You can view this report online at : https://www.hackerrank.com/x/tests/545682/candidates/39705672/report

Full Name:	Lin Xu
Email:	xulin.wh@qq.com
Test Name:	**IB Java Developer Test**
Taken On:	4 May 2022 04:08:03 PDT
Time Taken:	119 min 52 sec/ 120 min
Invited by:	Eduarda
Skills Score:	Problem Solving (Basic) 50/50
Tags Score:	Algorithms 66/125 Combinatorics 16/75 Core Skills 66/125 Databases 10/10 Dynamic Programming 66/125 Easy 110/110 HTML 5/5 Join 10/10 Math 16/75 Medium 16/75 Problem Solving 116/175 Role Based 10/10 SQL 10/10 Strings 100/100

63.5%

146/230

scored in **IB Java Developer Test** in 119 min 52 sec on 4 May 2022 04:08:03 PDT

Candidate Feedback: Great Job

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Object Oriented Class Design > **Multiple Choice**	2 min 25 sec	5/ 5	✓
Q2	CSS > **Multiple Choice**	3 min 5 sec	5/ 5	✓
Q3	DOM > **Multiple Choice**	4 min 52 sec	5/ 5	✓
Q4	Algorithmic Complexity > **Multiple Choice**	1 min 47 sec	5/ 5	✓
Q5	Reading Pseudocode > **Sentence Completion**	5 min 30 sec	0/ 20	✗
Q6	Understanding Scale > **Multiple Choice**	6 min 55 sec	0/ 5	✗

Q7	Are they Pangrams › Coding	15 min 37 sec	50/ 50	✓
Q8	Gem Stones › Coding	36 min 10 sec	50/ 50	✓
Q9	Employees 3 › DbRank	3 min 48 sec	10/ 10	✓
Q10	Volleyball Match › Coding	36 min 56 sec	16/ 75	✓

QUESTION 1

Correct Answer

Score 5

Object Oriented Class Design › Multiple Choice

QUESTION DESCRIPTION

A car dealership needs a program to store information about the cars for sale.

For each car, they want to keep track of the following information:

- Number of doors (2 or 4)
- Whether the car has air conditioning
- Average number of miles per gallon

Which of the following is the best design in an object-oriented language?

INTERVIEWER GUIDELINES

OO-Design is more of a subjective than objective question, but this question is worded in such a way that all the non-correct answers are absolute nonsense.

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ( ) Use a class Car which has three subclasses: Doors, AirConditioning, and MilesPerGallon.
- ( ) Use four unrelated classes: Car, Doors, AirConditioning, and MilesPerGallon.
- ✓ (•) Use one class, Car, which has three data fields: numDoors, hasAirConditioning, milesPerGallon
- ( ) Use a class Car, which has a subclass Doors, with a subclass AirConditioning, with a subclass MilesPerGallon.
- ( ) Use three classes: Doors, AirConditioning, and MilesPerGallon, each with a subclass Car.

No Comments

QUESTION 2

Correct Answer

Score 5

CSS › Multiple Choice

QUESTION DESCRIPTION

What is the correct way to reference an external stylesheet?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- <css type="text/stylesheet" value="my-external-stylesheet.css"></css>
- ✓ <link rel="stylesheet" type="text/css" href="my-external-stylesheet.css" />
- <style type="text/css" href="my-external-stylesheet.css"></style>
- <style rel="stylesheet" src="my-external-stylesheet.css" />

No Comments

QUESTION 3

Correct Answer

Score 5

DOM › Multiple Choice HTML

QUESTION DESCRIPTION

Given an HTML Document, how would you retrieve all Anchor Elements?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ✓ document.getElementsByTagName("a");
- document.getTagsByElementName("a");
- document.getElementsById("a")
- document.getElementsByName("a");

No Comments

QUESTION 4

Correct Answer

Score 5

Algorithmic Complexity › Multiple Choice

QUESTION DESCRIPTION

If I used the following pseudo-code to search for the largest sum of consecutive integers:

```
highestSum = array[0];
for(i from 0 to array.length)
  sum = 0
  for(j from i to array.length)
    sum += array[j]
    if(sum > highestSum) then
      sum = highestSum
```

What would be the average-case algorithmic complexity of that approach?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- O(1)
- O(log n)
- O(n)
- O(n log n)
- ✓ O(n^2)
- O(n^3)
- O(c^n)
- O(n!)

No Comments

QUESTION 5

Wrong Answer

Score 0

Reading Pseudocode › Sentence Completion

QUESTION DESCRIPTION

Problem Statement

Given the following pseudocode:

```
x = 1
y = 1
while x < 25
  z = x
  x = x + y
  y = z
```

Complete String

When this program terminates, the value of x will be {blank}.

