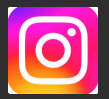
DEBUG WITH SHUBHAM

Accenture Technical interviews Detailed Overview

15-Oct-2024 Coding Interview Questions


https://www.youtube.com/@DebugWithShubham


https://www.linkedin.com/in/debugwithshubham/


https://www.instagram.com/debugwithshubham/


https://topmate.io/debugwithshubham


https://t.me/debugwithshubham

ALL SOLUTION (C++, JAVA, PYTHON) UPLOADED IN GITHUB WITH QUESTION NAME

Debug With Shubham

Made with Goodnotes

Question 1

Revisit Later

How to Attempt?

Rebound Height

Daniel has a ball. He wants to find the ball's rebound height, which he dropped from height **H** with an initial velocity **V**. After the $\mathbf{N^{th}}$ rebound the final velocity of the ball is $\mathbf{V_n}$. Your task is to help him find and return an integer value representing the height to which the ball rebounds after **N** bounces.

Note:

- $\mathbf{H' = H \times e^{2n}}$, where H' is the rebound height, H is the initial height, e is the coefficient of restitution and is the number of bounces.
- $\mathbf{e^n = V/V_n}$, where V is the initial velocity and V_n is the final velocity

Input Specification:

Debug With Shubham

Made with Goodnotes

main.py

```python

def rebound_height(H, V, Vn,N):
    e = V / Vn
    en = e ** (2*N)
    rebound_height = H * en
    return int(rebound_height)
H = 10
V = 15
Vn = 5
N = 3
print(rebound_height(H, V, Vn,N))

```

Main.java

```java
public class ReboundHeightCalculator {
    public static int reboundHeight(double H, double V, double Vn,
        int N) {
        double e = V / Vn;
        double en = Math.pow(e, 2 * N);
        double reboundHeight = H * en;
        return (int) reboundHeight;
    }
    public static void main(String[] args) {
        double H = 10;
        double V = 15;
        double Vn = 5;
        int N = 3;
        System.out.println(reboundHeight(H, V, Vn, N));
    }
}

```

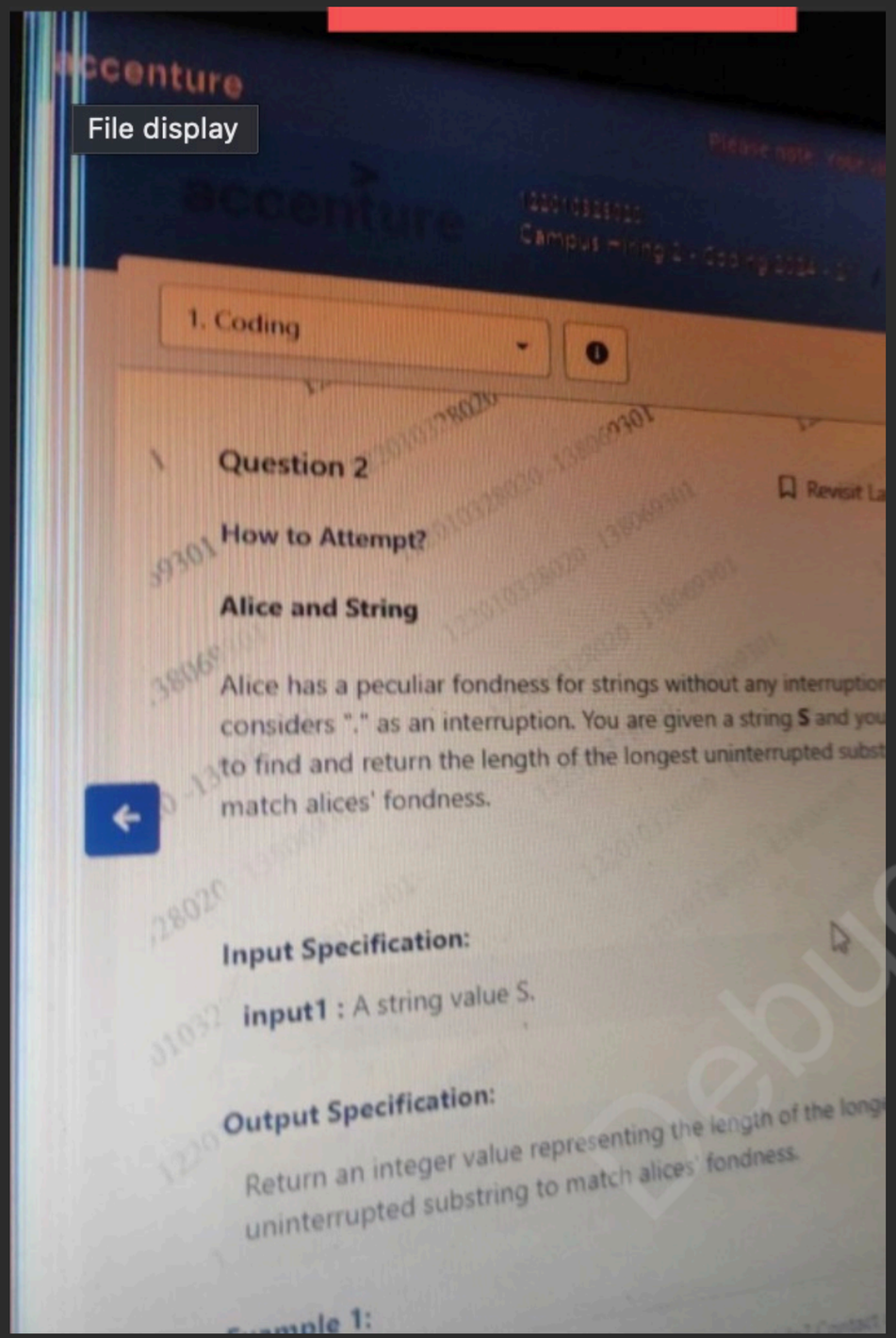
main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;
int reboundHeight(double H, double V, double Vn, int N) {
    double e = V / Vn;
    double en = pow(e, 2 * N);
    double reboundHeight = H * en;
    return (int) reboundHeight;
}
int main() {
    double H = 10;
    double V = 15;
    double Vn = 5;
    int N = 3;
    cout << reboundHeight(H, V, Vn, N) << endl;
    return 0;
}

```

Debrup Shubham

Made with Goodnotes



Debug With Shubham

Made with Goodnotes

With Split

main.py

