Largest Subarray With Contiguous Elements in C++

```
#include <iostream>
#include <unordered_set>
#include <vector>

using namespace std;

int solution(vector<int>& arr) {
    int ans = 0;

    for (int i = 0; i < arr.size() - 1; i++) {
        int min_val = arr[i];
        int max_val = arr[i];
        unordered_set<int> contiguous_set;
        contiguous_set.insert(arr[i]);

        for (int j = i + 1; j < arr.size(); j++) {
            if (contiguous_set.find(arr[j]) != contiguous_set.end()) {
                break; // If duplicate found, break the loop
            }

            contiguous_set.insert(arr[j]);
            min_val = min(min_val, arr[j]);
            max_val = max(max_val, arr[j]);

            if (max_val - min_val == j - i) {
                int len = j - i + 1;
                if (len > ans) {
                    ans = len;
                }
            }
        }
    }

    return ans;
}

int main() {
    vector<int> arr = {10, 12, 11};
    cout << solution(arr) << endl; // Output: 3

    return 0;
}
```

Understanding the Problem

- The function solution(arr) finds the length of the **longest contiguous subarray** where all elements are **distinct and consecutive**.
- A contiguous subarray is valid if:

  max_val - min_val = j - i

- **Example Input:** {10, 12, 11}
- **Expected Output:** 3 (as {10, 12, 11} forms a valid contiguous subarray)

Step-by-Step Dry Run

Outer Loop (i)	Inner Loop (j)	Subarray	min_val	max_val	max_val - min_val	j - i	Valid?	Current ans
0	0	{10}	10	10	0	0	✓	1
0	1	{10, 12}	10	12	2	1	✗	1
0	2	{10, 12, 11}	10	12	2	2	✓	**3**
1	1	{12}	12	12	0	0	✓	3
1	2	{12, 11}	11	12	1	1	✓	3
2	2	{11}	11	11	0	0	✓	3

Final Output: 3

Output:

```
3
```