

# Ch.siddartha

**Test ID:** 450011945000099 | 7013749956 | psid445@gmail.com

**Test Date:** July 5, 2025

<b>Computer Science</b>  <b>53</b> /100	<b>Logical Ability</b>  <b>69</b> /100	<b>Computer Programming</b>  <b>46</b> /100	<b>Quantitative Ability (Advanced)</b>  <b>51</b> /100
<b>English Comprehension</b>  <b>62</b> /100	<b>WriteX - Essay Writing</b>  <b>66</b> /100	<b>Automata</b>  <b>100</b> /100	<b>Automata Fix</b>  <b>29</b> /100
<b>Personality</b>  <b>Completed</b>			

Computer Science			<b>53</b> / 100
OS and Computer Architecture	DBMS	Computer Networks	
<b>68</b> / 100	<b>42</b> / 100	<b>27</b> / 100	

Logical Ability			<b>69</b> / 100
Inductive Reasoning	Deductive Reasoning	Abductive Reasoning	
<b>68</b> / 100	<b>73</b> / 100	<b>66</b> / 100	

## Computer Programming

46 / 100

Basic Programming

38 / 100

Data Structures

47 / 100

OOP and Complexity Theory

53 / 100

## Quantitative Ability (Advanced)

51 / 100

Basic Mathematics

52 / 100

Advanced Mathematics

50 / 100

Applied Mathematics

51 / 100

## English Comprehension

62 / 100

CEFR: **B2**

Grammar

70 / 100

Vocabulary

53 / 100

Comprehension

64 / 100

## WriteX - Essay Writing

66 / 100

CEFR: **B1**

Content Score

63 / 100

Grammar Score

75 / 100

## Automata

100 / 100

Programming Ability

100 / 100

Programming Practices

100 / 100

Functional Correctness

100 / 100

## Automata Fix

29 / 100

Logical Error

50 / 100

Code Reuse

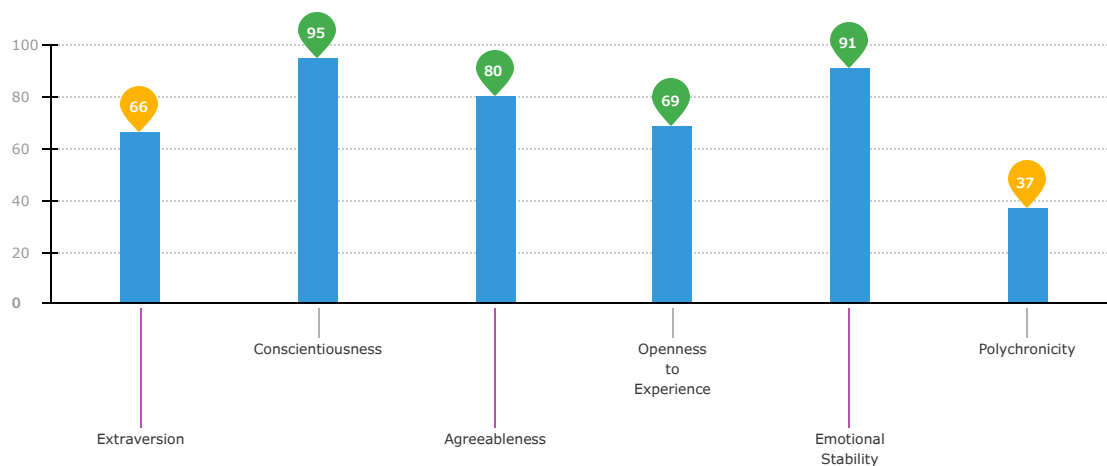
0 / 100

Syntactical Error

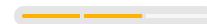
0 / 100

# Personality

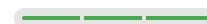
Completed



People Interaction



Self-Drive



Trainability



Repetitive Job Suitability



Work attributes

## 1 | Introduction

### About the Report

This report provides a detailed analysis of the candidate's performance on different assessments. The tests for this job role were decided based on job analysis, O\*Net taxonomy mapping and/or criterion validity studies. The candidate's responses to these tests help construct a profile that reflects her/his likely performance level and achievement potential in the job role

This report has the following sections:

The **Summary** section provides an overall snapshot of the candidate's performance. It includes a graphical representation of the test scores and the subsection scores.

The **Insights** section provides detailed feedback on the candidate's performance in each of the tests. The descriptive feedback includes the competency definitions, the topics covered in the test, and a note on the level of the candidate's performance.

The **Response** section captures the response provided by the candidate. This section includes only those tests that require a subjective input from the candidate and are scored based on artificial intelligence and machine learning.

The **Learning Resources** section provides online and offline resources to improve the candidate's knowledge, abilities, and skills in the different areas on which s/he was evaluated.

