C++프로그래밍 및 실습

Off-basket(오프라인 장바구니)

최종 보고서

제출일자: 2023.12.24
제출자명: 김서연
제출자학번: 222400

1. 프로젝트 목표 (16 pt)

1) 배경 및 필요성 (14 pt)

온라인 쇼핑몰에는 다양한 옷들이 있지만, 우리가 옷을 직접 매장에서 구매하는 이유는 인터넷으로만 보았을 때와 실제로 입어보았을 때 옷 재질과 착용감의 차이 때문이다. 따라서 나 또한 바지나 겉옷과 같은 옷은 주로 오프라인 매장에서 구매하는 편이다. 그러나 쇼핑을 하려는 마음이 생겼을 때, 어떤 옷 가게가 내 스타일과 맞는지 사전에 파악하기 어려워서 쇼핑에 많은 시간을 투자했고, 원하는 옷을 찾는 데 어려움을 겪었다.
이러한 문제점을 해결하기 위해서 생각해낸 게 "오프라인 장바구니"라는 앱이다. 이 앱을 사용하면 먼저 쇼핑을 하고자 하는 지역을 선택할 수 있다. 예를 들어 광주광역시 동구를 선택하게 되면, 해당 지역에 있는 여러 개인 옷 가게들이 리스트로 표시된다. 이러한 옷가게들을 하나씩 클릭하여 원하는 옷을 선택할 수 있다. 맘에 드는 옷은 앱의 장바구니에 넣어두고, 실제로 쇼핑을 하러 나가기로 결정했을 때 해당 옷가게를 찾아가서 직접 입어보고 구매 결정을 내릴 수 있다. 이 앱을 활용하면, 옷을 살 때 소요되는 시간을 절약할 뿐만 아니라 다양한 옷을 살펴볼 수 있으며, 자신이 원하는 옷을 직접 입어보고 구매할 수 있다. 특히 여성 구두와 같은 경우, 온라인으로 구매한 제품의 사이즈가 안 맞을 수 있기 때문에 이 앱을 활용하면 편할 거 같다.

2) 프로젝트 목표

고객들이 쇼핑하고자 하는 지역을 선택 후, 원하는 옷들을 장바구니에 담고, 옷의 수량을 체크하면서, 오프라인 매장에 대한 리뷰도 쓸 수 있게 만들 예정이다.
원래는 앱을 개발하고 싶었으나 실력이 부족해 이번 프로젝트는 광주광역시 동구 지역을 기준으로 작은 기능들을 구현해 볼 것이다

3) 차별점

기존 쇼핑몰 앱들은 대부분 온라인 옷가게들만 제공하고 있어, 실제로 옷을

직접 입어보고 살 수 없다. 하지만 이 앱에는 사용자가 쇼핑하고자 하는 특정

지역의 오프라인 매장들로 만 구성되어 있어, 담아 둔 옷들을 직접 착용해보고 구매할 수 있는 차별화된 기능을 제공한다.

2. 기능 계획

1) 옷 가게 위치 확인하기

① 상위지역 목록 출력하기

상위지역으로는 우리나라의 행정 구역인 1개의 특별시, 6개의 광역시, 7개의 도, 2개의 특별자치도, 1개의 특별자치시로 구성한다. 17개의 상위지역을 화면에 표시해 사용자가 선택할 수 있게 한다.

② 상위지역 입력 받기

사용자는 자신이 옷을 사고 싶은 상위지역을 17개중 하나 선택한다.
ex) 광주광역시

③ 선택된 상위지역에 대한 하위지역 목록 출력

선택한 상위지역에 대한 하위지역을 출력한다.
ex) 광주광역시 – 동구, 서구, 남구, 북구
전라남도 – 목포시, 여수시, 순천시, 나주시 등등

④ 하위지역 입력 받기

사용자는 선택한 상위지역에 대한 하위지역도 선택한다.
ex) 광주광역시 - 동구

⑤ 선택된 지역에 있는 옷가게들의 이름과 주소 출력

사용자가 상위지역과 하위지역을 선택하면 해당 지역의 옷가게들의 이름과 주소를 표시한다. 이는 사용자가 원하는 옷가게를 쉽게 찾을 수 있도록 도와준다.

2) 옷 재고 관리

옷 가게 관리자는 일상적으로 옷 수량을 업데이트해야 한다. 이렇게 하면 사용자들은 옷을 구매하기 전에 실제 수량을 확인할 수 있게 되며, 더욱 편리하게 쇼핑을 즐길 수 있다. 이 프로젝트에서는 사용자와 관리자 모두 활용할 수 있는 기능을 만들고 싶다. 따라서 관리자는 옷 수량을 업데이트하고, 그 업데이트 한 수량은 파일에 저장되어 관리하기 쉽게 만들 것이다. 또한 사용자는 이를 확인하는 기능을 구현해 볼 예정이다.

(단, 옷이 팔려서 수량이 줄어드는 기능은 구현하기 힘들기 때문에 아침마다 관리자가 직접 입력한 수량으로 체크할 것이다.)

① 메뉴 선택하기

1. 재고 업데이트
2. 재고 확인
3. 수량 파일 확인
4. 종료하기

이렇게 4가지의 메뉴를 선택할 수 있는데 재고 업데이트와 수량파일 확인은 관리자가 사용하는 기능이고, 재고 확인하는 기능은 사용자가 사용하는 기능이다. 파일을 만들 때는 아침마다 재고가 바뀌어야 되기 때문에 ios::trunc를 사용해서 새로운 내용으로 매번 교체 할 예정이다.

② 재고 업데이트 기능 (관리자)

관리자는 아침마다 가게 이름, 매장에 있는 옷의 이름, 수량을 입력한다.

③ 재고 확인하기 (사용자)

사용자는 아침에 관리자가 입력한 실제 수량을 확인한다.

④ 수량 파일 확인하기 (관리자)

관리자가 입력한 재고를 파일에 저장해 관리한다. 이 기능은 관리자가 옷 재고를 한눈에 보기가 쉬워서 재고 관리하기 편하다.

⑤ 추가 기능(오류 처리)

메뉴를 선택할 때 숫자가 아닌 문자열을 잘 못 입력한 경우, "숫자만 입력해주세요" 라고 출력한다.

3) 원하는 옷 장바구니에 담아두기

앱을 개발하는 게 아니라 어려움이 있지만 최대한 비슷하게 구현해 볼 예정이다.

① 옷 상세정보 출력하기

모든 옷 가게에 대한 옷 종류가 표시된다.
ex) 옷 색상, 종류, 재질, 사이즈,

② 파일에 저장하기

사용자가 맘에 드는 옷들을 선택하면 이러한 선택사항이 파일에 저장된다. (여기서 파일은 앱의 장바구니와 같은 의미다.)

4) 오프라인 매장 리뷰 작성/확인하기

선호하는 스타일의 옷 매장을 쉽게 찾는 방법은 리뷰를 읽는 것이다.
리뷰를 통해 옷의 재질과 품질이 좋은 지 확인할 수 있어서 조금 더 쉽게 옷을 고를 수 있다.

① 리뷰 작성하기

사용자는 그 옷 매장에 대해 리뷰를 남길 수 있다.
ex) 넉넉한 핏이고 생각보다 두꺼워서 따뜻하게 잘 입고 다녔어요!!
색감은 좀 어두운 거 같아요. 그래도 예뻐서 만족합니다.

② 리뷰 확인하기

사용자들은 옷의 재질과 색상을 미리 확인하기 위해 리뷰를 참고할 수

있다. 장바구니에 담아두고 매장에 방문하기 전에 리뷰를 통해 선별하면, 쇼핑할 때 시간을 절약할 수 있다.

3. 기능 구현

1) 기능1 구현: 옷 가게 위치 확인하기

(1) 상위지역 목록 출력하기

- 입출력

  입력: majorRegions = 상위 지역 목록

  ⇨ 이 벡터에는 출력하고자 하는 상위 지역들의 이름이 문자열 형태로 저장되어 있다.