CANDIDATE ANSWER

When this program terminates, the value of x will be 1 .

No Comments

QUESTION 6

Wrong Answer

Score 0

Understanding Scale › Multiple Choice

QUESTION DESCRIPTION

If it takes a computer approximately 1 millisecond to iterate through all the possible 32 bit unsigned integers, about how long will it take that computer to iterate through all possible 64 bit unsigned integers?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- 2ms
- 32ms
- 64ms
- 32 seconds
- 12 hours
- 17 days
- ✓ 7 weeks
- 4 months
- 3 years
- It's not possible to iterate through all of the 64 bit integers

No Comments

QUESTION 7

Correct Answer

Score 50

Are they Pangrams › Coding Easy Strings Problem Solving

QUESTION DESCRIPTION

A string is a pangram if it contains all letters of the English alphabet, ascii['a'-'z']. Given a list of strings, determine if each one is a pangram or not. Return "1" if true and "0" if false.

Example

pangram = ['pack my box with five dozen liquor jugs', 'this is not a pangram']

- the string *'pack my box with five dozen liquor jugs'* is a pangram , because it contains all the letters 'a' through 'z'
- the string *'this is not a pangram' is not a pangram*
- Assemble a string of the two results, in order. The result is *'10'.*

Function Description Complete the function *isPangram* n the editor below.

isPangram has the following parameter(s):

string pangram[n]: an array of strings

Returns:

string: a string where each position represents the results of a test. Use '*1*' for true and '*0*' for false.

Constraints

- $1 \le n \le 100$
- Each string *pangram[i]* (where $0 \le i < n$) is composed of *lowercase letters* and *spaces*.
- $1 \le$ *length of pangram[i]* $\le 10^5$

▼ 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 *pangram*.
The next *n* lines each contain an element, *pangram[i],* where *0 ≤ i < n*.

▼ Sample Case 0

Sample Input 0

```
STDIN                                                          Function
Parameters
-----                                                          ---------
----------
4                                                          →   pangram[]
size n = 4
we promptly judged antique ivory buckles for the next prize    →   pangram[]
= ["we promptly judged antique ivory buckles for the next prize",
we promptly judged antique ivory buckles for the prizes
"we promptly judged antique ivory buckles for the prizes",
the quick brown fox jumps over the lazy dog
"the quick brown fox jumps over the lazy dog",
the quick brown fox jump over the lazy dog
"the quick brown fox jump over the lazy dog" ]
```

Sample Output 0

```
1010
```

Explanation 0

```
pangram[0] = True
pangram[1] = False
pangram[2] = True
pangram[3] = False
```

- The strings *pangram[0]* and *pangram[2]* are pangrams, and the others are not.
- The result is *'1010'*

▼ Sample Case 1

```
STDIN
Function Parameters
-----
--------------------
4
→ pangram[] Size n = 4
cfchcfcvpalpqxenhbytcwazpxtthjumliiobcznbefnofyjfsrwfecxcbmoafes tnulqkvx
oxhctvhybtikkgeptqulzukfmmavacshugpouxoliggcomykdnfayayqutgwivwldrkp
gpecfrak zzaxrigltstcrdyhelhz rasrzibduaq  cnpuommogatqem
hbybsegucruhxkebrvmrmwhweirx mbkluwhfapjtga liiylfphmzkq
```

Sample Output 1

```
0000
```

Explanation 1

```
pangram[0] = False
pangram[1] = False
```

```
pangram[2] = False
pangram[3] = False
```

- No string is a pangram.
- The result is *'0000'*

INTERVIEWER GUIDELINES

▼ Hint 1

How can you keep track of when a character has been seen? Answer: Have a *seen* array of 26 integers initialized to zero. As a character is seen, update the array at the index ord(character) - ord('a') to 1. Ignore spaces.

▼ Hint 2

How will you know when all letters have been seen?

Answer: Each time you update the *seen* array, increment a counter.

▼ Solution

Skills: Iterating through strings, problem solving, data structures (set or array)

Brute Force Approach: A simple solution that analyzes every character of the string is to create a set from the alphabet, create a set from the string, remove the space character from the set and get their intersection. If the intersection equals the alphabet set, the string is a pangram. Creating a set takes O(n) running time.