### Score Interpretation

All the test scores are on a scale of 0-100. All the tests except personality and behavioural evaluation provide absolute scores. The personality and behavioural tests provide a norm-referenced score and hence, are percentile scores. Throughout the report, the colour codes used are as follows:

- Scores between 67 and 100
- Scores between 33 and 67
- Scores between 0 and 33

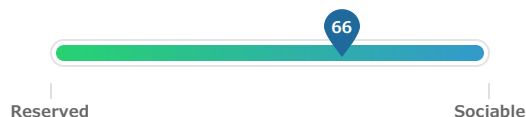
## 2 | Insights

### Personality

#### Competencies



#### Extraversion

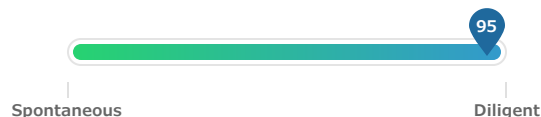


Extraversion refers to a person's inclination to prefer social interaction over spending time alone. Individuals with high levels of extraversion are perceived to be outgoing, warm and socially confident.

- You are comfortable socializing to a certain extent. You prefer small gatherings in familiar environments.
- You feel at ease interacting with your close friends but may be reserved among strangers.
- You indulge in activities involving thrill and excitement that are not too risky.
- You contemplate the consequences before expressing any opinion or taking an action.
- You take charge when the situation calls for it and you are comfortable following instructions as well.
- Your personality may be suitable for jobs demanding flexibility in terms of working well with a team as well as individually.



#### Conscientiousness



Conscientiousness is the tendency to be organized, hard working and responsible in one's approach to your work. Individuals with high levels of this personality trait are more likely to be ambitious and tend to be goal-oriented and focused.

- You value order and self discipline and tends to pursue ambitious endeavours.
- You believe in the importance of structure and is very well-organized.
- You carefully review facts before arriving at conclusions or making decisions based on them.
- You strictly adhere to rules and carefully consider the situation before making decisions.
- You tend to have a high level of self confidence and do not doubt your abilities.
- You generally set and work toward goals, try to exceed expectations and are likely to excel in most jobs, especially those which require careful or meticulous approach.



## Agreeableness



Agreeableness refers to an individual's tendency to be cooperative with others and it defines your approach to interpersonal relationships. People with high levels of this personality trait tend to be more considerate of people around them and are more likely to work effectively in a team.

- You are considerate and sensitive to the needs of others.
- You tend to put the needs of others ahead of your own.
- You are likely to trust others easily without doubting their intentions.
- You are compassionate and may be strongly affected by the plight of both friends and strangers.
- You are humble and modest and prefer not to talk about personal accomplishments.
- Your personality is more suitable for jobs demanding cooperation among employees.



## Openness to Experience

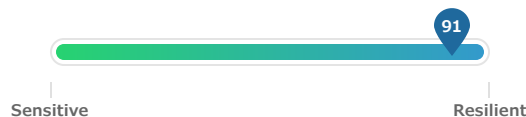


Openness to experience refers to a person's inclination to explore beyond conventional boundaries in different aspects of life. Individuals with high levels of this personality trait tend to be more curious, creative and innovative in nature.

- You tend to be curious in nature and is generally open to trying new things outside your comfort zone.
- You may have a different approach to solving conventional problems and tend to experiment with those solutions.
- You are creative and tends to appreciate different forms of art.
- You are likely to be in touch with your emotions and is quite expressive.
- Your personality is more suited for jobs requiring creativity and an innovative approach to problem solving.



## Emotional Stability



Emotional stability refers to the ability to withstand stress, handle adversity, and remain calm and composed when working through challenging situations. People with high levels of this personality trait tend to be more in control of their emotions and are likely to perform consistently despite difficult or unfavourable conditions.

- You are calm and composed in nature.
- You tend to maintain composure during high pressure situations.
- You are very confident and comfortable being yourself.
- You find it easy to resist temptations and practice moderation.
- You are likely to remain emotionally stable in jobs with high stress levels.



## Polychronicity



Polychronicity refers to a person's inclination to multitask. It is the extent to which the person prefers to engage in more than one task at a time and believes that such an approach is highly productive. While this trait describes the personality disposition of a person to multitask, it does not gauge their ability to do so successfully.

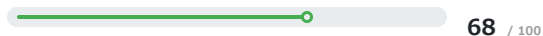
- You neither have a strong preference nor dislike to perform multiple tasks simultaneously.
- You are open to both options - pursuing multiple tasks at the same time or working on a single project at a time.
- Whether or not you will succeed in a polychronous environment depends largely on your ability to do so.

## Logical Ability

69 / 100



### Inductive Reasoning

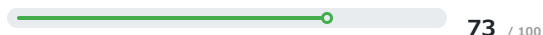


This competency aims to measure the your ability to synthesize information and derive conclusions.

It is commendable that you have excellent inductive reasoning skills. You are able to make specific observations to generalize situations and also formulate new generic rules from variable data.



