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;
}
```

Input:

- prices = {7, 1, 5, 3, 6, 4}

Initialization:

- maxP = 0 (maximum profit so far)
- minBP = prices[0] = 7 (minimum buying price)

Iteration:

1. **Day 1** (prc = 7):
    - tp = prc - minBP = 7 - 7 = 0
    - maxP = max(maxP, tp) = max(0, 0) = 0
    - minBP = min(minBP, prc) = min(7, 7) = 7
2. **Day 2** (prc = 1):
    - tp = prc - minBP = 1 - 7 = -6
    - maxP = max(maxP, tp) = max(0, -6) = 0
    - minBP = min(minBP, prc) = min(7, 1) = 1
3. **Day 3** (prc = 5):
    - tp = prc - minBP = 5 - 1 = 4
    - maxP = max(maxP, tp) = max(0, 4) = 4
    - minBP = min(minBP, prc) = min(1, 5) = 1
4. **Day 4** (prc = 3):
    - tp = prc - minBP = 3 - 1 = 2
    - maxP = max(maxP, tp) = max(4, 2) = 4
    - minBP = min(minBP, prc) = min(1, 3) = 1
5. **Day 5** (prc = 6):
    - tp = prc - minBP = 6 - 1 = 5
    - maxP = max(maxP, tp) = max(4, 5) = 5
    - minBP = min(minBP, prc) = min(1, 6) = 1
6. **Day 6** (prc = 4):
    - tp = prc - minBP = 4 - 1 = 3
    - maxP = max(maxP, tp) = max(5, 3) = 5
    - minBP = min(minBP, prc) = min(1, 4) = 1

Output:-

maxP = 5 (Maximum profit)