Easy	Hidden case	 Success	2	0.0246 sec	10.9 KB

Testcase 7: $O(n^2)$	Easy	Hidden case	Success	2	0.0268 sec	10.9 KB
Testcase 8: $O(n^2)$	Medium	Hidden case	Success	4	0.0303 sec	11 KB
Testcase 9: $O(n^2)$	Medium	Hidden case	Success	4	0.03 sec	11 KB
Testcase 10: $O(n^2)$	Medium	Hidden case	Success	5	0.0344 sec	11.5 KB
Testcase 11: $O(n^2)$	Medium	Hidden case	Success	5	0.0372 sec	11.7 KB
Testcase 12: $O(n^2)$	Medium	Hidden case	Success	6	0.0302 sec	11.6 KB
Testcase 13: $O(n)$ or $O(n \log n)$	Hard	Hidden case	Success	19	0.1704 sec	20.4 KB
Testcase 14: $O(n)$ or $O(n \log n)$	Hard	Hidden case	Success	19	0.3186 sec	30.2 KB

No Comments

QUESTION 5



Correct Answer

Score 75

Maximizing Element With Constraints > Coding

Medium

Binary Search

QUESTION DESCRIPTION

In this problem, the goal is to determine the maximum value of an element at a certain index in an array of integers that can be constructed under some constraints.

More specifically, n is the desired array size, $maxSum$ is the maximum allowed sum of elements in the array, and k is the index of the element that needs its value to be maximized. The 0-indexed array has the following constraints:

1. The array consists of n positive integers.
2. The sum of all elements in the array is at most $maxSum$.
3. The absolute difference between any two consecutive elements in the array is at most 1.

What is the maximum value of the integer at index k in such an array?

For example, let's say $n = 3$, $maxSum = 6$, and $k = 1$. So, the goal is to find the maximum value of the element at index 1 in an array of 3 positive integers, where the sum of elements is at most 6, and the absolute difference between every two consecutive elements is at most 1.

The maximum such value is 2, and it can be achieved, for example, by the array $[1, 2, 2]$. This array has 3 elements, each of them a positive integer. The sum of the elements does not exceed 6, and the absolute difference between any two consecutive elements is at most 1. There is no other such array that has a larger value at index $k = 1$. Therefore, the answer is 2 because that is the maximum value of the integer at index k .

Function Description

Complete the function `maxElement` in the editor below. The function must return an integer denoting the maximum value of the element at index k given the above constraints.

`maxElement` has the following parameter(s):

int n : the size of the array

int $maxSum$: the maximum allowed sum of the elements in the array

int k : the index of the element in the array where the value needs to be maximized

Returns

int: the maximum value of the element at index k given the above constraints

Constraints

- $1 \leq n \leq \text{maxSum} \leq 10^9$
- $1 \leq k \leq n$

▼ Input Format For Custom Testing

The first line contains an integer, n , denoting the number of elements in the array.

The second line contains an integer, maxSum , denoting the maximum allowed sum of the elements in the array.

The third line contains an integer k , denoting the index of the element in the array where the value needs to be maximized.

▼ Sample Case 0

Sample Input For Custom Testing

```
3
7
1
```

Sample Output

```
3
```

Explanation

In this case, $n = 3$, $\text{maxSum} = 7$, and $k = 1$. So, the goal is to find the maximum value of an element at index 1 in an array of 3 positive integers, where the sum of elements is at most 7, and the absolute difference between every two consecutive elements is at most 1.

The maximum such value is 3, and it is achieved, for example, by the array [2, 3, 2]. This array has 3 elements, each a positive integer. The sum of all elements does not exceed 7, and the absolute difference between any two consecutive elements is at most 1. There is no other such array that has a larger value at index $k = 1$. Therefore, the answer is 3 because that is the maximum value of the integer at index k .

▼ Sample Case 1

Sample Input For Custom Testing

```
4
6
2
```

Sample Output

```
2
```

Explanation

In this case, $n = 4$, $\text{maxSum} = 6$, and $k = 2$. So, the goal is to find the maximum value of an element at index 2 in an array of 4 positive integers, where the sum of elements is at most 6, and the absolute difference between every two consecutive elements is at most 1.

The maximum such value is 2, and it is achieved, for example, by the array [1, 1, 2, 1]. This array has 4 elements, each a positive integer. The sum of all elements does not exceed 6, and the absolute difference between any two consecutive elements is at most 1. There is no other such array that has a larger value at index $k = 2$. Therefore, the answer is 2 because that is the maximum value of the integer at index k .

INTERVIEWER GUIDELINES

This problem can be solved by performing binary search over the answer. In order to check if Andy can get at least x gifts on her birthday, we minimize the number of gifts that she gets on the other days.