### Deductive Reasoning

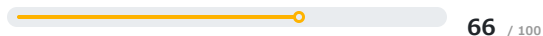


This competency aims to measure the your ability to synthesize information and derive conclusions.

It is commendable that you have excellent inductive reasoning skills. You are able to make specific observations to generalize situations and also formulate new generic rules from variable data.



### Abductive Reasoning



## Quantitative Ability (Advanced)

51 / 100

This test aims to measure your ability to solve problems on basic arithmetic operations, probability, permutations and combinations, and other advanced concepts.

You are good at basic arithmetic. You are able to solve real-world problems that involve simple addition, subtraction, multiplication and division.

## English Comprehension

62 / 100

CEFR: **B2**

This test aims to measure your vocabulary, grammar and reading comprehension skills.

You have a good understanding of commonly used grammatical constructs. You are able to read and understand articles, reports and letters/emails related to your day-to-day work. The ability to read, understand and interpret business-related documents is essential in most jobs, especially the ones that involve research, technical reading and content writing.



### 3 | Response

#### Automata



100 / 100

[Code Replay](#)

#### Question 1 (Language: Python3.7)

An e-commerce company is planning to give a special discount on all its products to its customers for the holiday. The company possesses data on its stock of N product types. The data for each product type represents the count of customers who have ordered the given product. If the data K is positive then the product has been ordered by K customers and is in stock. If the data K is negative then the product has been ordered by K customers but is not in stock. The company will fulfill the order directly if the ordered product is in stock. If it is not in stock then the company will fulfill the order after they replenish the stock from the warehouse. They are planning to offer a discount amount A for each product. The discount value will be distributed to the customers who have purchased that selected product. The discount will be distributed only if the discount amount A can be divided by the number of orders for a particular product.

Write an algorithm for the sales team to find the number of products out of N for which the discount will be distributed.

#### Scores

##### Programming Ability



100 / 100

Completely correct. A correct implementation of the problem using the right control-structures and data dependencies.

##### Functional Correctness



100 / 100

Functionally correct source code. Passes all the test cases in the test suite for a given problem.

##### Programming Practices



100 / 100

High readability, high on program structure. The source code is readable and does not consist of any significant redundant/improper coding constructs.

#### Final Code Submitted

Compilation Status: Pass

```

1
2 """
3
4 """
5 def noOfProducts(order, disAmount):
6     # Write your code here
7     c = 0
8     for i in order:
9         if disAmount%i==0:
10            c+=1

```

#### Code Analysis

##### Average-case Time Complexity

Candidate code:  $O(N)$

Best case code:  $O(N)$

\*N represents number of different types of products.

##### Errors/Warnings

```

11
12     return c
13
14 def main():
15     #input for order
16     order = []
17     order_size = int(input())
18     order = list(map(int,input().split()))
19
20     #input for disAmount
21     disAmount = int(input())
22
23
24     result = noOfProducts(order, disAmount)
25     print(result)
26
27 if __name__ == "__main__":
28     main()

```

There are no errors in the candidate's code.

#### Structural Vulnerabilites and Errors

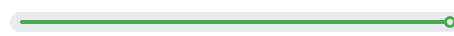
#### Readability & Language Best Practices

Line 7,10: Variable name doesn't conform to snake\_case naming style

#### Test Case Execution

Passed TC: 100%

Total score

 11/11

**100%**

Basic(6/6)

**100%**

Advance(2/2)

**100%**

Edge(3/3)

#### Compilation Statistics

1

Total attempts

1

Successful

0

Compilation errors

0

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:08:30

Average time taken between two compile attempts:

00:08:30

Average test case pass percentage per compile:

100%

## Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

## Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

### Question 2 (Language: Python3.7)

Design a way to sort the list of positive integers in descending order according to frequency of the elements. The elements with higher frequency come before those with lower frequency. Elements with the same frequency come in the same order as they appear in the given list.

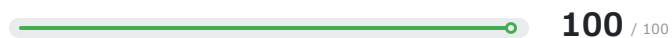
#### Scores

##### Programming Ability



Completely correct. A correct implementation of the problem using the right control-structures and data dependencies.

##### Functional Correctness



Functionally correct source code. Passes all the test cases in the test suite for a given problem.

##### Programming Practices



High readability, high on program structure. The source code is readable and does not consist of any significant redundant/improper coding constructs.

#### Final Code Submitted

Compilation Status: Pass

```
1
2 """
3
4 """
5 from collections import Counter
```

#### Code Analysis

##### Average-case Time Complexity

Candidate code:  $O(N)$

```

6 def freqSort(listEle):
7     # Write your code here
8
9     a = []
10    b = []
11    c = []
12    d = Counter(listEle)
13    for i in range(len(listEle)):
14        if listEle[i] not in b:
15            a.append([d[listEle[i]],i+1])
16            b.append(listEle[i])
17        else:
18            continue
19    e = sorted(a,reverse = True,key = lambda x:x[0])
20    # return e
21    for i in e:
22        m,n = i
23        c.extend([listEle[n-1] for _ in range(m)])
24
25    return c
26
27 def main():
28     #input for listEle
29     listEle = []
30     listEle_size = int(input())
31     listEle = list(map(int,input().split()))
32
33
34     result = freqSort(listEle)
35     print(" ".join([str(res) for res in result]))
36
37 if __name__ == "__main__":
38     main()

```

**Best case code:**  $O(N)$

\*N represents number of elements in the input array

#### Errors/Warnings

There are no errors in the candidate's code.

