<CPP 최종보고서>

<구현한 모든 기능>

**1. 클래스 및 객체 지향 프로그래밍 (OOP)**

* **클래스 정의**: FlightData, PricePredictor는 여행지와 추천 엔진의 데이터를 구조화하고 기능을 캡슐화하기 위해 사용.
* **멤버 함수**: 객체의 행동을 정의.
* calculateAveragePrice(), getLowestPrice() 등.
* **접근 제어자**:
* public: 클래스 외부에서 접근 가능한 데이터와 메서드.
* private: PricePredictor 클래스의 flightDataList처럼 외부에서 접근 불가능.

**2. STL 컨테이너 (std::vector)**

* **정의**: 동적 배열로, 크기를 동적으로 조절 가능.
* **활용**:
  + 여행지 데이터를 저장: vector<int> prices;, vector<string> bestMonths;.
  + 여행지 리스트 관리: vector<FlightData\*> flightDataList;.
* 주요 메서드:
  + push\_back: 데이터를 추가.
  + erase + remove\_if: 조건에 따라 요소 삭제.
  + sort: 정렬.
  + find, min\_element, max\_element: 검색 및 값 찾기.

**3. 표준 라이브러리 알고리즘 (<algorithm>)**

* **std::sort**:
  + 데이터를 특정 기준에 따라 정렬.
* **std::min\_element / std::max\_element**:
  + 벡터에서 최솟값 또는 최댓값 찾기.
* **std::find**:
  + 특정 값이 벡터에 존재하는지 확인.
* **std::remove\_if**:
  + 조건을 만족하는 요소를 제거.

**4. 입력/출력 스트림 (<iostream>)**

* **cin**: 사용자 입력 받기.
* **cout**: 데이터 출력.
* **getline**: 전체 줄을 입력받기.
* **endl**: 줄바꿈 및 출력 버퍼 플러시.

**5. 메모리 관리 (new와 delete)**

* **new**: 동적으로 객체 생성.
* **delete**: 동적 메모리 해제.

**6. 조건문과 반복문**

* **조건문 (if, else)**:
* **반복문 (for, for-each)**:

**7. 람다 함수**

* **정의**: 익명 함수로, 간단한 작업을 표현.
* **활용**:
  + 정렬 기준 설정.
* remove\_if 조건 설정.

**8. 문자열 처리 (std::string)**

* **std::string**:
  + 문자열 데이터를 관리.
  + 예: string country;, string month;.
* 주요 메서드:
  + empty: 문자열이 비어 있는지 확인.
  + getline: 사용자로부터 전체 줄 입력받기.

**9. 산술 및 논리 연산자**

* **산술 연산자**: +, -, \*, / 등으로 계산 수행.
* **논리 연산자**: &&, ||, !로 조건 결합.

**10. 삼항 연산자**

* **용도**: 간단한 조건에 따른 값을 반환.

**11. 타입 캐스팅**

* **static\_cast**:
  + 타입을 명시적으로 변환.

**12. 헤더 파일**

* **<iostream>**: 입출력을 처리.
* **<vector>**: 동적 배열 사용.
* **<string>**: 문자열 처리.
* **<algorithm>**: STL 알고리즘 사용.
* **<limits>**: 수치 데이터의 한계값 사용

**테스트 결과**

텍스트, 스크린샷, 메뉴, 폰트이(가) 표시된 사진

자동 생성된 설명

**제안서에서 목표했던 기능**

1. **개인화된 여행지 추천**:
   * 개인의 **성향**, **예산**, **여행 목적**을 반영하여 최적의 여행지를 추천.
   * 예산과 목적에 맞는 맞춤형 추천 제공.
2. **시기별 항공권 추천**:
   * 계절과 성수기를 기반으로 항공권 가격 변화를 분석.
   * 사용자가 입력한 여행 날짜에 따른 항공권 가격 추세 제공.
3. **비용 시뮬레이션**:
   * 여행 기간, 식사 유형, 숙소 종류에 따른 예상 의식주 비용 계산.
   * 입력한 조건을 바탕으로 전체 여행 예산을 계산하고 시뮬레이션.
4. **차별점**:
   * 예산을 기준으로 한 최적의 여행지 추천.
   * 경제적이면서도 만족스러운 여행 경험을 제공.
   * 숙박, 교통, 식비 등 여행 요소를 세분화하여 맞춤형 정보 제공.

**구현된 기능 (개인적인 목표 부합도 포함)**

1. **시기별 항공권 추천**:
   * 여행지별 항공권 가격(최저가, 최고가, 평균가) 계산 및 추천.
   * 특정 월에 적합한 여행지 추천 기능 추가.
   * 성수기, 비수기와 관련된 기본 데이터 관리 가능.
   * **제안서 내용과의 부합도**: **90%**
     + 항공권 데이터를 기반으로 추천 가능하지만, 가격 추세나 변동성을 분석하는 고급 기능은 구현되지 않음.
2. **비용 시뮬레이션**:
   * 식사 유형(저가, 중급, 고급)과 숙소 유형(호스텔, 호텔, 고급 숙소)에 따라 비용 계산.
   * 총 여행 비용 시뮬레이션 제공.
   * **제안서 내용과의 부합도**: **80%**
     + 여행 요소를 사용자가 세부적으로 입력하고, 이에 따라 비용을 계산하는 기능은 구현.
     + 다만, 더 세분화된 항목(교통, 활동비 등)은 반영되지 않음.
3. **개인화된 여행지 추천**:
   * 사용자의 예산과 여행 기간을 기반으로 적합한 여행지 필터링.
   * 특정 월에 방문하기 좋은 여행지 추천 가능.
   * **제안서 내용과의 부합도**: **70%**
     + 개인의 성향(예: 활동 유형, 선호도 등)을 반영하지 못함.
     + 단순히 예산과 월에 따른 필터링만 수행.
4. **차별점 구현 여부**:
   * 항공권 가격 기반의 가성비 높은 여행지 추천 구현.
   * 비용을 기준으로 한 추천은 가능하지만, 사용자의 여행 목적을 세밀히 반영하는 로직은 부족.
   * **제안서 내용과의 부합도**: **60%**
     + 실제 여행 요소(교통, 숙박 등)를 데이터로 처리하는 기능 부족.

**구현하지 못한 부분**

1. **실시간 데이터 활용 부족**:
   * 실시간 항공권 및 숙박 비용 데이터를 반영하지 못함.
   * API를 통한 데이터 통합이 필요.
2. **여행 목적에 따른 추천 부재**:
   * 사용자의 활동 선호도(예: 휴양, 탐험, 문화 체험 등)를 고려하지 못함.
3. **세분화 부족**:
   * 교통비, 활동비 등 추가적인 여행 요소를 비용 계산에 포함하지 못함.

**총평**

* **제안서 목표 대비 구현도: 약 75%**
  + 기본적인 여행지 추천 및 비용 계산 기능은 충실히 구현.
  + 다만, 실시간 데이터와 개인화된 추천 로직이 부족해 목표의 완전한 달성에는 미치지 못함.