  출력: 구분선, majorRegions 벡터에 있는 상위 지역 목록

- 설명

  상위 지역들의 이름을 담고 있는 문자열 벡터를 받아, 그 내용을 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  PrintMajorRegions 함수를 정의하고 있다.
  함수의 매개변수를 정의할 때 동적 배열을 가지고 있는 벡터를 사용하였다.
  또한 참조자를 사용해 매개변수가 전달될 때 원본 데이터에 대한 참조(주소)만 전달되게 만들었다. 이는 데이터 복사가 없어 메모리 사용량과 처리 시간을 절약할 수 있다. majorRegions 벡터의 각 요소들을 출력하기 위해 for 반복문을 사용하였다.
  I는 0부터 시작해 majorRegions 크기보다 작을 때까지 반복한다.
  이렇게 정의한 함수는 상위지역의 목록을 출력하는 기능을 한다.

-코드 스크린샷

```cpp
// 기능1 함수 정의
// 1.1 상위 지역의 목록을 출력하는 함수
void printMajorRegions(const std::vector<std::string>& majorRegions) {
    std::cout << "=========================================" << std::endl;
    // 반복문을 이용해 majorRegions에 있는 모든 요소들 출력하기
    for (int i = 0; i < majorRegions.size(); i++) {
        std::cout << i + 1 << ": " << majorRegions[i] << std::endl;
    }
    std::cout << "=========================================" << std::endl;
}
```

(2) 상위지역 입력 받기

- 입출력

  입력:

  1. majorRegions = 상위 지역 목록
  2. majorRegionIndex = 사용자 입력 값

  출력: 구분선, 선택한 상위 지역의 인덱스

- 설명

  사용자에게 상위 지역 인덱스를 입력 받아, 해당 인덱스를 반환한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  selectMajorRegion 함수를 정의하고 있다.
  함수의 매개변수를 정의할 때 동적 배열을 가지고 있는 벡터를 사용하였다.
  또한 참조자를 사용해 매개변수가 전달될 때 원본 데이터에 대한 참조(주소)만 전달되게 만들었다. 이는 데이터 복사가 없어 메모리 사용량과 처리 시간을 절약할 수 있다.
  사용자가 선택한 상위 지역을 출력하기 위해서 사용자로부터 원하는 지역의 인덱스를 입력하라는 메시지를 출력하고, 입력 받은 인덱스 값을 함수의 호출자에게 반환한다.

- 코드 스크린샷

```
// 1.2 상위지역 입력 받는 함수
int selectMajorRegion(const std::vector<std::string>& majorRegions) {
    std::cout << "원하는 상위 지역을 선택하세요: ";
    int majorRegionIndex;
    std::cin >> majorRegionIndex;
    std::cout << "=========================================" << std::endl;
    return majorRegionIndex;
}
```

(3) 선택된 상위 지역에 대한 하위 지역 목록 출력

- 입출력

  입력:

  1. minorRegions

     -> 이 매개변수는 상위 지역의 이름을 키로 하고, 해당 상위 지역에 해당하는 하위 지역들의 이름 목록을 값으로 하는 map이다.

  2. selectedMajorRegion

     -> 이 매개변수는 사용자가 선택한 상위 지역의 이름이다.

  출력: 구분선, 선택한 상위 지역에 해당하는 하위 지역들의 목록

- 설명

  주어진 상위 지역에 해당하는 하위 지역들의 목록을 사용자에게 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  printMinorRegions 함수를 정의하고 있다.
  함수의 매개변수를 정의할 때 동적 배열을 가지고 있는 벡터를 사용하였다.
  또한 참조자를 사용해 매개변수가 전달될 때 원본 데이터에 대한 참조(주소)만 전달되게 만들었다. 이는 데이터 복사가 없어 메모리 사용량과 처리 시간을 절약할 수 있다.
  반복문 for문을 이용해 각 벡터의 하위 지역에 대해 반복한다.
  selectedMajorRegion을 키로 사용하여 이에 해당하는 하위 지역들의 벡터를 반환한다.
  auto&를 이용해 자동 타입 추론을 사용하였고, 하위 지역의 이름을 참조한다.

- 코드 스크린샷

```
// 1.3 상위 지역에 대한 하위 지역 목록을 사용자에게 출력하는 함수
void printMinorRegions(
    const std::map<std::string, std::vector<std::string>>& minorRegions,
    const std::string& selectedMajorRegion) {
  int i = 1;
  // 반복문을 이용해 selectedMajorRegion에 해당하는 하위 지역들의 벡터를 반환
  for (const auto& region : minorRegions.at(selectedMajorRegion)) {
    std::cout << i++ << ": " << region << std::endl;
  }
  std::cout << "========================================" << std::endl;
}
```

(4) 하위 지역 입력 받기

- 입출력

  입력:

  1. minorRegions = 하위 지역 목록
  2. selectedMajorRegion = 사용자가 선택한 상위 지역 목록
  3. minorRegionIndex = 사용자 입력 값

  출력: 구분선, 선택한 하위 지역

- 설명

  사용자에게 하위 지역 인덱스를 입력 받아, 해당 인덱스를 반환한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  selectMinorRegion 함수를 정의하고 있다.
  함수의 매개변수를 정의할 때 동적 배열을 가지고 있는 벡터를 사용하였다.
  또한 참조자를 사용해 매개변수가 전달될 때 원본 데이터에 대한 참조(주소)만 전달되게 만들었다. 이는 데이터 복사가 없어 메모리 사용량과 처리 시간을 절약할 수 있다.
  사용자가 선택한 하위 지역을 출력하기 위해서 사용자로부터 원하는 지역의 인덱스를 입력하라는 메시지를 출력하고, 입력 받은 인덱스 값을 함수의 호출자에게 반환한다.

- 코드 스크린샷

```
// 1.4 하위지역 입력 받는 함수
int selectMinorRegion(
    const std::map<std::string, std::vector<std::string>>& minorRegions,
    const std::string& selectedMajorRegion) {
  std::cout << selectedMajorRegion << " 내에서 하위 지역을 선택하세요: ";
  int minorRegionIndex;
  std::cin >> minorRegionIndex;
  std::cout << "=========================================" << std::endl;
  return minorRegionIndex;
}
```

(5) 선택된 지역에 있는 옷가게들의 이름과 주소 출력

- 입출력

  입력:

  1. stores = Store 객체들의 벡터
  2. majorRegion = 상위 지역 목록
  3. minorRegion = 하위 지역 목록

  출력: 구분선, 선택한 지역에 해당하는 옷 가게들의 이름과 주소
  해당 옷 가게가 없는 경우, 없다는 메시지

- 설명

  사용자가 선택한 지역에 대한 옷 가게 이름과 주소를 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  printStores 함수를 정의하고 있다.
  함수의 매개변수를 정의할 때 동적 배열을 가지고 있는 벡터를 사용하였다.
  또한 참조자를 사용해 매개변수가 전달될 때 원본 데이터에 대한 참조(주소)만 전달되게 만들었다. 이는 데이터 복사가 없어 메모리 사용량과 처리 시간을 절약할 수 있다.
  stores 벡터의 모든 Store객체를 순서대로 처리하기 위해 반복문 for문을 사용했다.

auto&를 이용해 자동 타입 추론을 사용하였고, Store 객체를 참조한다.

if 반복문을 사용해 사용자가 선택한 majorRegion과 minorRegion이 일치한지 확인하고, 일치하면 옷 가게의 이름과 주소를 출력한다.

만약 일치하지 않으면 "옷가게가 없다"는 메시지를 출력한다.