#### Structural Vulnerabilites and Errors

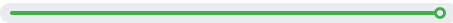
#### Readability & Language Best Practices

Line 9,10,11...: Variable name doesn't conform to snake\_case naming style

#### Test Case Execution

Passed TC: 100%

Total score

 19/19

**100%**

Basic(5/5)

**100%**

Advance(13/13)

**100%**

Edge(1/1)

## Compilation Statistics

13

Total attempts

13

Successful

0

Compilation errors

0

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:35:53

Average time taken between two compile attempts:

00:02:46

Average test case pass percentage per compile:

38.46%

### Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Automata Fix



29 / 100

[Code Replay](#)

### Question 1 (Language: Java)

The function/method **manchester** print space-separated integers with the following property: for each element in the input array *arr*, a counter is incremented if the bit *arr[i]* is the same as *arr[i-1]*. Then the increment counter value is added to the output array to store the result.

If the bit *arr[i]* and *arr[i-1]* are different, then 0 is added to the output array. For the first bit in the input array, assume its previous bit to be 0. For example, if *arr* is {0,1,0,0,1,1,1,0}, the function/method should print 1 0 0 2 0 3 4 0.

The function/method **manchester** accepts two arguments- *size*, an integer representing the length of the input array; *arr*, a list of integers representing an input array. Each element of *arr* represents a bit, 0 or 1.

The function/method **manchester** compiles successfully but fails to print the desired result for some test cases due to logical errors. Your task is to fix the code so that it passes all the test cases.

## Scores

Final Code Submitted	Compilation Status: Pass	Code Analysis
<pre> 1 public class Solution 2 { 3     void manchester(int size, int[] arr) 4     { 5         int res[] = new int[size]; 6         boolean result; 7         int count =0; 8         for(int i = 0; i &lt; size; i++) 9         { 10             if(i==0) 11                 result= (arr[i]==0); 12             else 13                 result = (arr[i]==arr[i-1]); 14             res[i] = (result)?(++count):(0); 15         } 16     } 17     for(int i=0; i&lt;size; i++) 18     { 19         System.out.print(res[i]+" "); 20     } 21 } 22 }</pre>		<b>Average-case Time Complexity</b> <p><b>Candidate code:</b> Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.</p> <p><b>Best case code:</b></p> <p>*N represents</p>
		<b>Errors/Warnings</b> <p>There are no errors in the candidate's code.</p>
		<b>Structural Vulnerabilites and Errors</b> <p>There are no errors in the candidate's code.</p>
Test Case Execution		Passed TC: 100%
Total score <div> <span>7/7</span> </div>		<div> <b>100%</b> Basic(5/5)                 </div> <div> <b>100%</b> Advance(1/1)                 </div> <div> <b>100%</b> Edge(1/1)                 </div>

## Compilation Statistics

4

Total attempts

4

Successful

0

Compilation errors

3

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:07:26

Average time taken between two compile attempts:

00:01:52

Average test case pass percentage per compile:

25%

### Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Question 2 (Language: Java)

The function/method ***findMaxElement*** return an integer representing the largest element in the given two input lists. The function/method ***findMaxElement*** accepts four arguments - *len1*, an integer representing the length of the first list, *arr1*, a list of integers representing the first input list, *len2*, an integer representing the length of the second input list and *arr2*, a list of integers representing the second input list, respectively.

Another function/method ***sortArray*** accepts two arguments - *len*, an integer representing the length of the list and *arr*, a list of integers, respectively and return a list sorted ascending order.

Your task is to use the function/method ***sortArray*** to complete the code in ***findMaxElement*** so that it passes all the test cases.

## Scores

### Final Code Submitted

Compilation Status: Fail

```

1 // You can print the values to stdout for debugging
2 class Solution
3 {
4     public int[] sortArray(int len, int[] arr)
5     {
6         int i=0,j=0,temp=0;
7         for(i=0;i<arr.length;i++)
8         {
9             for(j=i+1;j<arr.length;j++)
10            {
11                if(arr[i]<arr[j])
12                {
13                    temp = arr[i];
14                    arr[i] = arr[j];
15                    arr[j] = temp;
16                }
17            }
18        }
19        return arr;
20    }
21
22    public int findMaxElement(int len1, int[] arr1, int len2, int[] arr2)
23    {
24        // write your code here and return maximum element
25        int arr11 = new int[len1];
26        int arr22 = new int[len2];
27        arr11 = sortArray(len1,arr1);
28        arr22 = sortArray(len2,arr2);
29        boolean max = arr11[0]>arr22[0];
30        if (max==true){
31            return arr11[0];
32        }
33        else{
34            return arr22[0];
35        }
36    }
37 }

