Best time to buy and sell stocks in C++

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

using namespace std;

class BestTimeToBuyAndSellStock {
public:
    int maxProfit(vector<int>& prices) {
        if (prices.empty()) return 0;

        int maxP = 0;
        int minBP = prices[0];

        for (int prc : prices) {
            int tp = prc - minBP;
            if (tp > maxP) {
                maxP = tp;
            }
            minBP = min(minBP, prc);
        }

        return maxP;
    }
};

int main() {
    BestTimeToBuyAndSellStock solution;

    // Test case 1
    vector<int> prices1 = {7, 1, 5, 3, 6, 4};
    int maxProfit1 = solution.maxProfit(prices1);
    cout << "Max profit for prices1: " << maxProfit1 <<
endl; // Output: 5

    return 0;
}
```

Let's walk through a **dry run in tabular form** of your code for:

vector<int> prices1 = {7, 1, 5, 3, 6, 4};

🧠 **Variables:**

- minBP = Minimum Buying Price seen so far.
- tp = Temporary Profit (current price - minBP).
- maxP = Maximum Profit observed.

🔍 **Dry Run Table:**

Day (Index)	Price	minBP (min so far)	tp = price - minBP	maxP (max profit so far)
0	7	7	0	0
1	1	1	0	0
2	5	1	4	4
3	3	1	2	4
4	6	1	5	5 ✅
5	4	1	3	5

✅ **Final Answer:**

Max profit for prices1: 5

Output:-
maxP = 5 (Maximum profit)