- 코드 스크린샷

```
// 1.5 선택된 지역에 있는 옷가게들의 이름과 주소 출력하는 함수
void printStores(const std::vector<Store>& stores,
                 const std::string& majorRegion,
                 const std::string& minorRegion) {
    bool found =
        false;  // found 변수 설정 (주어진 지역내에서 가게를 찾았는지에 대한 여부)
    std::cout << majorRegion << " " << minorRegion
              << "에 있는 옷 가게 목록:" << std::endl;
    // store벡터의 모든 Store 객체를 순서대로 한 번씩 처리하는 반복문
    for (const auto& store : stores) {
        // 찾고자 하는 지역에, 옷가게가 있을 경우 이름과 주소 출력
        if (store.majorRegion == majorRegion && store.minorRegion == minorRegion) {
            std::cout << store.name << ": " << store.address << std::endl;
            found = true;
        }
    }

    // 해당 지역에 옷가게가 없을 경우
    if (!found) {
        std::cout << majorRegion << " " << minorRegion << "에는 옷 가게가 없습니다."
                  << std::endl;
    }
}
```

2) 기능2 구현: 옷 재고 관리

<기능2. cpp>파일

(1) 메뉴 선택하기 (1. 재고 업데이트, 2. 재고 확인, 3. 수량 파일 확인, 4. 종료)

- 입출력

  입력: choice = 메뉴 선택

  출력: 구분선, 메뉴 리스트

- 설명

  4가지의 메뉴리스트를 출력하고, 사용자는 사용하고 싶은 기능을 선택할 수 있다.

  만약 사용자가 숫자가 아닌 문자열을 입력한 경우에는 validNumber Input 함수가 호출

되어, 오류 메시지가 뜬다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

If조건문을 이용해 입력된 값이 숫자인지 검증한다. 입력이 숫자가 아닐 경우에는 false를 반환하고, 숫자일 경우에는 true를 반환한다. 만약 숫자면 continue문을 통해 다음 반복으로 진행한다. 또한 입력 받은 값에 따라 다른 작업을 수행하는 switch 조건문을 사용했다.

- 코드 스크린샷

```
// 기능2 : 옷 재고 관리
void manageInventory(ClothingInventory& inventory, const std::vector<Store>& stores) {
    while (true) {
        // 2.1 메뉴 선택하기
        std::cout << "₩n2. 옷 재고 관리" << endl;
        std::cout << "========================================" << std::endl;
        std::cout << "1. 재고 업데이트" << std::endl;
        std::cout << "2. 재고 확인" << std::endl;
        std::cout << "3. 수량 파일 확인" << std::endl;
        std::cout << "4. 종료" << std::endl;
        std::cout << "========================================" << std::endl;
        std::cout << "메뉴를 선택해주세요: ";


        // 사용자에게 메뉴 선택 입력받기
        int choice;
        std::cin >> choice;
        if (!validNumberInput()) continue; // 숫자 입력 검증
        std::cin.ignore();

        switch (choice) {
```

(2) 재고 업데이트

- 입출력

입력: storeName, itemName, quantiy (가게 이름, 상품 이름, 수량)

출력: 구분선, 입력 안내 메시지, 가게를 찾지 못 했을 때의 메시지

- 설명

사용자로부터 가게이름, 상품 이름, 그리고 수량을 입력 받아 재고를 업데이트한다. 사용자가 유효한 가게이름을 입력하지 않았을 경우, 오류 메시지를 표시하고, 유효한 입력인 경우에만 재고를 업데이트 할 수 있다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

Bool storeFound = false 라는 논리형 자료형을 이용해 일치하는 가게를 찾았는지 여부를 추적한다. For 반복문을 사용해 사용자가 입력한 storeName과 일치하는 가게를 찾는다. If 조건문을 이용해서 만약 일치하는 가게를 찾으면 updateStock 함수를 호출하여 재고를 업데이트한다.

- 코드 스크린샷

```
// 2-2. 재고 업데이트
case 1: {
    // 사용자로부터 가게 이름, 상품 이름, 수량 입력받기
    std::string storeName, itemName;
    int quantity;
    std::cout << "가게 이름: ";
    getline(std::cin, storeName);
    std::cout << "상품 이름: ";
    getline(std::cin, itemName);
    std::cout << "수량: ";
    std::cin >> quantity;
    // 숫자 입력 검증
    if (!validNumberInput()) continue;
    std::cin.ignore();

    bool storeFound = false;  // storeFound 라는 변수 선언 : 일치하는 가게를 찾았는지 여부 추적
    for (const auto& store : stores) {
        // store 객체의 옷가게와 이름이 같으면 updateStock 함수 호출
        if (store.name == storeName) {
            inventory.updateStock(store, itemName, quantity);
            storeFound = true;
            break;
        }
    }
    // 가게이름이 다를 때
    if (!storeFound) {
        std::cout << "가게를 찾을 수 없습니다." << std::endl;
    }
    break;
}
```

(3) 재고 확인하기

- 입출력

입력: storeName, itemName (가게 이름, 상품 이름)

출력: 구분선, 입력 안내 메시지, 가게를 찾지 못 했을 때의 메시지

- 설명

사용자로부터 가게 이름과 상품 이름을 입력 받아 해당 상품의 재고 상태를 확인한다. 사용자가 유효한 가게 이름을 입력하지 않았을 경우, 적절한 오류 메시지를 표시하고, 유효한 입력인 경우에만 재고 확인을 수행한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

Bool storeFound = false 라는 논리형 자료형을 이용해 일치하는 가게를 찾았는지 여부를 추적한다. For 반복문을 사용해 사용자가 입력한 storeName과 일치하는 가게를 찾는다. If 조건문을 이용해서 만약 일치하는 가게를 찾으면 checkStock 함수를 호출하여 해당 상품의 재고를 확인한다.

- 코드 스크린샷

```
// 2-3. 재고 확인하기 기능
case 2: {
    std::string storeName, itemName;
    std::cout << "가게 이름: ";
    getline(std::cin, storeName);
    std::cout << "상품 이름: ";
    getline(std::cin, itemName);

    bool storeFound = false; // storeFound 라는 변수 선언 : 일치하는 가게를 찾았는지 여부 추적
    for (const auto& store : stores) {
        // store 객체의 옷가게와 이름이 같으면 checkStock 함수 호출
        if (store.name == storeName) {
            inventory.checkStock(store, itemName);
            storeFound = true;
            break;
        }
    }
    // 가게이름이 다를 때
    if (!storeFound) {
        std::cout << "가게를 찾을 수 없습니다." << std::endl;
    }
    break;
}
```

(4) 수량 파일 확인하기

- 입출력

입력: 수량.txt (재고 정보 파일)

⇨ Ifstream을 사용하여 파일을 읽기 모드로 연다.

출력: 구분선, "옷 재고 확인" 문구 출력, 파일에 있는 각 줄의 내용 출력

- 설명

수량.txt 파일의 내용을 확인하는 기능을 수행한다. 만약 파일 열기에 성공하면 파일의 내용을 한 줄씩 읽어 출력하고, 실패하면 오류 메시지를 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

파일 입출력 관련 클래스인 ifstream을 사용해 file을 생성한다. If조건문을 사용해 파일을 열 수 없는 경우, 오류 메시지를 표준 에러 출력 스트림 'cerr'을 통해 출력한다. 만약 파일 열기에 성공하면 while 반복문과 'getline'을 사용해 파일의 각 줄을 'line' 변수에 저장하고, 콘솔에 출력한다. 마지막으로 file.close 를 통해 파일 스트림을 닫는다.

- 코드 스크린샷

```
// 2-4. 수량 파일 확인하기
case 3: {
    // 수량.txt 파일 열기
    ifstream file("수량.txt");
    string line;

    // 파일 열기를 실패했을 때
    if (!file) {
        cerr << "파일 오픈에 실패하였습니다." << endl;
    }
    // 파일 내용 읽기 및 출력
    else {
        cout << "========================================" << endl;
        cout << "<옷 재고 확인>" << endl;
        // file의 내용을 한줄 씩 읽어 콘솔창에 출력
        while (getline(file, line)) {
            cout << line << endl;
        }
    }
    // 파일 닫기
    file.close();
    break;
}
```

(5) 입력 값 오류 처리

- 입출력

  입력: 사용자의 입력 값

  출력: 오류 메시지

- 설명

  사용자의 입력이 숫자인지 검증하는 함수이며, 숫자가 아닌 입력을 받았을 때 오류 메시지를 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

validNumber Input 함수를 정의했다. 만약 숫자가 아닌 값을 입력하면 false를 반환하고, 입력이 숫자일 경우 true를 반환한다.