```

### Code Analysis

#### Average-case Time Complexity

**Candidate code:** Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.

**Best case code:**

\*N represents

#### Errors/Warnings

Solution.java:25: error: incompatible types: int[] cannot be converted to int  
int arr11 = new int[len1];  
^

Solution.java:26: error: incompatible types: int[] cannot be converted to int  
int arr22 = new int[len2];  
^

Solution.java:27: error: incompatible types: int[] cannot be converted to int  
arr11 = sortArray(len1,arr1);  
^

Solution.java:28: error: incompatible types: int[] cannot be converted to int  
arr22 = sortArray(len2,arr2);  
^

Solution.java:29: error: array required, but int found  
boolean max = arr11[0]>arr22[0];  
^

Solution.java:29: error: array required, but int found  
boolean max = arr11[0]>arr22[0];  
^

Solution.java:31: error: array required, but int found  
return arr11[0];  
^

Solution.java:34: error: array required, but int found  
return arr22[0];  
^

8 errors

#### Structural Vulnerabilities and Errors

There are no errors in the candidate's code.



## Compilation Statistics

6

Total attempts

1

Successful

5

Compilation errors

1

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:08:37

Average time taken between two compile attempts:

00:01:26

Average test case pass percentage per compile:

0%

### Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Question 3 (Language: Java)

The function/method ***sumElement*** return an integer representing the sum of the elements in the input array that are greater than twice the input number K and present at the even index. The function/method ***sumElement*** accepts three arguments - *size*, an integer representing the size of the input array, *numK*, an integer representing the input number K and *inputArray*, a list of integers representing the input array.

The function/method ***sumElement*** compiles successfully fails to return the desired result for some test cases. Your task is to debug the code so that it passes all the test cases.

### Note

Index of the input array starts from 0.

## Scores

Final Code Submitted	Compilation Status: Pass	Code Analysis
<pre> 1 class Solution 2 { 3     int sumElement(int size, int numK, int[] inputArray) 4     { 5         int i,sum =0; 6         for(i=0; i&lt;size; i++) 7         { 8             if(inputArray[i]&gt;2*numK &amp;&amp; i % 2 == 0) 9             { 10                sum+= inputArray[i]; 11            } 12        } 13        return sum; 14    } 15 }</pre>		<b>Average-case Time Complexity</b> <p><b>Candidate code:</b> Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.</p> <p><b>Best case code:</b></p> <p>*N represents</p>
		<b>Errors/Warnings</b> <p>There are no errors in the candidate's code.</p>
		<b>Structural Vulnerabilites and Errors</b> <p>There are no errors in the candidate's code.</p>

Test Case Execution	Passed TC: 100%		
Total score <div> </div>	<b>100%</b> Basic(5/5)	<b>100%</b> Advance(3/3)	<b>100%</b> Edge(1/1)

Compilation Statistics					
<div>3</div> Total attempts	<div>2</div> Successful	<div>1</div> Compilation errors	<div>1</div> Sample failed	<div>0</div> Timed out	<div>0</div> Runtime errors
Response time:					00:01:36
Average time taken between two compile attempts:					00:00:32
Average test case pass percentage per compile:					55.6%

### Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Question 4 (Language: Java)

You are given predefined structure **Time** containing *hour*, *minute*, and *second* as members. A collection of functions/methods for performing some common operations on times is also available. You must make use of these functions/methods to calculate and return the difference.

The function/method ***difference\_in\_times*** accepts two arguments - *time1*, and *time2*, representing two times and is supposed to return an integer representing the difference in the number of seconds.

You must complete the code so that it passes all the test cases.

.

### Helper Description

The following class is used to represent a Time and is already implemented in the default code (Do not write this definition again in your code):

```
public class Time
{
    public int hour;
    public int minute;
    public int second;

    public Time(int h, int m, int s)
    {
```

```

hour = h;

minute = m;

second = s;
}

public int compareTo(Object anotherTime)
{
    /*Return 1, if time1 is greater than time2.

    Return -1 if time1 is less than time2

    or, Return 0, if time1 is equal to time2

    This can be called as -

    * If time1 and time2 are two Time then -

    * time1.compareTo(time2) */
}

public void addSecond()
{
    /* Add one second in the time;

    This can be called as -

    * If time1 is Time then -

    * time1.addSecond() */
}

```

## Scores

### Final Code Submitted

### Compilation Status: Fail

```

1 // You can print the values to stdout for debugging
2 class Solution
3 {
4     int difference_in_times(Time time1, Time time2)
5     {
6         // write your code here
7     }
8 }
9 }

```

### Code Analysis

#### Average-case Time Complexity

**Candidate code:** Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.

**Best case code:**

\*N represents

10

### Errors/Warnings

```
Solution.java:8: error: missing return statement
}
^
Time.java:5: warning: [rawtypes] found raw type:
Comparable
public class Time implements Comparable
^
missing type arguments for generic class Comparable
where T is a type-variable:
T extends Object declared in interface Comparable
1 error
1 warning
```

### Structural Vulnerabilities and Errors

There are no errors in the candidate's code.