Optimal Solution:

The most efficient solution starts with an *seen* array of 26 zeros, and a counter initialized to 0. Iterate through the string, checking if a character has already been seen. If it has not, update the value in the *seen* array and increment the counter. When the counter equals 26, it is a pangram. If you reach the end of the string and the counter is less than 26, it is not a pangram. Note that the test cases are such that a solution will work even if they iterate through all characters in every string. A minor optimization is to break out of the loop when the counter reaches 26.

```
def isPangram(pangram):
    # Write your code here
    answer = ''
    for sentence in pangram:
        seen = [0] * 26
        count = 0
        for c in sentence:
            if not c == ' ':
                idx = ord(c) - ord('a')
                if seen[idx] == 0:
                    seen[idx] += 1
                    count += 1
                if count == 26:
                    break
        answer += '1' if count == 26 else '0'
    return answer
```

▼ Complexity Analysis

Time Complexity - O(n).

All characters of the string need to be checked if it is not a pangram.

Space Complexity - O(1) - Constant extra space is required.

Regardless of string size, only an additional 26 element array and a counter variable are required.

▼ Follow up Question

What if we just wanted to identify if a subset of the alphabet is included in a string? For

example, the subset is all the characters in "aeiou"?

Psuedo Code -

```
def containsChars(testString, universe):
    # mark all as seen
    seen = [1] * 26
    # mark all characters in universe as unseen
    # also, count unique characters
    uc = 0
    for c in universe:
        idx = ord(c) - ord('a')
        if seen[idx]:
            uc += 1
            seen[idx] = 0
    # now do just as in the pangrams exercise
    # but test against the unique count instead of 26
    count = 0
    for c in testString:
        idx = ord(c) - ord('a')
        if not c == ' ' and seen[idx] == 0:
            count+= 1
            seen[idx] = 1
        if count == uc:
            break
    return True if count == uc else False
```

CANDIDATE ANSWER

Language used: **Java 15**

```
class Result {

    /*
     * Complete the 'isPangram' function below.
     *
     * The function is expected to return a STRING.
     * The function accepts STRING_ARRAY pangram as parameter.
     */

    public static String isPangram(List<String> pangram) {
        int size = pangram.size();
        List<String> list = new ArrayList<>();
        for(int j = 0; j < size; j++ ) {
            boolean[] alphaList = new boolean[26];
            int index = 0;
            int flag = 1;
            String str = pangram.get(j);
            for(int i = 0; i < str.length(); i++ ) {
                if ( str.charAt(i) >= 'A' && str.charAt(i) <= 'Z') {
                    index = str.charAt(i) - 'A';
                } else if ( str.charAt(i) >= 'a' && str.charAt(i) <= 'z') {
                    index = str.charAt(i) - 'a';
                }
                alphaList[index] = true;
            }
            for(int k = 0; k <= 25; k++) {
                if(alphaList[k] == false) {
                    flag = 0;
                }
            }
            list.add(flag+"");
        }
        StringBuffer stringBuffer = new StringBuffer();
```

```
        for(String str: list) {
            stringBuffer.append(str);
        }

        return stringBuffer.toString();
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	1	0.0875 sec	33.2 KB
TestCase 1	Easy	Sample case	Success	1	0.105 sec	32.9 KB
TestCase 2	Easy	Sample case	Success	1.5	0.1572 sec	32.8 KB
TestCase 3	Easy	Sample case	Success	1.5	0.1002 sec	33.4 KB
TestCase 4	Easy	Sample case	Success	5	0.1561 sec	33.3 KB
TestCase 5	Easy	Hidden case	Success	5	0.153 sec	35.2 KB
TestCase 6	Easy	Hidden case	Success	5	0.144 sec	38 KB
TestCase 7	Easy	Hidden case	Success	5	0.1415 sec	37.8 KB
TestCase 8	Easy	Hidden case	Success	5	0.1874 sec	35.3 KB
TestCase 9	Easy	Hidden case	Success	5	0.2265 sec	40.7 KB
TestCase 10	Easy	Hidden case	Success	5	0.2178 sec	40 KB
TestCase 11	Easy	Hidden case	Success	5	0.1772 sec	40.6 KB
TestCase 12	Easy	Hidden case	Success	5	0.2702 sec	39.9 KB

No Comments

QUESTION 8

Correct Answer

Score 50

Gem Stones › Coding

Easy | Strings | Algorithms | Dynamic Programming | Core Skills | Problem Solving

QUESTION DESCRIPTION

Charles has found a rock collection in the basement of his deceased grandmother's house. Each rock is composed of various elements, and each element is represented by a lowercase English letter from 'a' to 'z'. An element can be present multiple times in a rock. An element is called a 'gem-element' if it occurs at least once in each of the rocks.

Complete the function "*gemstones*" which contains *1* parameter:

- A string array rocks where i^{th} element denotes the composition of i^{th} rock.

It must return the number of gem-elements that exist in those rocks.

Input Format

The locked stub code in your editor reads the following input from stdin and passes it to your function:
The first line contains an integer *n*.
Next *n* lines contain a string in each line, where string in i^{th} line denotes *rock[i]*.

Constraints:

- $1 \leq n \leq 100$
- Each composition consists of only small latin letters ('a'-'z').

- Each composition consists of only small latin letters ('a'-'z').
- *1* ≤ Length of each composition ≤ *100*

Output Format

Your function must return a single integer that gives the number of gem-elements that are common in these rocks. If there are none, return 0. This is printed to stdout by the locked stub code in your editor.

Sample Input 1