- 코드 스크린샷

```
// 2-5 입력 값 오류 처리
bool validNumberInput() {
    if (std::cin.fail()) {
        std::cin.clear();
        std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        std::cout << "숫자만 입력해주세요." << std::endl;
        return false;
    }
    return true;
}
```

(6) 기타

- 입출력

입력: 없음

출력: 프로그램 종료 메시지, 잘못 선택했을 때의 오류 메시지

- 설명

사용자가 프로그램 종료를 선택했을 때 종료하고, 예상된 범위 외의 선택을 했을 때 해당하는 경고 메시지를 표시하고 다시 선택하게 한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

Switch문 내에서 case 중 어느 것에도 해당하지 않는 모든 상황을 default에서 처리한다. 이 코드에서는 1부터 4가 아닌 다른 숫자를 입력했을 때 해당한다.

- 코드 스크린샷

```
    // 프로그램 종료
    case 4: {
        std::cout << "프로그램을 종료합니다." << std::endl;
        return;
    }
    // 잘못 입력했을 때
    default: {
        std::cout << "잘못된 선택입니다. 다시 시도해 주세요." << std::endl;
        break;
    }
    }
}
```

<기능.h> 파일

=> ClothingInventory 클래스 정의하여 옷가게의 재고 관리 시스템 구현

(1) 파일에 재고 정보 저장하는 함수(saveToFile)

- 입출력

  입력: 없음

  출력: 수량.txt 파일

- 설명

  재고 정보를 파일에 저장하는 'saveToFile' 함수를 정의하고 있다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  파일 출력 파일 스트림 클래스인 ofstream을 사용해서 "수량.txt" 파일을 쓰기 모드로 연다. 파일 쓰기 작업이 완료되면 close()를 호출하여 파일 스트림을 닫는다.

- 코드 스크린샷

```
// 기능 2: 옷가게의 재고 관리하는 클래스
class ClothingInventory {
private:
    map<string, int> stock;

    // 재고 정보를 파일에 저장하는 함수
    void saveToFile() {
        ofstream file("수량.txt");
        for (const auto& pair : stock) {
            file << pair.first << " " << pair.second << endl;
        }
        file.close();
    }
```

(2) 파일 읽는 함수 (loadFromFile)

- 입출력

  입력: 수량.txt 파일

  출력: 없음

- 설명

  파일로부터 재고 데이터를 읽어 stock 맵에 로드하는 "loadFromFile"함수를 정의하고 있다. Ios::trunc를 사용해서 파일을 열 때 파일의 내용을 비워 관리자가 매일매일 새로운 데이터를 입력할 수 있게 했다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  입력 파일 스트림 클래스인 ifstream을 사용해 파일에서 데이터를 읽는다. Ios::trunc를 사용해서 새로운 내용으로 교체된다. 또한 while 반복문을 사용해 각 상품의 key와 수량을 읽어 들인다.

- 코드 스크린샷

```
// 파일을 읽는 함수
void loadFromFile() {
    // 매일매일 수량이 달라지기 때문에 파일이 열릴 때마다 내용이 비워짐 (고민)
    ifstream file("수량.txt", ios::trunc);
    string key;
    int quantity;
    while (file >> key >> quantity) {
        stock[key] = quantity;
    }
    file.close();
}
```

(3) 재고 업데이트하는 함수(updateStock)

- 입출력

  입력: 가게이름, 상품 이름, 수량

  출력: 재고 업데이트 정보

- 설명

  재고 정보를 업데이트하고 이를 파일에 저장하여 재고 상태를 최신 상태로 유지한다. 사용자는 가게와 상품, 그리고 수량 정보를 통해 재고를 업데이트 할 수 있으며, 이러한 업데이트는 콘솔에 출력되고 파일에 저장된다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  재고를 업데이트하는 함수를 정의하였고, 파라미터 store는 Store 타입의 참조 변수로, 업데이트할 재고가 속한 상점을 나타낸다. Item도 string 타입의 참조 변수로, 업데이트할 재고의 상품 이름을 나타낸다. Quantity는 해당 상품의 새로운 재고 수량을 나타낸다. Const는 함수 내에서 수정되지 않음을 의미한다.

- 코드 스크린샷

```
public:
    ClothingInventory() { loadFromFile(); }
    // 재고 업데이트하는 함수
    void updateStock(const Store& store, const string& item, int quantity) {
        string key = store.name + ":" + item;
        stock[key] = quantity;
        cout << "가게: " << store.name << ", 상품: " << item
             << ", 수량: " << quantity << endl;
        saveToFile();
    }
```

(4) 재고 체크하는 함수(checkStock)

- 입출력

  입력: 가게이름, 상품 이름, 수량

  출력: 재고 정보, 재고가 없다는 메시지

- 설명

  사용자는 상점과 상품의 이름을 통해 재고를 확인할 수 있으며, 해당 상품의 재고가 있으면 재고 수량을 콘솔에 출력하고, 없으면 없다는 메시지를 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  재고를 업데이트하는 함수를 정의하고 있다. 여기서 it은 특정원소를 가리키는 반복자이다. find함수를 통해 얻을 수 있다. 또한 ->인 포인터 접근 연산자를 사용해 반복자가 가리키는 객체(수량)에 접근한다.

- 코드 스크린샷

```
    // 재고 체크하는 함수
    void checkStock(const Store& store, const string& item) {
        string key = store.name + ":" + item;
        // stock 에서 key에 해당하는 요소 찾기
        auto it = stock.find(key);
        // 찾고자 하는 키가 존재하면 정보 출력하기
        if (it != stock.end()) {
            cout << "가게: " << store.name << ", 상품: " << item
                << ", 수량: " << it->second << endl;
        }
        else {
            cout << "재고가 없습니다."<< endl;
        }
    }
};
```

3) 기능3 구현: 원하는 옷 장바구니에 담아두기

(1) 옷의 상세정보 출력

- 입출력

  입력: subchoice

  출력: 구분선, 메뉴 리스트, 옷 상품 정보

- 설명

  사용자가 메뉴에서 옷 상품보기를 누르면 벡터에 저장된 옷 상품 정보를 순회하면서 각 상품의 상세정보를 출력한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  If 조건문을 사용해 변수의 값이 1인 경우에 옷 상세정보를 보여준다.
  for 루프를 사용하여 menu 벡터의 각 항목을 반복하고, displayDetails 함수를 통해 menu 벡터의 1번째 요소인 옷 상품의 상세정보를 표시한다.

- 코드 스크린샷

<main.cpp>

```
switch (mainChoice) {
  // 기능3 : 옷 장바구니에 담기
  case 1: {
    subChoice = -1;
    while (subChoice != 0) {
      std::cout << "₩n3. 장바구니 메뉴" << std::endl;
      std::cout << "========================================="
                << std::endl;
      std::cout << "1: 옷 상품 보기" << std::endl;
      std::cout << "2: 장바구니에 담기" << std::endl;
      std::cout << "0: 메인메뉴 돌아가기" << std::endl;
      std::cout << "========================================="
                << std::endl;
      std::cout << "옵션을 선택해주세요: ";
      std::cin >> subChoice;

      // 3.1 옷 상세정보 출력하기
      if (subChoice == 1) {
        // 옷 상품 보기
        for (size_t i = 0; i < menu.size(); ++i) {
          std::cout << i + 1 << ": ";
          menu[i].displayDetails();
        }
      }
```

<기능3.cpp> 함수 정의

```
// 옷의 상세정보 출력하는 함수
void ClothingItem::displayDetails() const {
  std::cout << store << " => "
            << "옷 종류: " << type << ", 색깔: " << color
            << ", 재질: " << material << ", 사이즈: " << size << std::endl;
}
```

=> 옷 상세정보 출력

(2) 장바구니에 담기

- 입출력

  입력: 옷 상품의 번호, subchoice

  출력: 안내문

- 설명

  사용자로부터 옷 상품의 번호를 입력 받아 해당 상품을 장바구니(파일)에 추가한다. 만약 입력한 번호가 유효한 범위를 벗어난 경우 유효하지 않는 번호라는 메시지를 표시한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  If 조건문을 사용해 입력한 인덱스가 menu벡터의 범위 내에 있는지 확인한다. 만약 범위 내에 있다면 선택한 상품을 push_back을 사용하여 장바구니 cart 벡터에 추가하고 아닐 경우에는 오류 메시지 출력한다. 마지막으로 saveCartToFile 함수를 써서 cart.txt 파일에 저장한다.