### Compilation Statistics

0

Total attempts

0

Successful

0

Compilation errors

0

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:00:50

Average time taken between two compile attempts:

00:00:00

Average test case pass percentage per compile:

0%

## Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

## Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

### Question 5 (Language: Java)

The function/method ***deleteNonRepeat*** accepts two arguments - *size*, an integer representing the size of the list and *inputList*, representing the list of integers.

The function/method ***deleteNonRepeat*** removes the non-repeated integers from the *inputList* and prints the remaining repeated integers separated by space from the *inputList*.

If all the elements in the *inputList* are unique then the function/method ***deleteNonRepeat*** should not print anything.

For example, for the given *inputList* {2, 3, 2, 2, 5, 6, 6, 7}, the expected output is 2 2 2 6 6.

The function/method ***deleteNonRepeat*** compiles unsuccessfully due to syntactical error. Your task is to debug the code so that it passes all the test cases.

### Scores

Final Code Submitted	Compilation Status: Fail	Code Analysis
<pre> 1 class Solution 2 { 3     void deleteNonRepeat(int size, int inputList[]) 4     { 5         int count = 0, i,j; 6         int counter[] = new int[size]; 7         for(i=0;i&lt;size;i++) 8         { 9             counter[i]=1; 10        } 11        for(i=0;i&lt;size;i++) </pre>		<b>Average-case Time Complexity</b> <p><b>Candidate code:</b> Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.</p> <p><b>Best case code:</b></p> <p>*N represents</p>
		<b>Errors/Warnings</b>

```

12  {
13      for(j = i+1; j<size; j++)
14      {
15          if(inputList[i]== inputList[j])
16          {
17              counter[i]++;
18              counter[j]++;
19          }
20      }
21  }
22  for(i=0;i<size;i++)
23  {
24      if(counter[i]>1)
25      {
26          count++;
27      }
28  }
29  int newArr[]= new int[count];
30  j=0;
31  for(i=0;i<size;i++)
32  {
33      if(counter[i]>1)
34      {
35          System.out.print(inputList[i]+" ");
36      }
37  }
38  }
39 }

```

Solution.java:35: error: '.class' expected  
 System.out.print(inputList[i]+" ");  
 ^  
 1 error

#### Structural Vulnerabilities and Errors

There are no errors in the candidate's code.

#### Compilation Statistics

1

Total attempts

0

Successful

1

Compilation errors

0

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:01:05

Average time taken between two compile attempts:

00:01:05

Average test case pass percentage per compile:

0%

## Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

## Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

### Question 6 (Language: Java)

The function/method **productMatrix** accepts three arguments - *rows*, an integer representing the rows of the matrix; *columns*, an integer representing the columns of the matrix and *matrix*, a two-dimensional array of integers, respectively.

The function/method **productMatrix** return an integer representing the product of the odd elements whose  $i^{\text{th}}$  and  $j^{\text{th}}$  index are the same. Otherwise, it returns 0.

The function/method **productMatrix** compiles successfully but fails to return the desired result for some test cases. Your task is to debug the code so that it passes all the test cases.

### Scores

Final Code Submitted	Compilation Status: Pass	Code Analysis
<pre> 1 public class Solution 2 { 3     public int productMatrix(int rows, int columns, int matrix[][]) 4     { 5         int result=0; 6         for(int i=0;i&lt;rows;i++) 7             for(int j=0;j&lt;columns;j++) 8                 if((i==j)    (matrix[i][j]%2!=0)) 9                     result *=matrix[i][j]; 10        if(result&lt;=1) 11            return 0; 12        else </pre>		<div>Average-case Time Complexity</div> <p><b>Candidate code:</b> Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.</p> <p><b>Best case code:</b></p> <p>*N represents</p> <div>Errors/Warnings</div>



```

13     return result;
14 }
15 }

```

There are no errors in the candidate's code.

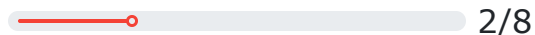
#### Structural Vulnerabilities and Errors

There are no errors in the candidate's code.

### Test Case Execution

Passed TC: 25%

Total score



0%

Basic(0/5)

100%

Advance(2/2)

0%

Edge(0/1)

### Compilation Statistics

1

Total attempts

1

Successful

0

Compilation errors

1

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:00:24

Average time taken between two compile attempts:

00:00:24

Average test case pass percentage per compile:

0%

### *i* Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### *i* Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

### Question 7 (Language: Java)

The function/method ***printFibonacci*** accepts an integer *num*, representing a number.