```
3
abcdde
baccd
eeabg
```

Sample Output 1

```
2
```

Explanation 1

Only 'a' and 'b' occur in each of the 3 rocks.

Sample Input 2

```
3
aa
bb
cc
```

Sample Output 2

```
0
```

Explanation 2

There are no gemstones which are present in all the rocks.

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'gemstones' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts STRING_ARRAY rocks as parameter.
     */

    public static int gemstones(List<String> rocks) {
    // Write your code here
        int size = rocks.size();
        List<Set<Character>> listSet = new ArrayList<>();
        for(int i = 0; i <size; i++) {
```

```
            Set<Character> set = new HashSet<>();
            String str = rocks.get(i);
           for(int k = 0; k < str.length(); k++) {
               set.add(str.charAt(k));
           }
            listSet.add(set);
        }
        List<Set<Character>> setList = new ArrayList<>();
        for(Set<Character> set : listSet) {
            setList.add(set);
        }

        List<Set<Character>> resultSetList =
setList.stream().filter(Objects::nonNull).collect(Collectors.toList());
        Set<Character> resultSet = setList.get(0);
        resultSetList.forEach(item -> {
            resultSet.retainAll(item);
        });

        return resultSet.size();
    }

}

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	1	0.132 sec	29.8 KB
Corner Case	Easy	Sample case	Success	1	0.1645 sec	30.2 KB
Testcase 2	Medium	Hidden case	Success	1	0.1646 sec	30 KB
Testcase 3	Easy	Hidden case	Success	2	0.1349 sec	30.1 KB
Testcase 4	Easy	Hidden case	Success	2	0.202 sec	30.2 KB
Testcase 5	Easy	Hidden case	Success	2	0.186 sec	29.9 KB
Testcase 6	Medium	Hidden case	Success	4	0.1899 sec	30 KB
Testcase 7	Medium	Hidden case	Success	4	0.1692 sec	30 KB
Testcase 8	Medium	Hidden case	Success	4	0.1276 sec	30.2 KB
Testcase 9	Hard	Hidden case	Success	7	0.2151 sec	30.3 KB
Testcase 10	Hard	Hidden case	Success	8	0.1772 sec	30.3 KB
Testcase 11	Hard	Hidden case	Success	7	0.1772 sec	30.2 KB
Testcase 26	Hard	Hidden case	Success	7	0.2143 sec	30.4 KB

No Comments

QUESTION 9

Correct Answer

Score 10

Employees 3 › DbRank

SQL | Easy | Join | Role Based | Databases

QUESTION DESCRIPTION

You are given two tables: *EMPLOYEES* and *EMPLOYEE_PAN*. Write a query to print the employee *NAME* and *UIN* in each line. If no *UIN* is present print *NULL* in its place. The order of output does not matter.

Input Format

EMPLOYEES

Name	Type	Description
ID	Integer	The ID of the employee.This is a primary key.
NAME	String	The name of the employee having *[1, 20]* characters.
USERS	Integer	The number of users.
AGE	Integer	The age of the employee.
ADDRESS	String	The address of the employee having *[1, 25]* characters.
SALARY	Float	The salary of the employee.

EMPLOYEE_PAN		
Name	Type	Description
ID	Integer	The ID of the employee.This is a primary key.
UIN	Integer	The unique identification number of the employee.

Sample Input

EMPLOYEES					
ID	NAME	USERS	AGE	ADDRESS	SALARY
2	Ileana	9162	27	Paris	74635.00
3	Bob	6170	30	Sydney	72167.00
4	Julia	1533	29	Paris	75299.00
1	Samantha	1694	47	Sydney	46681.00
5	David	6486	27	Texas	11843.00

EMPLOYEE_PAN	
ID	UIN
1	23
3	27
2	45

Sample Output

