**[제한 사항]**

|  |  |
| --- | --- |
| 시간 | 25개 테스트케이스를 합쳐서 C++ 의 경우 3초 / Java 의 경우 3초 |
| 메모리 | 힙, 정적 메모리 합쳐서 256MB 이내, 스택 메모리 1MB 이내 |

**※** 본 문제는 임직원의 S/W 문제해결 역량 향상을 위한 **Professional 실전형 연습문제**이며

함께 제공되는 **Testcase 또한 학습을 위한 연습용**입니다.

|  |
| --- |
| **①   C 또는 C++로 답안을 작성하시는 응시자께서는 검정시스템에 제출 시, Language 에서 C++ 를 선택하신 후 제출하시기 바랍니다.**  **②   Main 과 User Code 부분으로 구성되어 있습니다.**  **ㅇ   A.**Main         : 수정할 수 없는 코드이며, 채점 시 비 정상적인 답안 검출 등 평가를 위한 로직이 추가 될 수 있습니다.  **ㅇ   B.**User Code  : 실제 응시자가 작성해야 하는 코드이며, 제출 시에는 표준 입출력 함수가 포함되어 있으면 안 됩니다.  **③   Local PC 에서 프로그래밍 시 유의 사항**  **ㅇ   A.**2개의 파일을 생성하셔야 합니다. ( main.cpp / solution.cpp 또는 Solution.java / UserSolution.java )  **ㅇ   B.**Main 부분의 코드를 main.cpp 또는 Solution.java 에 복사해서 사용하시기 바랍니다.  **ㅇ   C.**sample\_input.txt 를 사용하시기 위해서는 Main 부분의 코드 내에  **ㅇ   C.**표준 입력을 파일로 전환하는 코드 ( 주석처리 되어 있음 ) 의 주석을 풀어서 사용하시면 됩니다.  **ㅇ   D.**User Code 부분의 코드를 작성하신 후 서버에 제출하실 때,  **ㅇ   D.**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]**

N개의 공항이 있다.

각 공항은 0 이상 N-1 이하의 고유 번호를 가진다.

하루는 24시간이다.

시각의 기준은 모두 통일되어 있으므로, 공항 간의 시차는 고려하지 않아도 된다.

모든 항공 노선은 **하루에 한 번** 정해진 시각에 운행하는 정기 노선이다.

쉬는 날은 없으며, 취소되는 날도 없다.

항공 노선에 대한 정보로 출발 공항 번호, 도착 공항 번호, 출발 시각, 비행 소요 시간, 티켓 가격이 주어진다.

출발 시각이 빨라지거나 지연되는 경우는 없다. 비행 소요 시간이 줄어들거나 늘어나는 경우도 없다.

따라서 도착 시각은 출발 시각과 비행 소요 시간을 통해 정확히 알아낼 수 있다.

출발 시각과 도착 시각은 항상 정각이다.

[Fig. 1]은 항공 노선의 한 예시이다.

이 항공 노선의 출발 공항 번호는 0, 도착 공항 번호는 1이다.

또한 출발 시각은 23시, 비행 소요 시간은 3시간이다.

하루는 24시간이므로, 도착 시각은 다음날 2시임을 알아낼 수 있다.

즉, 이 항공 노선은 매일 23시에 0번 공항에서 출발하여, 다음 날 2시에 1번 공항에 도착하는 정기 노선이다.

티켓 가격은 100 달러이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 1]**

**환승에는 시간이 소요되지 않는다.**

어떤 공항에 2시에 도착하면, 그 즉시, 그 공항에서 2시에 출발하는 항공 노선을 이용할 수 있다.

그러나 2시에 도착한 후, 3시에 출발하는 항공 노선을 이용하고자 할 때는 1시간을 대기해야 한다.

[Fig. 2]는 4개의 항공 노선이 있는 경우의 예시이다.

0번 공항에서 20시에 출발하여, 최대한 빠르게 2번 공항으로 갈 경우, 총 19시간이 소요된다. 그 방법은 아래와 같다.

(0번 공항, 20시) → 3시간 대기 → (0번 공항, 23시) → 3시간 이동 → (1번 공항, 2시) → 3시간 대기 → (1번 공항, 5시) → 10시간 이동 → (2번 공항, 15시)

0번 공항에서 2번 공항으로, 최대한 적은 비용으로 이동할 경우, 총 255 달러가 소모된다. 그 방법은 아래와 같다.

(0번 공항) → 123달러 → (3번 공항) → 132달러 → (2번 공항)

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 2]**

아래 API 설명을 참조하여 각 함수를 구현하라.

※ 아래 함수 signature는 C/C++에 대한 것으로 다른 언어에 대해서는 제공되는 Main과 User Code를 참고하라.