```python
def longest_uninterrupted_substring(S):
    substrings = S.split('.')
    max_length = 0
    for substring in substrings:
        if len(substring) > max_length:
            max_length = len(substring)
    return max_length

S = "this.is.a.debugwithshubham"
output = longest_uninterrupted_substring(S)
print(output)

```

Main.java

```java
public class LongestUninterruptedSubstring {
    public static int longestUninterruptedSubstring(String S) {

        String[] substrings = S.split("\\.");
        int maxLength = 0;
        for (String substring : substrings) {
            if (substring.length() > maxLength) {
                maxLength = substring.length();
            }
        }
        return maxLength;
    }
    public static void main(String[] args) {
        String S = "this.is.a.debugwithshubham";
        int output = longestUninterruptedSubstring(S);
        System.out.println(output);
    }
}

```

main.cpp

```cpp
#include <iostream>
#include <string>
#include <vector>
#include <sstream>
using namespace std;
vector<string> split(const string &S, char delimiter) {
    vector<string> substrings;
    stringstream ss(S);
    string temp;
    while (getline(ss, temp, delimiter)) {
        substrings.push_back(temp);
    }
    return substrings;
}
int longestUninterruptedSubstring(const string &S) {
    vector<string> substrings = split(S, '.');
    int maxLength = 0;
    for (const string &substring : substrings) {
        if (substring.length() > maxLength) {
            maxLength = substring.length();
        }
    }
    return maxLength;
}
int main() {
    string S = "this.is.a.debugwithshubham";
    int output = longestUninterruptedSubstring(S);
    cout << output << endl;
    return 0;
}

```

Debug With Shubham

Made with Goodnotes

main.py

```python
def longest_uninterrupted_substring(S):
    max_length = 0
    current_length = 0
    for char in S:
        if char == '.':
            max_length = max(max_length, current_length)
            current_length = 0
        else:
            current_length += 1
    max_length = max(max_length, current_length)
    return max_length
S = "this.is.a.debugwithshubham"
output = longest_uninterrupted_substring(S)
print(output)
```

Main.java

Share

```java
public class LongestUninterruptedSubstring {
    public static int longestUninterruptedSubstring(String S) {
        int maxLength = 0;
        int currentLength = 0;
        for (int i = 0; i < S.length(); i++) {
            if (S.charAt(i) == '.') {
                maxLength = Math.max(maxLength, currentLength);
                currentLength = 0;
            } else {
                currentLength++;
            }
        }
        maxLength = Math.max(maxLength, currentLength);

        return maxLength;
    }
    public static void main(String[] args) {
        String S = "this.is.a.debugwithshubham";
        int output = longestUninterruptedSubstring(S);
        System.out.println(output);
    }
}
```

main.cpp

```cpp
#include <iostream>
#include <string>
using namespace std;
int longestUninterruptedSubstring(const string &S) {
    int maxLength = 0;
    int currentLength = 0;
    for (char c : S) {
        if (c == '.') {
            maxLength = max(maxLength, currentLength);
            currentLength = 0;
        } else {
            currentLength++;
        }
    }
    maxLength = max(maxLength, currentLength);
    return maxLength;
}
int main() {
    string S = "this.is.a.debugwithshubham";
    int output = longestUninterruptedSubstring(S);
    cout << output << endl;
    return 0;
}
```

Debug with Shubham

Made with Goodnotes