```
Samantha 23
Ileana 45
Bob 27
Julia NULL
David NULL
```

CANDIDATE ANSWER

Language used: **MySQL**

```
/*
Enter your query here.
*/

SELECT t.NAME,ep.UIN FROM EMPLOYEES t LEFT JOIN EMPLOYEE_PAN ep ON t.id=ep.id
```

Time taken: **0.22 sec**

No Comments

QUESTION 10

Correct Answer

Score 16

Volleyball Match › Coding

Medium | Algorithms | Dynamic Programming | Math | Combinatorics | Core Skills | Problem Solving

QUESTION DESCRIPTION

Sally is a big sports fan, especially volleyball. She has a habit of writing down the final scores of each game after it has ended in her notebook.

If you are not familiar with the rules of volleyball, here's a brief:

- 2 teams play a match
- During the course of the game, each team gets points, and thus increases its score by 1.
- The initial score is 0 for both teams.

The game ends when

- One of the teams gets 25 points and another team has < 24 points (strictly less than 24).
- If the score ties at 24:24, the teams continue to play until the absolute difference between the scores is 2.

Given the final score of a game in the format *A*:*B* i.e., the first team has scored *A* points and the second has scored *B* points, can you find the number of different sequences of getting points by teams that leads to this final score?

Complete the function ***volleyball*** in your editor. It has *2* parameters:

1. An integer *A*.
2. An integer *B*.

It must return the number of different possible sequences of getting those points. As the answer could be very large, return the value of *result % (10^9 + 7)*.

Input Format

The locked stub code in your editor reads the following input from stdin and passes it to your function:
The first line contains a single integer *A*.
The next line contains a single integer *B*.

Constraints

- $0 \leq A,B \leq 10^9$

Output Format

The locked code in the editor prints the return value of the function.
Your function must return the number of different possible sequences of getting those points. As the answer could be very large, return the value of *result % (10^9 + 7)*.

Sample Input 1

```
3
25
```

Sample Output 1

```
2925
```

Explanation 1

There are 2925 different sequences to reach the score (3,25).

Sample Input 2

```
24
17
```

Sample Output 2

```
0
```

Explanation 2

There can be no game of volleyball that ends with a score of 24 : 17.

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'volleyball' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts following parameters:
     *  1. INTEGER A
     *  2. INTEGER B
     */

    public static int volleyball(int A, int B) {
    // Write your code here
        int total = 0;
        if(A < 25 && B < 25 ) {
            return 0;
        } else if (A == 25 && B == 25) {
            return 0;
        } else if( A > 25 && B > 25 && (Math.abs(A - B) != 2)) {
            return 0;
        } else {
            if (A > B) {
                for(int i = 0; i < A; i++) {
                    for(int j = 0; j < B; j++) {
                        total = total + j;
                    }
                }
            } else {
                for(int i = 0; i <  (B); i++) {
                    for(int j = 0; j < ( A +1) * (B); j++) {
                         total = total + 1;
                    }
                   // total = total - 1;
                }
            }
        }
        return total;
```

```
        return total;
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Wrong Answer	0	0.08 sec	23.4 KB
Testcase 1	Easy	Sample case	Success	1	0.0727 sec	23.6 KB
Testcase 2	Easy	Hidden case	Wrong Answer	0	0.096 sec	23.6 KB
Testcase 3	Easy	Hidden case	Success	5	0.0624 sec	23.6 KB
Testcase 4	Easy	Hidden case	Wrong Answer	0	0.082 sec	23.5 KB
Testcase 5	Easy	Hidden case	Wrong Answer	0	0.0926 sec	23.6 KB
Testcase 6	Medium	Hidden case	Success	5	0.0888 sec	23.6 KB
Testcase 7	Medium	Hidden case	Success	5	0.0871 sec	23.4 KB
Testcase 8	Medium	Hidden case	Terminated due to timeout	0	4.0044 sec	23.7 KB
Testcase 9	Medium	Hidden case	Wrong Answer	0	0.157 sec	23.7 KB
Testcase 10	Easy	Hidden case	Wrong Answer	0	0.1114 sec	23.5 KB
Testcase 11	Medium	Hidden case	Terminated due to timeout	0	4.0062 sec	24 KB
Testcase 12	Easy	Hidden case	Wrong Answer	0	0.1071 sec	23.5 KB
Testcase 13	Easy	Hidden case	Wrong Answer	0	0.1033 sec	23.6 KB
Testcase 14	Easy	Hidden case	Wrong Answer	0	0.1116 sec	23.6 KB

No Comments

PDF generated at: 4 May 2022 13:10:14 UTC