- 코드 스크린샷

<main.cpp>

```
    // 3.2 장바구니에 담기
    else if (subChoice == 2) {
        // 장바구니에 담기
        std::cout << "담을 옷 상품의 번호를 입력하세요: ";
        int index;
        std::cin >> index;
        --index;

        if (index >= 0 && index < menu.size()) {
            cart.push_back(menu[index]);
            std::cout << "상품이 장바구니에 담겼습니다." << std::endl;
        } else {
            std::cout << "유효하지 않는 번호입니다." << std::endl;
        }
        saveCartToFile(cart, "cart.txt");
    }
}
break;
```

- 함수 설명

  장바구니에 있는 옷 상품 정보를 주어진 파일에 저장하는 함수이다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

함수를 정의했고, ofstream 통해 파일을 연다. For 반복문을 통해 장바구니 벡터에 있는 각 clothingitem 객체를 반복해서 객체의 정보를 파일에 저장한다. 작업이 다 끝나면 file.close를 사용해서 파일을 닫는다.

- 코드 스크린샷

<기능3. Cpp>

```
void saveCartToFile(const std::vector<ClothingItem>& cart,
                    const std::string& filename) {
    std::ofstream file(filename);

    if (!file) {
        std::cerr << "파일을 열 수 없습니다: " << filename << std::endl;
        return;
    }

    for (const auto& item : cart) {
        file << item.store << "," << item.type << "," << item.color << ","
             << item.size << std::endl;
    }

    file.close();
    std::cout << "장바구니가 파일에 저장되었습니다: " << filename << std::endl;
}
```

<기능.h>

-클래스 설명

옷 상품의 정보를 표현하고 관리하는 데 사용되는 클래스이다. displayDetails 함수를 사용하여 이 클래스에 포함 되어있는 객체의 정보를 출력할 것이다.

-적용된 배운 내용

멤버 변수 store, type, color, material, size를 지정하고, 멤버 함수 displayDetails(), gestStoreName을 정의해준다. getStoreName 함수는 store 멤버 변수의 값을 반환하는 함수다.

-코드 스크린 샷

```cpp
// 기능 3: 옷에 관한 정보를 나타내는 클래스
class ClothingItem {
public:
    std::string store;     // 가게 이름
    std::string type;      // ex) 상의, 하의
    std::string color;     // 옷 색깔
    std::string material;  // 옷 재질
    std::string size;      // 옷 사이즈

    ClothingItem(std::string st, std::string t, std::string c, std::string m,
        std::string s);

    void displayDetails() const;

    std::string getStoreName() const { return store; }
};
```

4) 기능4 구현: 리뷰 작성하고 확인하기

(1) 리뷰 작성하기

- 입출력

  입력: subchoice , 리뷰 작성할 때 매장 이름, 리뷰 내용

  출력: 메뉴 옵션, 입력을 위한 안내 메시지

- 설명

  사용자가 1을 입력하면 매장을 입력하고, 리뷰를 작성할 수 있다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  While 반복문을 사용해 사용자가 0을 입력하여 루프를 종료할 때까지 계속 실행한다.
  for 반복문을 사용해 입력한 매장 이름과 일치하는 매장이 같은지 확인한다.
  switch문을 사용해 사용자의 선택에 따라 작업을 수행한다. 또한 auto를 사용하여 변수의 타입을 자동으로 결정하게 한다. storeReview에 대한 참조로 storeReview의 타입에 따라 타입이 결정된다.

- 코드 스크린샷

```
}
// 기능4 : 리뷰 작성하고 확인하기
case 2: {
  subChoice = -1;
  while (true) {
    std::cout << "₩n4. 리뷰 메뉴" << std::endl;
    std::cout << "========================================="
              << std::endl;
    std::cout << "1: 리뷰 작성하기" << std::endl;
    std::cout << "2: 리뷰 확인하기" << std::endl;
    std::cout << "0: 메인 메뉴 돌아가기" << std::endl;
    std::cout << "========================================="
              << std::endl;
    std::cout << "옵션을 선택해주세요: ";
    std::cin >> subChoice;

    if (subChoice == 0) break;

    switch (subChoice) {
      // 4.1 리뷰 작성하기
      case 1:
        std::cin.ignore();
        std::cout << "매장 이름을 입력하세요: ";
        std::getline(std::cin, storeName);
        std::cout << "리뷰를 입력하세요: ";
        std::getline(std::cin, review);

        // 입력한 매장이름과 일치하는 매장 정보 찾기
        for (auto& reviewObj : storeReviews) {
          if (reviewObj.getStoreName() == storeName) {
            reviewObj.writeReviewToFile(review);
            break;
          }
        }
        break;
```

- 함수 설명

  옷 가게 이름을 입력하면 리뷰를 작성하고, 그 작성한 내용을 파일에 저장하는 함수다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  함수를 정의했고, ofstream을 통해 파일을 연다. 파일이 성공적으로 열리면 전달받은 리뷰 내용을 파일에 작성한다. 작성이 완료되면 file.close를 사용하여 파일을 닫는다.

- 코드 스크린샷

<기능4. Cpp>

```cpp
// 리뷰 파일에 저장하는 함수
void StoreReview::writeReviewToFile(const std::string& review) {
    std::string filename = storeName + "_reviews.txt";
    std::ofstream file(filename, std::ios::app);
    if (file.is_open()) {
        file << review << "\n";
        file.close();
        std::cout << "리뷰가 파일에 저장되었습니다." << std::endl;
    } else {
        std::cout << "파일을 열 수 없습니다." << std::endl;
    }
}
```

(2) 리뷰 확인하기

- 입출력

  입력: 매장 이름

  출력: 매장 이름을 입력하라는 안내 메시지, 매장의 리뷰

- 설명

  사용자가 선택한 매장의 리뷰를 확인할 수 있다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  For 반복문을 사용하여 입력한 매장이름과 일치하는 매장 정보를 찾고, auto를 사용하여 변수의 타입을 자동으로 결정하게 한다. storeReview에 대한 참조로 storeReview의 타입에 따라 타입이 결정된다.

- 코드 스크린샷

```
// 4.2 리뷰 확인하기
case 2:
    std::cin.ignore();
    std::cout << "매장 이름을 입력하세요: ";
    std::getline(std::cin, storeName);

    // 입력한 매장이름과 일치하는 매장 정보 찾기
    for (auto& reviewObj : storeReviews) {
        if (reviewObj.getStoreName() == storeName) {
            reviewObj.showReviews();
            break;
        }
    }
    break;
```

- 함수 설명

  매장의 리뷰를 파일에서 읽어와 화면에 출력하는 함수다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

  Ifstream을 사용하여 입력 파일 스트림을 연다. If문을 통해 파일이 성공적으로 열리는지 확인한 후 성공적으로 열리면 각 줄을 읽어와 화면에 출력한다. 모든 리뷰를 다 읽으면 file.close를 통해 파일을 닫는다.

- 코드 스크린샷

```
// 파일에 있는 리뷰 확인하기
void StoreReview::showReviews() const {
    std::string filename = storeName + "_reviews.txt";
    std::ifstream file(filename);
    if (file.is_open()) {
        std::string line;
        std::cout << "리뷰 목록:" << std::endl;
        while (getline(file, line)) {
            std::cout << "- " << line << std::endl;
        }
        file.close();
    } else {
        std::cout << "리뷰가 없거나 파일을 열 수 없습니다." << std::endl;
    }
}

std::string StoreReview::getStoreName() const { return storeName; }
```

<기능.h>

-클래스 설명

매장에 대한 리뷰를 관리하는 기능을 제공하는 클래스이다.

