SW Expert Academy 모의 테스트

박용성

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

13454. [2022 동계 대학생 양성과정] 시간여행자의 주식

문제 내용

시간 : 25개 테스트케이스를 합쳐서 C++의 경우 3초 / Java의 경우 3초

메모리 : 힙, 정적 메모리 합쳐서 256MB 이내, 스택 메모리 1MB 이내

※ SW expert 아카데미의 문제를 무단 복제하는 것을 금지합니다.

- 코드배틀 비밀번호 및 게시된 문제의 타인 공유 / 무단 복제는 엄격히 금지됩니다.
- 지원언어 : C++ / JAVA
*** STL 라이브러리 사용 가능**

① **C 또는 C++로 답안을 작성하시는 응시자께서는 검정시스템에 제출 시, Language 에서 C++ 를 선택하신 후**

② **Main 과 User Code 부분으로 구성되어 있습니다.**

A. Main : 수정할 수 없는 코드이며, 채점 시 비 정상적인 답안 검출 등 평가를 위한 로직이 추가 될 수

B. User Code : 실제 응시자가 작성해야 하는 코드이며, 제출 시에는 표준 입출력 함수가 포함되어 있으면

③ **Local PC 에서 프로그래밍 시 유의 사항**

A. 2개의 파일을 생성하셔야 합니다. (main.cpp / solution.cpp 또는 Solution.java / UserSolution.java)

B. Main 부분의 코드를 main.cpp 또는 Solution.java 에 복사해서 사용하시기 바랍니다.

C. sample_input.txt 를 사용하시기 위해서는 Main 부분의 코드 내에

표준 입력을 파일로 전환하는 코드 (주석처리 되어 있음) 의 주석을 풀어서 사용하시면 됩니다.

D. User Code 부분의 코드를 작성하신 후 서버에 제출하실 때,

디버깅을 위한 표준 입출력 함수를 모두 삭제 또는 주석 처리해 주셔야 합니다.

④ **문제 내에 제약조건을 모두 명시하지 않으므로 주어지는 코드를 분석하셔야 합니다.**

⑤ **코드는 개발 언어에 따라 상이할 수 있으므로, 작성할 언어를 기준으로 분석하셔야 합니다.**

[문제 설명]

시간 여행자의 의뢰를 받아, 주식 시장 프로그램을 작성하고자 한다.

프로그램은 매수 주문과 매도 주문을 처리할 수 있어야 한다.

매수 주문은 주식을 사기 위한 주문, 매도 주문은 주식을 팔기 위한 주문을 의미한다.

주문에는 주문 번호, 주식 종목, 주문 수량, 희망 가격이 포함된다.

주문은 주문 번호로 구별된다.

매수 주문의 희망 가격이 매도 주문의 희망 가격 이상일 경우에만, 두 주문의 거래가 체결될 수 있다.

매수 주문이 새로 들어오면 미체결 매도 주문들과 거래를 체결한다.

거래를 체결할 미체결 매도 주문은 매도 희망 가격이 낮은 주문, 가격이 동일하다면 주문 번호가 낮은 주문 순으로 선택한다.

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

[Table 1]과 같이 미체결 주문들이 있는 상황을 생각해보자.

주식 종목 1					
미체결 매수 주문			미체결 매도 주문		
주문 번호	개수	희망 가격	주문 번호	개수	희망 가격
2	5	80	1	10	100
5	10	85	3	5	90
			4	5	100

[Table 1]

이제 주문 번호는 6, 주식 종목은 1, 주문 수량은 13, 희망 가격은 95인 매수 주문이 들어온 경우를 생각해보자.

미체결 매도 주문 중에서 희망 가격이 95 이하인 주문은 3번 주문 밖에 없다.

이에 따라 3번 주문과 6번 주문의 거래가 체결되며, 체결되는 주문 수량은 5개, 체결 가격은 90이다.

6번 주문의 남은 주문 수량 8개는 미체결 주문에 남는다.