The function/method ***printFibonacci*** prints first *num* numbers of the Fibonacci series.

For example, given input 5, the function should print the string "0 1 1 2 3" (without quotes).

The function/method compiles successfully but fails to give the desired result for some test cases. Your task is to debug the code so that it passes all the test cases.

## Scores

The candidate did not make any changes in the code.

## Compilation Statistics



Total attempts



Successful



Compilation errors



Sample failed



Timed out



Runtime errors

Response time:

00:00:00

Average time taken between two compile attempts:

00:00:00

Average test case pass percentage per compile:

0%

## Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

## Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Question

The use of messaging apps like WhatsApp/WeChat have bridged the communication gap between people living far apart.

Do you agree or disagree? In your view, how have these apps transformed relationships? Substantiate your response with reasons and suitable examples.

## Scores

Content Score

Grammar Score






 63 / 100

 75 / 100

## Response

I agree to the statement that whatsapp/wechat have bridged the communication gap between people living far apart. It's because of apps like these I think that a husband can happily work onsite which is quite far away from his home without missing his families. Students can study abroad happily and even their parents can also be happy. Through apps like Whatsapp, you can also send images, videos and voice messages to those living far creating an almost virtual you. Many students going for abroad for their pursuit of higher studies rely on apps like these. Many kinds of people who can't communicate that well in front of a person use these softwares to connect them online. These apps have enabled various features and maintain our communication strictly confidential by implementing various features like end-to-end encryption softwares and also by implementing various features to socialize even when you aren't that confident in it. Many use cases include relatives who are living far apart, communicating through these apps and also various organizations also adding to these softwares even for business purposes. These apps have made recent advancements with the help of advancing technological features enabling them to be far more useful and sophisticated. Even strangers from various locations are able to connect through whatsapp and various other social media platforms like facebook, instagram. Parents can experience their child's emotions and various feelings with the help of several features and advancements in these apps. Though these apps have bridged the communication gap, I personally feel that it has made people far lazier. Usually people had to wait and the results were far sweeter than expected and the result of wait was worth it improving the mental health of several families. WhatsApp has made this look like a complete joke and has made people to expect more and made them lazier than ever. Now that communication was made easy, people expect their partners to be available 24/7. To cater to people's expectations companies are introducing new softwares which in turn is making people more lazier than ever. But though we have to bow down to the fact that various communication routes have enabled making people to connect more than ever. People are now worrying less and also have to say missing others less but as they say in-person presence is far greater when you compare with virtual presence. I end my essay by stating that yes I agree to the communication gap bridged

## Error Summary

	Spelling	8
	White Space	12
	Style	0
	Grammar	25
	Typographical	1

## Essay Statistics

406

Total words

18

Total sentences

23

Average sentence  
length

207

Total unique words

156

Total stop words

## Error Details

### Spelling

I agree to the statement that **whatsapp**/wechat have bridged the communication g...

Possible spelling mistake found

I agree to the statement that whatsapp/**wechat** have bridged the communication gap betw...

Possible spelling mistake found

...that well infront of a person use these **softwares** to connect them online. These apps,have...

Possible spelling mistake found

...to connect them online. These apps,have **enabed** various features and maintain our commu...

Possible spelling mistake found

...ous features like end-to-end encryption **softwares** and also by implementing various featur...

Possible spelling mistake found

...ous features to socialize even when you **arent** that confident in it. Many use cases i...

Possible spelling mistake found

...pps and also various organizations also **adaing** to these softwares even for business pu...

Possible spelling mistake found

...s locations are able to connect through **whatsapp** and various other social media platform...

Possible spelling mistake found

...ectations companies are introducing new **softwares** which in turn is making people more laz...

Possible spelling mistake found

...r greater when you compare with virtual **prescence**. I end my essay by stating that yes I a...

Possible spelling mistake found

compare with virtual prescence. I **end** my essay by stating

Possible spelling mistake found. Consider replacing the highlighted text with: 'finish'.

### White Space

...ion gap between people living far apart.**It's** because of apps like these I think th...

Add a space between sentences

...om his home without missing his families,**Students** can study abroad happily and even their...

Put a space after the comma

...nd even their parents can also be happy.**Through** apps like Whatsapp, you can also send i...

Add a space between sentences

... like Whatsapp, you can also send images,**videos** and voice messages to those living far ...

Put a space after the comma

...ce messages to those living far creating an almost virtual you.Many students goin...

Possible typo: you repeated a whitespace

...ing far creating an almost virtual you. **Many** students going for abroad for their pur...

Add a space between sentences

...wares to connect them online. These apps, **have** enabled various features and maintain ou...

Put a space after the comma

...se softwares even for business purposes. **These** apps have made recent advancements with...

Add a space between sentences

...media platforms like facebook, instagram. **Parents** can experience their child's emotions a...