-적용된 배운 내용

멤버변수 storeName을 저장하고 멤버함수 writeReviewToFile, showReview를 지정한다. getStoreName은 storeName의 값을 반환한다. Const는 객체의 상태를 변경하지 않음을 나타낸다.

- 코드 스크린샷

```
// 기능 4 : 리뷰 쓰고 확인하는 클래스
class StoreReview {
private:
    std::string storeName;

public:
    StoreReview(const std::string& storeName);

    void writeReviewToFile(const std::string& review);
    void showReviews() const;
    std::string getStoreName() const;
};
```

4. 테스트 결과

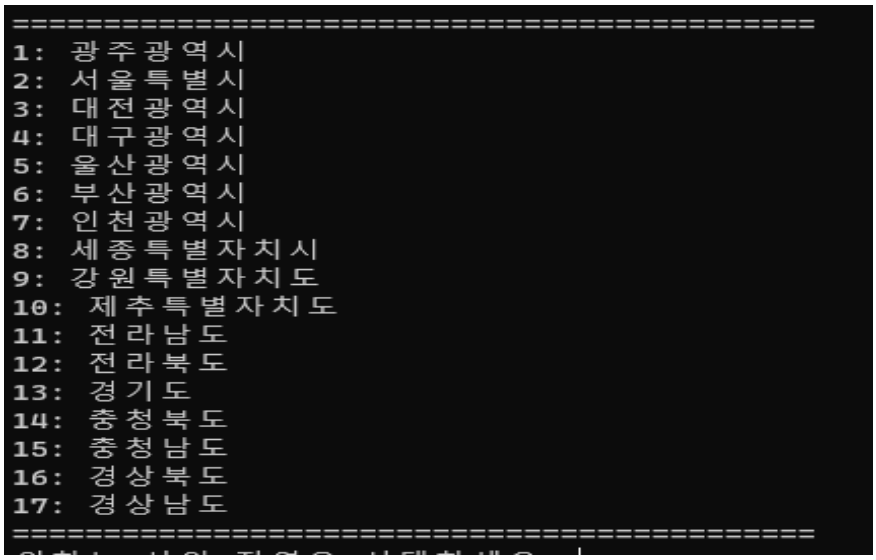
1. 기능1 테스트

(1) 상위지역 목록 출력하기

- 설명

상위 지역들의 이름을 담고 있는 문자열 벡터를 받아, 그 내용을 출력한다

- 테스트 결과 스크린샷

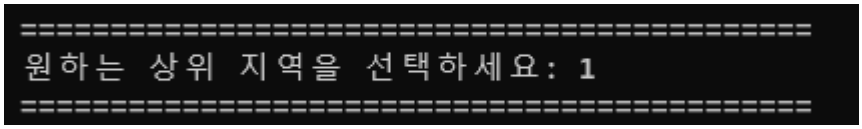


(2) 상위지역 입력 받기

- 설명

  사용자에게 상위 지역 인덱스를 입력 받아, 해당 인덱스를 반환한다.

- 테스트 결과 스크린샷



(3) 선택된 상위 지역에 대한 하위 지역 목록 출력

- 설명

  주어진 상위 지역에 해당하는 하위 지역들의 목록을 사용자에게 출력한다.

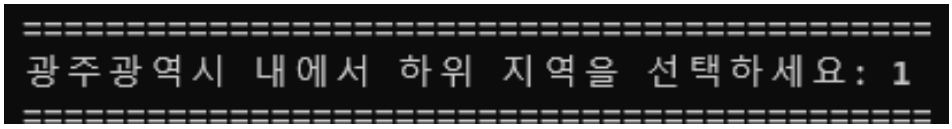
- 테스트 결과 스크린샷

(4) 하위 지역 입력 받기

- 설명

  사용자에게 하위 지역 인덱스를 입력 받아, 해당 인덱스를 반환한다.

- 테스트 결과 스크린샷



(5) 선택된 지역에 있는 옷가게들의 이름과 주소 출력

- 설명

  사용자가 선택한 지역에 대한 옷 가게 이름과 주소를 출력한다

- 테스트 결과 스크린샷

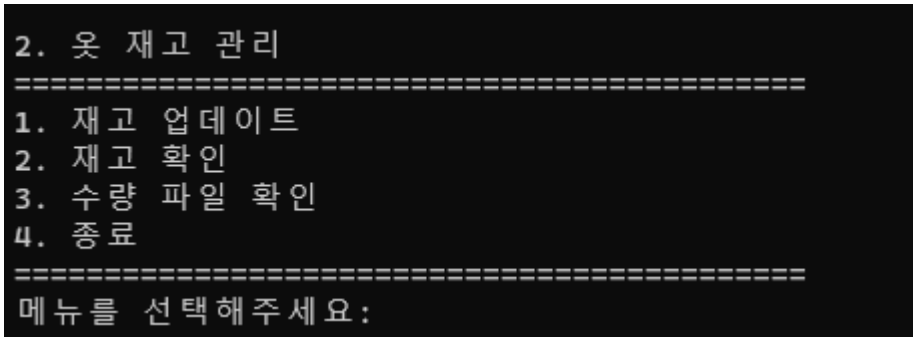
2. 기능 2 테스트

(1) 메뉴 선택하기

- 설명

 4가지의 메뉴리스트를 출력하고, 사용자는 사용하고 싶은 기능을 선택할 수 있다.

- 테스트 결과 스크린샷

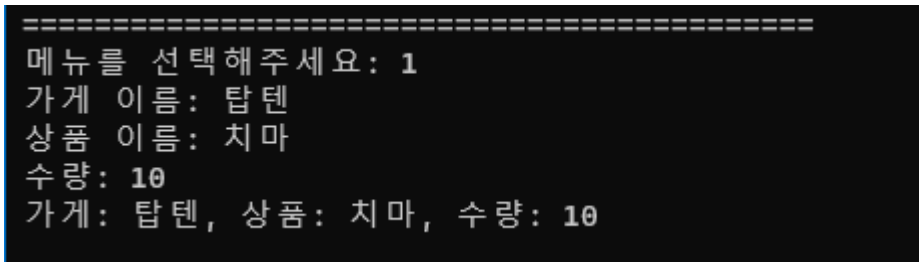


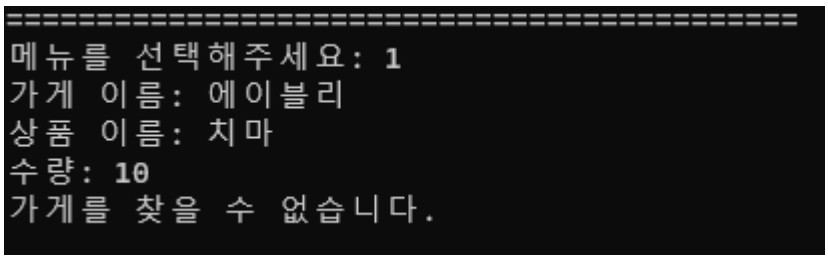
(2) 재고 업데이트

- 설명

  사용자로부터 가게이름, 상품 이름, 그리고 수량을 입력 받아 재고를 업데이트한다.
  사용자가 유효한 가게이름을 입력하지 않았을 경우, 오류 메시지를 표시하고, 유효한 입력인 경우에만 재고를 업데이트 할 수 있다.

