Largest Perimeter triangle in C++

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int largestPerimeter(vector<int>& nums)
{
   sort(nums.begin(), nums.end());

   int p = 0;

   for (int i = nums.size() - 1; i >= 2; --i) {
      if (nums[i - 1] + nums[i - 2] >
nums[i]) {
         p = nums[i - 1] + nums[i - 2] +
nums[i];
         break;
      }
   }

   return p;
}

int main() {
   vector<int> nums = {25, 6, 9, 11, 8, 12,
10, 3, 2};

   cout << largestPerimeter(nums) <<
endl;

   return 0;
}
```

Step-by-step check after sorting:

nums = {2, 3, 6, 8, 9, 10, 11, 12, 25}

We're looping from the end (i = 8) down to 2, checking this:

if (nums[i-1] + nums[i-2] > nums[i]) // triangle inequality

🧠 **Dry Run Table with Full Checks:**

i	nums[i-2]	nums[i-1]	nums[i]	Sum of two smallest	Valid triangle?	Perimeter
8	11	12	25	11 + 12 = 23	❌ (23 < 25)	-
7	10	11	12	10 + 11 = 21	✅	33

So, yes — the **first valid triangle** found is {10, 11, 12}, with perimeter = 33.

🛠 **Why not {11, 12, 25}?**

Because 11 + 12 = 23, which is **less than 25** — **fails triangle condition**.

✅ **Correct Output:**

33

33