6번 주문을 처리한 후, 미체결 주문들의 상황은 [Table 2]와 같다.

주식 종목 1					
미체결 매수 주문			미체결 매도 주문		
주문 번호	개수	희망 가격	주문 번호	개수	희망 가격
2	5	80	1	10	100
5	10	85	4	5	100
6	8	95			

[Table 2]

매도 주문이 새로 들어오면 미체결 매수 주문들과 거래를 체결한다.

거래를 체결할 미체결 매수 주문은 매수 희망 가격이 높은 주문, 가격이 동일하다면 주문 번호가 낮은 주문 순으로 선택한다.

이때 체결되는 주문 수량은 두 주문의 남은 주문 수량 중 최솟값이며, 체결 가격은 매수 주문의 희망 가격이다.

체결되지 못한 주문 수량은 미체결 주문에 남는다.

[Table 2] 상태에서 주문 번호는 7, 주식 종목은 1, 주문 수량은 13, 희망 가격은 85인 매도 주문이 들어올 경우를 생각해보자.

미체결 매수 주문 중에서 희망 가격이 85 이상인 주문에는 5번 주문과 6번 주문이 있다.

이 중 매수 희망 가격이 더 높은 6번 주문과의 거래가 먼저 체결된다.

체결되는 주문 수량은 8개, 체결 가격은 95이다.

그 후 5번 주문과의 거래가 체결된다.

체결되는 주문 수량은 5개, 체결 가격은 85이다.

7번 주문은 모두 체결되었으므로 미체결 주문에 남지 않는다.

7번 주문을 처리한 후, 미체결 주문들의 상황은 [Table 3]과 같다.

주식 종목 1					
미체결 매수 주문			미체결 매도 주문		
주문 번호	개수	희망 가격	주문 번호	개수	희망 가격
2	5	80	1	10	100
5	5	85	4	5	100

[Table 3]

프로그램은 미체결 주문을 취소할 수 있어야 한다.

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

주식 종목 1

미체결 매수 주문			미체결 매도 주문		
주문 번호	개수	희망 가격	주문 번호	개수	희망 가격
5	5	85	1	10	100
			4	5	100

[Table 4]

의뢰인인 시간 여행자는 과거로 시간여행을 갈 수 있다.

그는 부자가 되기 위해, mStock을 입력하면 아래의 값을 반환하는 함수를 요구했다.

(mStock 주식의 임의의 시점(B)에서의 체결 가격 – mStock 주식의 임의의 시점(A)에서의 체결 가격)의 최댓값

단, 시점(B)는 함수 호출 시점 이전이어야 하며, 시점(A)는 시점(B) 이전이어야 한다.

예를 들어 mStock 주식의 거래 체결 가격을 시간순으로 나열한 것이 아래와 같을 경우, 20 – 7, 즉, 13을 반환해야 한다.

체결 가격	21	25	9	7	20	6

[Table 5]

아래 API 설명을 참조하여 각 함수를 구현하라.

※ 아래 함수 signature는 C/C++에 대한 것으로 Java에 대해서는 제공되는 Solution.java와 UserSolution.java를 참고하라.

아래는 User Code 부분에 작성해야 하는 API 의 설명이다.

void init()
각 테스트 케이스의 처음에 호출된다. 테스트 케이스의 시작 시 미체결 주문은 없다.
int buy(int mNumber, int mStock, int mQuantity, int mPrice)

SW Expert Academy 모의 테스트

박용성

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

첫 주문 시 mNumber는 1이고, 이후 buy 혹은 sell 함수가 호출될 때마다 1씩 증가한다.

mNumber 주문과 미체결 매도 주문들 사이의 거래를 체결한 후, 남은 주문 수량을 미체결 매수 주문에 남긴다.

두 주문의 거래가 체결되기 위해선 매수 주문의 희망 가격이 매도 주문의 희망 가격 이상이어야 한다

거래를 체결할 미체결 매도 주문은 매도 희망 가격이 낮은 주문, 가격이 동일하다면 주문 번호가 낮은 주문 순으로 선택한다.

