CSE 2002 자료구조와 실습

설계문제 #1: 리스트를 이용한 항공권 예약 시스템

단원	리스트	난이도	초급 /중급 /고급
참여인원	1명 /4주 (man/month)	마감	2014년 4월 30일(수) 자정

1. 과제 개요

(1) 정의

프로젝트 문제는 한마디로 비행기 예약 시스템을 C 언어로 프로그래밍 하는 것이다. 예약 시스템은 항공사의 모든 항공편과 각 편마다 예약한 고객을 보관하고 있어야 한다. 또한 각 항공편마다 완전히 매진된 후 대기 중인 리스트를 보관하고 있어야 한다. 고객은 여러 항공편에 동시에 예약할 수도 있고 대기 리스트에 올려놓고 다른 항공편에 예약할 수도 있다.

(2) 필요성

강의 시간에 단순연결 리스트와 이중 연결 리스트에 대한 기본 개념을 설명하였다. 또한 실습 시간에 두 가지 자료구조에 대한 기본 오퍼레이션을 프로그래밍 하였다. 그러나 이러한 기본 자료구조가 실제 문제에서 어떻게 응용되며 Open-ended problem에서 어떻게 적용될 수 있는지 생각해보고 적용하는 능력이 중요하다. 이번 설계 문제에서는 리스트의 기본 오퍼레이션들이 실제 문제에서 어떻게 사용되는지, 그리고 이러한 기본 오퍼레이션을 빌딩 블록으로 하여 더욱 복잡한 애플리케이션을 설계하는 방법을 학습한다.

2. 과제 목표 및 주요 내용

(1) 과제목표

- 리스트 오퍼레이션을 이용한 항공권 예약 자료구조의 설계 및 구현
- 이중 연결 리스트를 이용한 항공편 리스트 설계 및 구현
- 이중 연결 리스트를 이용한 탑승객의 리스트 설계 및 구현

(2) 주요 내용

이번 프로젝트에서는 이중 연결 리스트라는 자료구조를 사용하여야 한다. 뒤에 필요한 자료구조의 자세한 내용을 설명하겠지만 우선 항공편(flight)에 대한 순서리스트와 예약 고객에 대한 리스트가 이중 연결 리스트로 구성되어야 한다. 이런 자료구조를 이용하여 항공편을 관리하고 고객의

예약을 처리하고 여러 가지 질의에 답하는 오퍼레이션을 구현하시오.

□ 새로운 항공편을 리스트에 추가(insert)

IF *fright_number from_city to_city capacity*
fright_number는 4자리의 숫자(1234나 0096과 같이)이며,
from_city는 비행기의 출발지로 30자까지의 도시이름이고,
to_city는 비행기의 도착지로 역시 30자까지의 도시이름이고,
capacity는 비행기가 최대 탑승 고객수로 3자리 수이다.
오류 검사: 이미 그 항공편이 리스트에 있다면 다시 추가하지 않는다.

□ 항공편을 리스트에서 삭제(delete)

DF *flight_number*
지시한 항공편을 리스트에서 삭제
오류 검사: 삭제하려는 항공편이 리스트에 없으면 오류

□ 고객을 항공편에 예약

RP *name flight_number*
name은 고객의 이름이고 20자까지의 문자.
고객을 지정한 항공편에 예약한다. 만일 자리에 없다면 대기 리스트에 추가한다.
오류 검사: 예약하려는 사람의 이름이 이미 예약자나 대기 리스트에 있다면 오류, 또한
예약하려는 항공편이 리스트에 없다면 오류이다.

□ 고객의 예약을 취소

CP *name flight_number*
고객의 예약 리스트에서 지정한 사람을 삭제한 후 대기 리스트의 첫 번째 고객을 예약자 리스트에 추가한다.
만일 flight_number가 0이라면 그 고객의 모든 예약과 대기를 취소한다.
오류 검사: 고객이름이 리스트에 없거나 항공편이 리스트에 없다면 오류

□ 인쇄

■ PA
예약 정보를 모두 인쇄하되 항공편의 순서에 따라 이미 예약된 고객과 대기자 명단을 인쇄한다. 대기자 명단은 큐에 도착한 순서대로 인쇄한다.
■ PF

항공편에 대한 정보(항공편 번호, 출발지, 도착지, 최대 탑승객수, 예약된 고객 수)를 항공편 번호 순으로 인쇄한다.