This can be done by giving $x-1$ gifts to her on days that are 1 day away from her birthday, $x-2$ gifts on days 2 days away from her birthday and so on until Andy either gets a gift on each of the n days or we reach 1 gift. If we reach 1 gift z days before or after Andy's birthday, then Andy gets 1 gift on all days that are more than z days away from her birthday.

If the number of days before Andy's birthday, $\text{num} > x-1$, the minimum number of gifts required for the days before her birthday is $x*(x-1)/2 + \text{num} - (x-1)$. Else, the minimum number of gifts required = $((2*x - \text{num} - 1)*\text{num})/2$. The same reasoning follows for the days after her birthday.

Setter's solution:

```
bool check(int x,int& n,int& num_gifts,int& m) {
    long long ans=x;
    long long nm=n-m;
    if(nm<x){
        long long cur=(1ll*(2*x-nm-1)*nm)/2;
        ans+=cur;
    }
    else{
        long long cur=((1ll*x*(x-1))/2+nm-(x-1));
        ans+=cur;
    }

    nm= m-1;
    if(nm<x){
        long long cur=(1ll*(2*x-nm-1)*nm)/2;
        ans+=cur;
    }
    else{
        long long cur=((1ll*x*(x-1))/2+nm-(x-1));
        ans+=cur;
    }
    return (ans<=num_gifts);
}

int maxElement(int n, int num_gifts, int m) {
    int lo=1,hi=num_gifts;
    while(lo<hi-1){
        long long mid=(hi+lo)/2;
        if(check(mid,n,num_gifts,m)) lo=mid;
        else hi=mid;
    }
    if(check(hi,n,num_gifts,m)) return hi;
    return lo;
}
```

Tester's Solution:

```
#
# Complete the 'maxChocolates' function below.
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
# 1. INTEGER n
# 2. INTEGER num_chocolates
# 3. INTEGER pos
#

def maxElement(n, num_chocolates, pos):
    # Write your code here

    assert 1 <= pos <= n <= num_chocolates <= 1000000000

    def check(x):

        def sumn(k):
            return (k*(k+1))//2
```

```

def sumx(k):
    if x <= k:
        sum1 = sumn(x) + k - x
    else:
        sum1 = sumn(x) - sumn(x - k)
    return sum1

ans = sumx(pos) + sumx(n - pos + 1) - x
return ans <= num_chocolates

lo = 1
hi = num_chocolates + 1
while hi - lo > 1:
    mid = (hi+lo)//2
    if check(mid):
        lo = mid
    else:
        hi = mid
return lo

```

CANDIDATE ANSWER

Language used: Python 3

```

1
2 #
3 # Complete the 'maxElement' function below.
4 #
5 # The function is expected to return an INTEGER.
6 # The function accepts following parameters:
7 # 1. INTEGER n
8 # 2. INTEGER maxSum
9 # 3. INTEGER k
10 #
11
12 def maxElement(n, maxSum, k):
13     # Write your code here
14
15     def calculateSum(x):
16         left = min(x - 1, k)
17         sum_left = (x - left + x - 1) * left // 2 + (k - left)
18         right = min(x-1,n-k-1)
19         sum_right = (x-right+x-1)*right // 2 + (n - k - 1 - right)
20         return sum_left + sum_right + x
21
22     left, right = 1, maxSum
23
24     while left < right:
25         mid = (left + right + 1) // 2
26         if calculateSum(mid) <= maxSum:
27             left = mid
28         else:
29             right = mid-1
30
31     return left
32

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
-	-	-	-	-	-	-

TestCase 0	Easy	Sample case	✔	Success	1	0.0228 sec	10.9 KB
TestCase 1	Easy	Sample case	✔	Success	1	0.0243 sec	10.9 KB
TestCase 2	Easy	Sample case	✔	Success	1	0.0247 sec	10.8 KB
TestCase 3	Easy	Hidden case	✔	Success	8	0.0255 sec	10.8 KB
TestCase 4	Easy	Hidden case	✔	Success	9	0.0251 sec	10.9 KB
TestCase 5	Easy	Hidden case	✔	Success	2	0.025 sec	10.8 KB
TestCase 6	Easy	Hidden case	✔	Success	2	0.0241 sec	10.9 KB
TestCase 7	Easy	Hidden case	✔	Success	2	0.0239 sec	10.9 KB
TestCase 8	Easy	Hidden case	✔	Success	3	0.0231 sec	10.9 KB
TestCase 9	Easy	Hidden case	✔	Success	3	0.0263 sec	10.8 KB
TestCase 10	Easy	Hidden case	✔	Success	3	0.0236 sec	10.8 KB
TestCase 11	Easy	Hidden case	✔	Success	10	0.0258 sec	10.8 KB
TestCase 12	Easy	Hidden case	✔	Success	10	0.0241 sec	10.8 KB
TestCase 13	Easy	Hidden case	✔	Success	10	0.0217 sec	10.9 KB
TestCase 14	Easy	Hidden case	✔	Success	10	0.0253 sec	10.8 KB

No Comments