먼저 선택된 주문과의 거래가 나중에 선택된 주문과의 거래보다 이른 시점에 체결된다.

거래 시 체결되는 주문 수량은 두 주문의 남은 주문 수량 중 최솟값이며, 체결 가격은 매도 주문의 희망 가격이다.

미체결 매수 주문에 남은 mNumber 주문의 주문 수량을 반환한다.

mNumber 주문이 미체결 매수 주문에 남지 않는 경우, 0을 반환한다.

Parameters

mNumber : 주문 번호 (1 ≤ mNumber ≤ 200,000)

mStock : 주식 종목 (1 ≤ mStock ≤ 5)

mQuantity : 주문 수량 (1 ≤ mQuantity ≤ 1,000,000)

mPrice : 매수 희망 가격 (1 ≤ mPrice ≤ 1,000,000)

Returns

미체결 매수 주문에 남은 mNumber 주문의 주문 수량

int sell(int mNumber, int mStock, int mQuantity, int mPrice)

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

첫 주문 시 mNumber는 1이고, 이후 buy 혹은 sell 함수가 호출될 때마다 1씩 증가한다.

mNumber 주문과 미체결 매수 주문들 사이의 거래를 체결한 후, 남은 주문 수량을 미체결 매도 주문에 남긴다.

두 주문의 거래가 체결되기 위해선 매수 주문의 희망 가격이 매도 주문의 희망 가격 이상이어야 한다

거래를 체결할 미체결 매수 주문은 매수 희망 가격이 높은 주문, 가격이 동일하다면 주문 번호가 낮은 주문 순으로 선택한다.

먼저 선택된 주문과의 거래가 나중에 선택된 주문과의 거래보다 이른 시점에 체결된다.

거래 시 체결되는 주문 수량은 두 주문의 남은 주문 수량 중 최솟값이며, 체결 가격은 매수 주문의 희망 가격이다.

미체결 매도 주문에 남은 mNumber 주문의 주문 수량을 반환한다.

mNumber 주문이 미체결 매도 주문에 남지 않는 경우, 0을 반환한다.

Parameters

mNumber : 주문 번호 (1 ≤ mNumber ≤ 200,000)

mStock : 주식 종목 (1 ≤ mStock ≤ 5)

mQuantity : 주문 수량 (1 ≤ mQuantity ≤ 1,000,000)

mPrice : 매도 희망 가격 (1 ≤ mPrice ≤ 1,000,000)

Returns

미체결 매도 주문에 남은 mNumber 주문의 주문 수량

void cancel(int mNumber)

미체결 주문 중 mNumber 주문을 취소한다.

Parameters

mNumber : 주문 번호 (1 ≤ mNumber ≤ 200,000)

int bestProfit(int mStock)

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

단, 시점(B)는 bestProfit 함수 호출 시점 이전이어야 하며, 시점(A)는 시점(B) 이전이어야 한다.

'시점(B) 이전'은 '시점(B)'를 포함한다.

buy 혹은 sell 함수가 한 번만 호출되었을 때에도, 여러 번의 거래 체결이 생길 수 있음에 유의하라.

이 경우, 어떤 거래 체결이 더 이른 시점에 발생하는 지는 각 함수의 설명을 참조하라.

mStock 주식의 거래 체결이 한 번 이상 있었음이 보장된다.

Parameters

mStock : 주식 종목 (1 ≤ mStock ≤ 5)

Returns

(mStock 주식의 임의의 시점(B)에서의 체결 가격 – mStock 주식의 임의의 시점(A)에서의 체결 가격) 의 최댓값

[예제]

아래 표에서 [a, b]는 매수 주문인 a 주문과 매도 주문인 b 주문 사이의 거래를 의미한다.