아래는 User Code 부분에 작성해야 하는 API 의 설명이다.

|  |
| --- |
| **void init(int N)** |
| 각 테스트 케이스의 처음에 호출된다.    공항은 N개이다.    테스트 케이스의 시작 시, 존재하는 항공 노선은 없다.    ***Parameters***     N : 공항의 수 (3 ≤ N ≤ 60) |
| **void add(int mStartAirport, int mEndAirport, int mStartTime, int mTravelTime, int mPrice)** |
| mStartAirport번 공항에서 mEndAirport번 공항으로 가는 항공 노선을 추가한다.    출발 시각은 mStartTime시이다.  비행 소요 시간은 mTravelTime 시간이다.  티켓 가격은 mPrice 달러이다.    두 공항 사이에 여러 개의 항공 노선이 존재할 수 있음에 유의하라.    mStartAirport와 mEndAirport는 서로 다름이 보장된다.    ***Parameters***     mStartAirport : 출발 공항 번호 (0 ≤ mStartAirport ≤ N-1)     mEndAirport : 도착 공항 번호 (0 ≤ mEndAirport ≤ N-1)     mStartTime : 출발 시각 (0 ≤ mStartTime ≤ 23)     mTravelTime : 비행 소요 시간 (1 ≤ mTravelTime ≤ 23)     mPrice : 티켓 가격 (10 ≤ mPrice ≤ 5,000) |
| **int minTravelTime(int mStartAirport, int mEndAirport, int mStartTime)** |
| mStartAirport번 공항에서 mStartTime시에 출발하여, mEndAirport번 공항으로 소요 시간을 최대한 적게 하면서 이동할 때의 소요 시간을 반환한다.    mStartAirport번 공항에서 mStartTime시에 출발하여, mEndAirport번 공항으로 갈 수 없으면, -1을 반환한다.    mStartAirport와 mEndAirport는 서로 다름이 보장된다.    ***Parameters***     mStartAirport : 출발 공항 번호 (0 ≤ mStartAirport ≤ N-1)     mEndAirport : 도착 공항 번호 (0 ≤ mEndAirport ≤ N-1)     mStartTime : 출발 시각 (0 ≤ mStartTime ≤ 23)    ***Return***     최소 이동 시간 |
| **int minPrice(int mStartAirport, int mEndAirport)** |
| mStartAirport번 공항에서 mEndAirport번 공항으로 비용을 최대한 적게 하면서 이동할 때의 비용을 달러 단위로 반환한다.  비용은 이용한 항공 노선들의 티켓 가격의 총합이다.    mStartAirport번 공항에서 mEndAirport번 공항으로 갈 수 없으면, -1을 반환한다.    mStartAirport와 mEndAirport는 서로 다름이 보장된다.    ***Parameters***     mStartAirport : 출발 공항 번호 (0 ≤ mStartAirport ≤ N-1)     mEndAirport : 도착 공항 번호 (0 ≤ mEndAirport ≤ N-1)    ***Return***     최소 비용 |

**[예제]**

Note 부분에서, (a, b)는 (a번 공항, b시)을 의미한다. 즉, (0, 23)는 (0번 공항, 23시)를 의미한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order** | **Function** | **Return** | **Note** | **Figure** |
| 1 | init(6) |  |  |  |
| 2 | add(0, 1, 23, 3, 100) |  |  | [Fig. 1] |
| 3 | minTravelTime(0, 1, 23) | 3 | (0, 23)→(1, 2) |  |
| 4 | minPrice(0, 1) | 100 | 0→1 |  |
| 5 | minTravelTime(0, 2, 0) | -1 |  |  |
| 6 | minPrice(1, 0) | -1 |  |  |

**[Table 1]**

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 1]**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order** | **Function** | **Return** | **Note** | **Figure** |
| 7 | add(0, 3, 7, 5, 123) |  |  |  |
| 8 | add(3, 2, 12, 5, 132) |  |  |  |
| 9 | add(1, 2, 5, 10, 200) |  |  | [Fig. 2] |
| 10 | minTravelTime(0, 2, 20) | 19 | (0, 20)→(0, 23)→(1, 2)→(1, 5)→(2, 15) |  |
| 11 | minPrice(0, 2) | 255 | 0→3→2 |  |

**[Table 2]**

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 2]**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order** | **Function** | **Return** | **Note** | **Figure** |
| 12 | add(3, 4, 17, 23, 30) |  |  |  |
| 13 | add(2, 5, 23, 1, 20) |  |  |  |
| 14 | add(4, 5, 12, 12, 500) |  |  |  |
| 15 | add(5, 4, 0, 1, 15) |  |  | [Fig. 3] |
| 16 | minTravelTime(3, 4, 12) | 13 | (3, 12)→(2, 17)→(2, 23)→(5, 0)→(4, 1) |  |
| 17 | minPrice(3, 4) | 30 | 3→4 |  |
| 18 | minPrice(0, 5) | 275 | 0→3→2→5 |  |
| 19 | minTravelTime(5, 4, 1) | 24 | (5, 1)→(5, 0)→(4, 1) |  |
| 20 | minTravelTime(1, 4, 20) | 29 | (1, 20)→(1, 5)→(2, 15)  →(2, 23)→(5, 0)→(4, 1) |  |

**[Table 3]**

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 3]**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order** | **Function** | **Return** | **Note** | **Figure** |
| 21 | add(0, 1, 23, 3, 50) |  |  | [Fig. 4] |
| 22 | minPrice(0, 1) | 50 | 0→1 |  |

**[Table 4]**

텍스트이(가) 표시된 사진

자동 생성된 설명

**[Fig. 4]**

**[제약사항]**

1. 각 테스트 케이스 시작 시 init() 함수가 호출된다.

2. 각 테스트 케이스에서 add() 함수의 호출 횟수는 30,000 이하이다.

3. 각 테스트 케이스에서 minTravelTime() 함수의 호출 횟수는 1,000 이하이다.

4. 각 테스트 케이스에서 minPrice() 함수의 호출 횟수는 5,000 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.