Add a space between sentences

...ect their partners to be available 24/7. **To** cater to people's expectations companies...

Add a space between sentences

...n is making people more lazier than ever. **.** But though we have to bow down to the f...

Don't put a space before the full stop

...making people to connect more than ever. **People** are now worrying less and also have to ...

Add a space between sentences

## Grammar

I agree **to** the statement that whatsapp/wechat have bridged the communication gap between people living far apart.

Possible grammar error found. Consider replacing it with "with".

It's because of apps like these I think that a husband can happily work **onsite** which is quite far away from his home without missing his families, Students can study abroad happily and even their parents can also be happy.

Possible grammar error found. Consider replacing it with "onsite,".

Through apps like Whatsapp, you can also send images, videos and voice messages to those living far **creating** an almost virtual you.

Possible grammar error found. Consider inserting "away," over here.

Many students going **for** abroad for their pursuit of higher studies rely on apps like these.

Possible grammar error found. Consider removing "for" from here.

Many kinds of people who can't communicate that well infringe on a person use **these** softwares to connect them online.

Possible grammar error found. Consider replacing it with "this".

These apps, have enabled various features and **maintain** our communication strictly confidential by implementing various features like end-to-end encryption softwares and also by implementing various features to socialize even when you aren't that confident in it.

Possible grammar error found. Consider replacing it with "keep".

Many **use** cases include relatives who are living far apart, communicating through these apps and also various organizations also adding to these softwares even for business purposes.

Possible grammar error found. Consider replacing it with "used".

These apps have made recent advancements with the help of advancing technological **features** enabling them to be far more useful and sophisticated.

Possible grammar error found. Consider replacing it with "features,".

Parents can experience their **child's** emotions and various feelings with the help of several features and advancements in these apps.

Possible grammar error found. Consider replacing it with "child".

Parents can experience their child's **emotions** and various feelings with the help of several features and advancements in these apps.

Possible grammar error found. Consider inserting "is" over here.

**Though** these apps have bridged the communication gap, I personally feel that it has made people far lazier.

Possible grammar error found. Consider replacing it with "Although".

Though these apps have bridged the communication gap, I personally feel that **it** has made people far lazier.

Possible grammar error found. Consider replacing it with "they".

Though these apps have bridged the communication gap, I personally feel that it **has** made people far lazier.

Possible grammar error found. Consider replacing it with "have".

Usually people had to wait and the results were far **sweet** than expected and the result of wait was worth it improving the mental health of several families.

Possible grammar error found. Consider replacing it with "better".

Usually people had to wait and the results were far sweet than expected and the result of **wait** was worth it improving the mental health of several families.

Possible grammar error found. Consider inserting "the" over here.

Whatsapp has made this look like a complete joke and has made people **to** expect more and made them lazier than ever.

Possible grammar error found. Consider removing "to" from here.

Now that communication **was** made easy, people expect their partners to be available 24/7.

Possible grammar error found. Consider replacing it with "has".

Now that communication was made **easy**, people expect their partners to be available 24/7.

Possible grammar error found. Consider inserting "it" over here.

To cater to people's **expectations** companies are introducing new softwares which in turn is making people more lazier than ever.

Possible grammar error found. Consider inserting "is" over here.

To cater to people's **expectations** companies are introducing new softwares which in turn is making people more lazier than ever.

Possible grammar error found. Consider replacing it with "expectations,".

To cater to people's expectations companies are introducing new softwares which in **turn** is making people more lazier than ever.

Possible grammar error found. Consider replacing it with "turn,".

But though we have to bow down to the fact that various communication routes have enabled **making** people to connect more than ever.

Possible grammar error found. Consider removing "making" from here.

People are now worrying less and also have to say missing others less but as they **say** in - person presence is far greater when you compare with virtual presence.

Possible grammar error found. Consider replacing it with "say,".

People are now worrying less and also have to say missing others less but as they say in - person presence is far greater when you compare **with** virtual presence.

Possible grammar error found. Consider inserting "it" over here.

I end my essay by stating that yes I agree **to** the communication gap bridged

Possible grammar error found. Consider replacing it with "with".

## Typographical

... missing others less but as they say in - person presence  
is far greater when you...

Consider using an m-dash if you do not want to join two  
words.

## 4 | Learning Resources

### Logical Ability

[Take a course on advanced logic](#)



[Play chess and challenge your reasoning skills](#)



[Practice inductive reasoning on visual data](#)



### Quantitative Ability (Advanced)

[Learn about percentages](#)



[Learn about simple and compound interests](#)



[Watch a video on time, speed and distance](#)



### English Comprehension

[Improve your hold on the language by reading Shakespearan plays](#)



[Learn about how to get better at reading](#)



[Read opinions to improve your comprehension](#)



### Icon Index



Free Tutorial



Paid Tutorial



Youtube Video



Web Source



Wikipedia



Text Tutorial



Video Tutorial



Google Playstore