- 테스트 결과 스크린샷



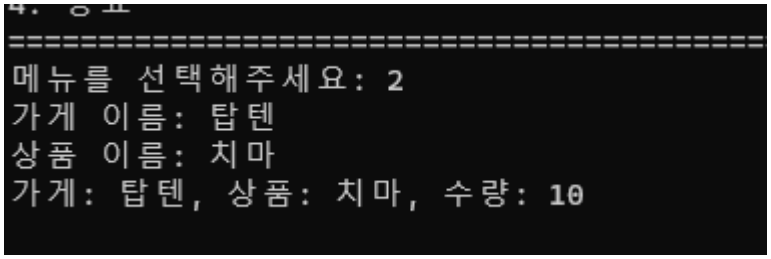


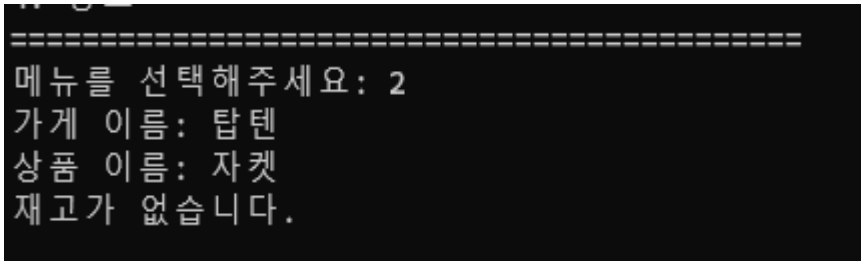
(3) 재고 확인하기

- 설명

  사용자로부터 가게 이름과 상품 이름을 입력 받아 해당 상품의 재고 상태를 확인한다.
  사용자가 유효한 가게 이름을 입력하지 않았을 경우, 적절한 오류 메시지를 표시하고, 유효한 입력인 경우에만 재고 확인을 수행한다.

- 테스트 결과 스크린샷



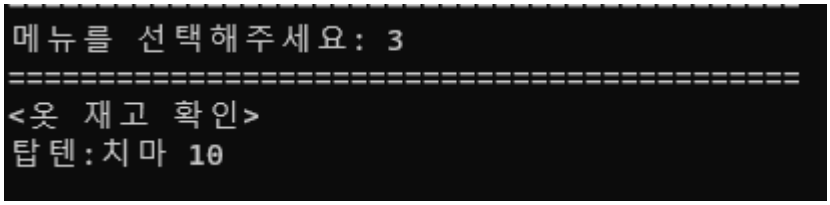


(4) 수량 파일 확인하기

- 설명

  수량.txt 파일의 내용을 확인하는 기능을 수행한다. 만약 파일 열기에 성공하면 파일의 내용을 한 줄씩 읽어 출력하고, 실패하면 오류 메시지를 출력한다.

- 테스트 결과 스크린샷

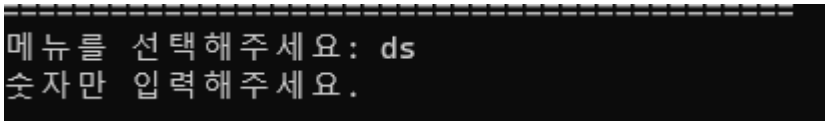


(5) 입력 값 오류 처리하기

- 설명

  사용자의 입력이 숫자인지 검증하는 함수이며, 숫자가 아닌 입력을 받았을 때 오류 메시지를 출력한다

- 테스트 결과 스크린샷



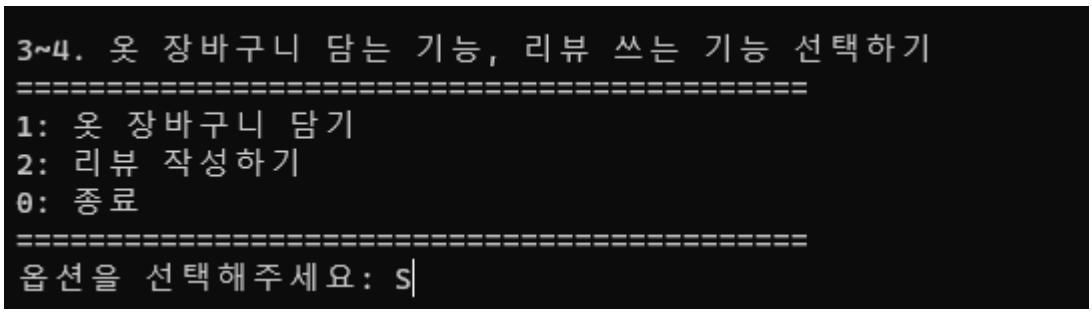
3. 기능 3 테스트

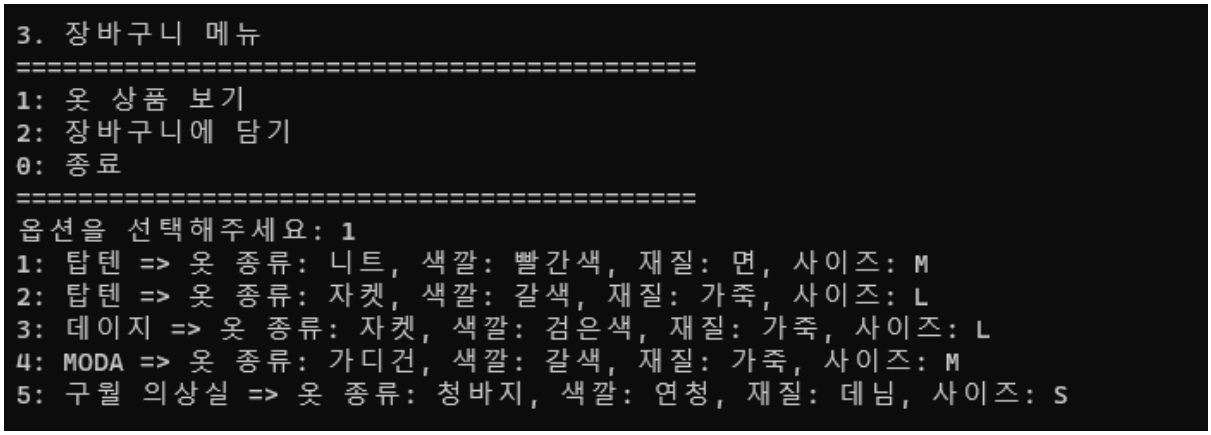
(1) 옷 상세정보 출력하기

- 설명

  사용자가 메뉴에서 옷 상품보기를 누르면 벡터에 저장된 옷 상품 정보를 순회하면서 각 상품의 상세정보를 출력한다.

- 테스트 결과 스크린샷



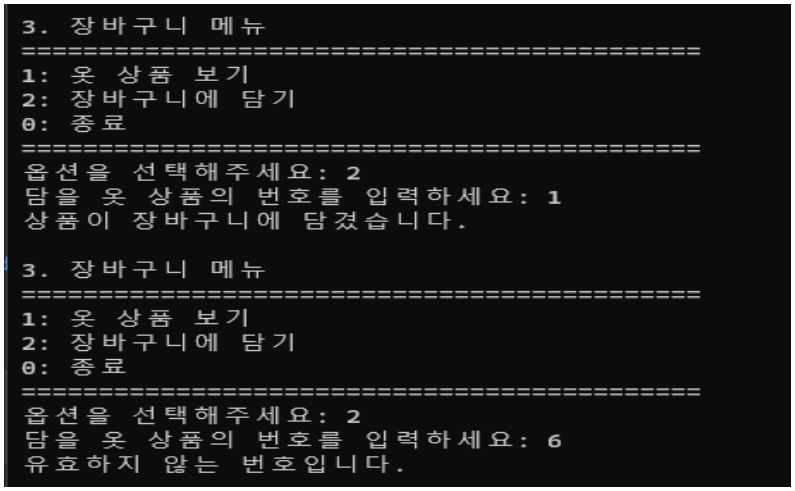


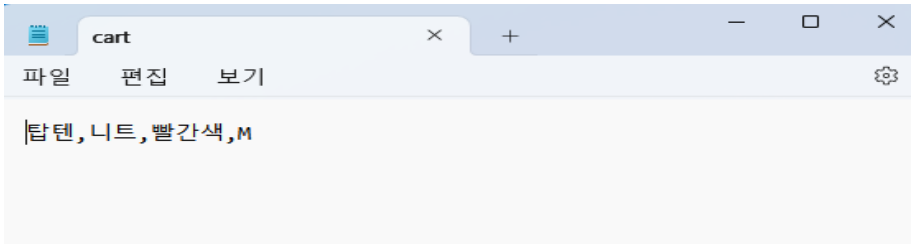
(2) 장바구니에 담기

- 설명

  사용자로부터 옷 상품의 번호를 입력 받아 해당 상품을 장바구니(파일)에 추가한다. 만약 입력한 번호가 유효한 범위를 벗어난 경우 유효하지 않는 번호라는 메시지를 표시한다.