Order	Function	Description	Return
1	init()		
2	buy(1, 1, 5, 105)		5
3	buy(2, 1, 5, 100)		5
4	sell(3, 1, 12, 100)	[1, 3] : 가격 105에 5개 체결 [2, 3] : 가격 100에 5개 체결	2
5	bestProfit(1)	체결 가격 : 105, 100	0
6	sell(4, 1, 8, 90)		8
7	sell(5, 1, 1, 110)		1
8	buy(6, 1, 11, 110)	[6, 4] : 가격 90에 8개 체결 [6, 3] : 가격 100에 2개 체결 [6, 5] : 가격 110에 1개 체결	0
9	bestProfit(1)	체결 가격 : 105, 100, 90, 100, 110	20
10	buy(7, 1, 1, 80)		1
11	buy(8, 1, 1, 85)		1
12	sell(9, 1, 3, 70)	[8, 9] : 가격 85에 1개 체결 [7, 9] : 가격 80에 1개 체결	1
13	cancel(9)		
14	buy(10, 1, 1, 70)		1
15	bestProfit(1)	체결 가격 : 105, 100, 90, 100, 110, 85, 80	20
16	sell(11, 5, 500000, 1000000)		500000
17	sell(12, 5, 499999, 999999)		499999
18	buy(13, 5, 1000000, 1000000)	[13, 12] : 가격 999999에 499999개 체결 [13, 11] : 가격 1000000에 500000개 체결	1

SW Expert Academy 모의 테스트 박용성

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

[제약사항]

1. 각 테스트 케이스 시작 시 init() 함수가 호출된다.

2. 각 테스트 케이스에서 buy() 함수의 호출 횟수는 100,000 이하이다.

3. 각 테스트 케이스에서 sell() 함수의 호출 횟수는 100,000 이하이다.

4. 각 테스트 케이스에서 cancel() 함수의 호출 횟수는 200,000 이하이다.

6. 각 테스트 케이스에서 bestProfit() 함수의 호출 횟수는 100,000 이하이다.

[입출력]

입출력은 제공되는 Main 부분의 코드에서 처리하므로 User Code 부분의 코드에서는 별도로 입출력을 처리하지 않는다.

Sample input 에 대한 정답 출력 결과는 "#TC번호 결과" 의 포맷으로 보여지며 결과가 100 일 경우 정답, 0 일 경우 오답을 의미한다.

입력

```
5 100
19
1
2 1 1 5 105 5
2 2 1 5 100 5
3 3 1 12 100 2
5 1 0
3 4 1 8 90 8
...

// 첨부파일 sample_input.txt 참조
```

sample_input.txt

출력

```
#1 100
#2 100
#3 100
#4 100
#5 100
```

sample_output.txt

문제 풀이

Language C++14 (gcc-10.3)

Main

```
#ifndef _CRT_SECURE_NO_WARNINGS
#define _CRT_SECURE_NO_WARNINGS
#endif

#include <stdio.h>

#define CMD_INIT 1
#define CMD_BUY 2
#define CMD_SELL 3
#define CMD_CANCEL 4
#define CMD_BEST_PROFIT 5

extern void init();
extern int buy(int mNumber, int mStock, int mQuantity, int mPrice);
extern int sell(int mNumber, int mStock, int mQuantity, int mPrice);
```

SW Expert Academy 모의 테스트 박용성

제출결과

제출횟수 1 / 10

제출회차	제출시간	결과
1차	18:43:20	Fail (제한시간 초과) 코드보기

```
#include <iostream>
#include <deque>
#include <algorithm>

using namespace std;

typedef struct order {
    int mNumber, mQuantity, mPrice;
} order;

struct Find {
    int num;
    bool operator() (order temp) {
        return (temp.mNumber == num);
    }

```

소스 코드 초기화

TEST

Input 을 입력하고 Run을 선택하면 Output 결과를 확인할 수 있습니다. Input 값을 넣지 않으면 기본 Input값이 적용되어 실행됩니다. Test는 채점을 하는 것이 아니며 정답 여부를 알려주지 않습니다.

Input

Input값을 입력해 주세요.

Output

Run Clear

임시저장 컴파일 제출