■ PR *flight_number*

특정 항공편에 예약된 고객을 이름순으로 인쇄하고 대기 중인 고객을 큐에 입력된 순서로 인쇄한다.
오류 검사: 항공편이 없다면 오류

■ PP *name*

지정된 이름의 고객에 대한 비행 계획을 항공편 번호 순으로 인쇄한다. 고객이 예약한 항공편 번호, 출발지, 도착지, 예약이 OK 되었는지 대기 상태인지를 인쇄한다.
오류 검사: 고객의 이름이 리스트에 있어야 한다.
입력 파일에는 한 줄에 하나씩의 명령어가 기록되어 있다. 메인 프로그램에서는 입력 파일을 한 줄씩 읽어 해당되는 명령어를 echo 프린트하고 처리한다. 만일 명령어에 오류가 있다면 오류 메시지를 내 보낸 후 다음 명령어를 처리한다.

3. 자료 구조의 설계

다음과 같은 여러 가지의 이중 연결리스트가 사용되어야 한다.

□ 항공편 번호 순으로 정렬된 항공편 리스트
□ 고객의 리스트
□ 고객의 예약에 대한 모든 정보는 다음 세 가지 방법으로 연결된 리스트로 구성되어야 한다.

항공편마다 확정된 예약 리스트와 대기 리스트(리스트의 고객은 이름순으로 순서 정렬되어 있어야 하며 대기 리스트에는 큐에 도착한 순으로 추가한다)가 있어야 한다.
고객마다 예약된 또는 대기 된 항공편의 리스트가 있어야 한다.
결국 다음 세 가지 타입의 리스트 노드에 대한 정의가 필요하다.

□ 항공편을 위한 다음 정보를 보관하는 노드
■ 항공편에 대한 정보: 번호, 출발지, 도착지, 최대 탑승객 수
■ 예약이 끝났는지 표시
■ 다음 항공편에 대한 포인터
■ 예약 리스트에 헤더에 대한 포인터
■ 대기 리스트의 front 및 rear에 대한 포인터
■ 다른 정보: 예를 들면 예약 OK된 고객의 수
□ 고객에 대한 노드는 다음과 같은 정보를 포함하고 있어야 한다.
■ 이름
■ 다음(바로 전) 고객에 대한 포인터
■ 예약 리스트에 대한 포인터
■ 다른 필요한 정보

□ 항공편에 대한 고객의 실제 예약 정보 노드
■ 고객 노드에 대한 포인터
■ 항공편에 대한 포인터
■ 이 고객이 예약한 다음(이전) 항공편 예약 노드대한 포인터
■ 같은 항공편을 예약한 다음(이전) 고객에 대한 예약 노드 포인터
■ 이 예약이 OK되었는지 대기상태인지 알리는 flag

모든 노드는 리스트에 적당한 자리에 추가되고 삽입되어야 한다. 즉 예약 리스트에 예약 노드를 추가할 때는 이름순으로 순서정렬이 되도록 맞는 순서를 찾아 추가한다. 이렇게 하면 나중에 순서정렬을 할 필요가 없다. 모든 정보를 인쇄할 때는 해당되는 연결 리스트를 방문하면서 출력하면 된다.

4. 실행

제공되는 테스트 화일(samplein.p1)을 입력 받아 처리하고 출력된 결과를 OUTFILE.P1으로 저장하시오.

<samplein.p1>

```
IF 0056 Lafayette Detroit 004
IF 1234 Chicago Baltimore 006
IF 0196 Dayton Indianapolis 003
PA
RP Smith 0056
RP Hubbard 0196
RP Jones 0056
RP Churchill 0056
RP Sauvage 0056
RP Smith 0196
RP Smith 0056
RP Smith 1234
RP Jones 0196
RP Hubbard 0056
PF
PR 0056
PP Smith
CP Churchill 0056
RP Churchill 0196
RP Wiley 0196
PA
CP Smith 0
PR 0196
DF 0196
PA
```