- 테스트 결과 스크린샷





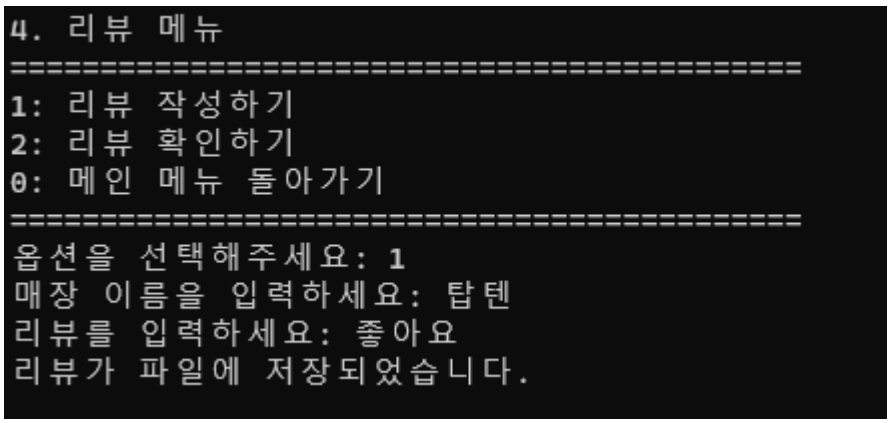
4. 기능 4 테스트

(1) 리뷰 작성하기

- 설명

  사용자가 1을 입력하면 매장을 입력하고, 리뷰를 작성할 수 있다.

- 테스트 결과 스크린샷

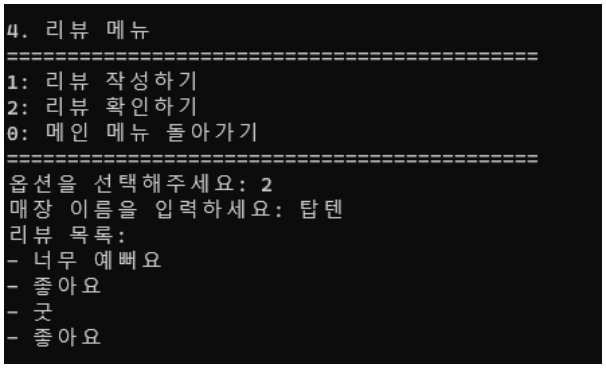


(2) 리뷰 확인하기

- 설명

  사용자가 선택한 매장의 리뷰를 확인할 수 있다.

- 테스트 결과 스크린샷



5. 계획 대비 변경 사항

- 전체적으로 기능을 조금 더 상세하게 나눴다.

1) 기능2 변경

- 이전

  관리자가 옷 수량을 입력하면, 사용자가 실제 수량을 알 수 있게 기능

- 이후

  관리자가 옷 수량을 입력하면, 옷 수량이 파일로 저장되어 관리하기 편하게 기능을 추가했다. 또한 사용자는 수량을 확인할 때 파일을 열어서 볼 수 있게 변경하였다.
  또한 수량과, 메뉴에 숫자가 아닌 문자열을 입력했을 경우 오류가 뜨기 때문에 숫자만 입력해주라는 메시지를 만들었다.

- 사유

  원래는 동적배열 벡터를 이용해서 수량을 관리하려고 했는데, 수업시간에 파일 입출력과 관련된 내용을 배우면서, 파일을 읽어 데이터를 객체에 저장하고, 객체들을 벡터에 저장 후 콘솔에 출력하는 코드를 실습으로 쳐본 적이 있다. 실습을 하면서 이번 기말 프로젝트에 이 내용을 적용하면, 사용자들이 조금 더 편하게 관리할 수 있을 거 같아서 이 기능을 추가하였다. 또한 숫자가 아닌 문자열을 입력할 때는 프로그램에 오류가 뜨기 때문에 오류를 막을 수 있는 기능도 추가했다.

2) 기능3 변경

- 이전

  "원하는 옷 장바구니에 담아두기" 기능
  원래는 이미지를 넣어서 그 이미지를 저장하는 방식으로 프로젝트를 만들 계획이었다.

- 이후

  옷 종류를 표시하고, 사용자가 특정 종류를 택하면 해당 종류의 옷에 대한 상세 정보 (색깔, 재질, 크기, 종류) 가 화면에 나타난다. 사용자는 원하는 옷을 선택하여
  그 정보를 파일에 저장하는 방식으로 기능을 수정하였다.

- 사유

이 프로젝트가 한 학기 최종 프로젝트인 만큼 수업 시간에서 배운 내용을 주로 활용하고자 한다. 이미지를 삽입하고 저장하는 부분은 수업 내용과는 무관하여, 이 프로젝트와는 어울리지 않는다고 판단하였다. 그래서 수업 시간에 학습한 내용을 기반으로 구현하기 위해 기능 3번을 변경하려고 한다.

6. 느낀점

먼저 프로젝트를 해본 게 처음이었기에 계획한 것들을 모두 이루지 못해 아쉬움이 남았던 거 같습니다. 특히 기능 2번에서 ios::trunc를 사용했던 부분에 대한 해결책을 찾기 위해 노력했지만, 아직 부족한 실력으로 인해 구현하지 못한 것이 안타까웠습니다. 프로젝트를 진행하면서 각 기능을 더 완벽하게 만들고자 하는 열망이 커져, 여러 시도를 해봤지만 오류가 계속 발생했고, 아직 그러한 문제들을 해결할 만큼의 실력이 부족함을 느꼈습니다.

그럼에도 불구하고, 혼자서 결과물을 만들어내고 보고서를 작성하는 과정은 매우 힘들었지만, 완료한 후에는 큰 뿌듯함을 느꼈습니다. 특히 보고서를 작성하며 수업에서 배운 내용을 다시 한 번 복습할 수 있었던 점은 매우 유익했습니다. 평소에 수업을 듣고 과제를 제출하면 금방 잊어버리는 경향이 있었는데, 이번에는 계속해서 코드를 보며 수업 내용을 상기시키니 내용이 더 오래 기억에 남았습니다.

솔직히 말씀드리면, 이번 프로젝트를 진행하며 ChatGPT의 도움을 많이 받았지만, 그 과정에서 발생한 문제들을 찾아내고 코드를 분석하면서 제 실력이 조금씩 성장하고 있다는 것을 느낄 수 있었습니다. 프로그래밍이 여전히 어렵고 부

족하다고 느껴지지만, 앞으로 더 많은 공부를 통해 프로그램을 완성도 있게 만들고 싶습니다.

한 학기 동안 수업을 들으면서 이렇게 열정적이고 학생들을 깊이 생각하는 교수님을 만나게 되어 영광이었습니다. 교수님의 열정적인 가르침 덕분에 많은 것을 배울 수 있었고, 수업에 더 열심히 참여하고자 하는 동기가 생겼습니다. 프로그래밍 실력이 뛰어난 다른 학생들 사이에서 때때로 위축되기도 했지만, 수업을 마친 지금은 많은 것을 배웠고, 앞으로 더 잘하고 싶다는 욕구가 생겼습니다. 이번 프로젝트를 완벽하게 마무리하지 못한 것에 대한 아쉬움이 크지만, 나중에 이 주제에 대해 더 발전시키고 싶습니다.

한 학기 동안 수고하신 교수님께 깊은 감사를 드립니다. 여태까지 만난 교수님들 중에서 학생들을 진심으로 생각해주는 것 같아 감사한 마음입니다. 교수님 덕분에 수업에서 많은 것을 배울 수 있었습니다.