

6조 전공기초프로젝트2(2040)

[단위검사 보고서]



조장 : 201511206 왕윤성

조원 : 201611186 김나경

201511197 방승희

제출일 : 2018년 11월 27일

[목 차]

1. 관리자 부분

1.1. 돌아가기 추가(유저 편의를 위한 수정)

2. 사용자 부분

2.1. 기계 잔돈 상황 출력 조건 변경(에러 수정)

2.2. 금고 용량 초과 시 잔돈 반환 방법 변경(에러 수정)

3. 단순 UI 변경

1. 관리자 부분

1.1. 돌아가기 추가(유저 편의를 위한 수정)

```
cout << "슬롯 번호>";  
cin >> select_num;  
//exception_test함수는 첫번째 매개변수가 올바른 값이면 true,  
input_test = exception_test(select_num, DRINK_MAX_SLOT, 1);
```

:이전 코드

```
cout << "*****1~10까지의 정수, 되돌아 가려면 11을 입력하세요*****" << endl;  
cout << "슬롯 번호>";  
cin >> select_num;  
//exception_test함수는 첫번째 매개변수가 올바른 값이면 true, 아니면 false를 리  
//여기서 슬롯번호 선택은 -> 최대 값은 11, 최소 값은 1 (11은 돌아가기를 위한 변  
input_test = exception_test(select_num, DRINK_MAX_SLOT + 1, 1);
```

:현재 코드

변경 위치 : Admin클래스의 int select(int choice)멤버 함수

슬롯 번호(1부터 10)를 입력받는 부분에서 11도 추가적으로 입력받게 설계함으로 뒤로
가기를 구현했다. select(int choice)함수를 사용하는 부분에서 어떻게 돌아가기를
구현했는지는 바로 밑에서 설명할 것이다.

```

//빈 슬롯을 입력받을 때까지 반복
while (1) {
    cout << "슬롯 번호>";
    cin >> slot_num;
    //슬롯 번호 예외처리 - 한글, 최대최소 거름
    if (exception_test(slot_num, DRINK_MAX_SLOT, 1)) {
        //입력한 슬롯이 등록된 슬롯이면 다시 입력 받음
        if (M->drink[slot_num - 1]._isRegist)
            cout << "!이미 등록된 슬롯 입니다! 슬롯 번호를 확인하세요!" << endl << endl;
        //입력한 슬롯이 빈 슬롯이면 while문 탈출
        else
            break;
    }
}

```

:이전 코드

```

int slot_num = A->select(0);
if (slot_num == 11) {
    cout << "돌아가기" << endl << endl;
    break;
}

```

:현재 코드

변경 위치 : main함수의 슬롯 등록 부분에서 슬롯 번호를 입력받은 직후


이전엔 이 부분에서 select함수를 사용하지 않았고, 11을 잘못된 입력이라고 판단한 반면에, 현재 코드는 select함수를 사용하고, 11을 입력했을 때 이전 메뉴로 돌아가게 설계했다.

```

case 2:
{
    //관리자의 select(0)에서 0의 역할은 슬롯을 받는다는 의미
    int slot_num = A->select(0);
    //슬롯 제거는 음료의 생성자로 초기화함
    M->_drink[slot_num - 1] = Drink();
    cout << endl;
}
break;

```

:이전 코드



```

case 2:
{
    //단위검사 : 돌아가기 구현
    //관리자의 select(0)에서 0의 역할은 슬롯을 받는다는 의미
    int slot_num = A->select(0);
    if (slot_num == 11) {
        cout << "돌아가기" << endl << endl;
        break;
    }
}

```

:현재 코드

변경 위치 : main함수의 슬롯 제거 부분에서 슬롯 번호를 입력받은 직후

슬롯 번호를 입력할 때, 이전 코드는 11을 잘못된 입력이라고 판단한 반면에, 현재 코드는 11을 입력했을 때 이전 메뉴로 돌아가게 설계했다.

```

case 3:
{
    M->show_drink();
    int slot, change;
    //cout << "슬롯 번호>";
    slot = A->select(0);

    if (M->_drink[slot - 1]._isRegist) {
        //cout << "변경할 재고 수>";
        //관리자의 select(1)에서 1의 역할은 재고를 받는다는 의미
        change = A->select(1);
        M->_drink[slot - 1].set_stock(change);
    }
    else
        cout << "미등록 슬롯" << endl;
}
break;

```

:이전 코드

```

case 3:
{
    //단위검사 : 변수 이름을 slot에서 slot_num으로 바꿈, 돌아가기 구현
    M->show_drink();
    int slot_num, change;
    //cout << "슬롯 번호>";
    slot_num = A->select(0);

    if (slot_num == 11) {
        cout << "돌아가기" << endl << endl;
        break;
    }
    else {

```

:현재 코드

변경 위치 : main함수의 재고 편집 부분에서 슬롯 번호를 입력받은 직후

슬롯 번호를 입력할 때, 이전 코드는 11을 잘못된 입력이라고 판단한 반면에, 현재 코드는 11을 입력했을 때 이전 메뉴로 돌아가게 설계했다. 또한 코드의 통일성을 위해 슬롯 번호를 입력받는 변수의 이름을 int slot에서 int slot_num으로 수정했다.

2. 사용자 부분

2.1. 기계 잔돈 상황 출력 조건 변경(에러 수정)

```
//잔돈 없음을 알려주는 창 의 역할 (이때 잔돈은 100원과 500원)
cout << "기계 잔돈 상황 : ";
if (M->_mmoney->_100 == 0 || M->_mmoney->_500 == 0)
    cout << "없음" << endl;           //금고 내 잔돈이 하나도 없는 상황
else
    cout << "있음" << endl;           //금고 내 잔돈이 있는 상황
```

:이전 코드

```
//단위검사 : 뜻밖의 에러!! ||을 &&로 바꿔줌
//잔돈 없음을 알려주는 창 의 역할 (이때 잔돈은 100원과 500원)
cout << "기계 잔돈 상황 : ";
if (M->_mmoney->_100 == 0 && M->_mmoney->_500 == 0)
    cout << "없음" << endl;           //금고 내 잔돈이 하나도 없는 상황
else
    cout << "있음" << endl;           //금고 내 잔돈이 있는 상황
```

:현재 코드

변경 위치 : main함수의 사용자 부분에서 기계 잔돈 상황을 출력해주는 곳

사용자에게 현재 기계의 기계 잔돈 상황을 알려주는 출력구문에서 프로그램 기획서에서 명시한 기계 잔돈 상황의 정의(기계 잔돈 상황은 자판기가 보유한 금고에 동전(100원, 500원)이 100원이 하나 있든, 500원이 하나 있든 하나라도 존재하면 있음이라고 표시한다.)에 따라 100원과 500원이 모두 없을 때 “없음”이라고 출력할 수 있게끔 조건문을 수정했다.

2.2. 금고 용량 초과 시 잔돈 반환 방법 변경(에러 수정)

```
//input의 값을 그대로 change로 옮기고(여러번째 거래일 경우 input에 이전 change가 남아있기때문)
M->change = M->input;
//change의 값을 user_input_money의 절대 값으로(이 경우에 Money의 파라미터 중 음수가 반드시 있으므로) 추가한다.
M->change->edit_money(new Money(abs(user_input_money->_100), abs(user_input_money->_500), abs(user_input_money->_1000)));
//돈을 돌려준다.
M->return_change();
break;
```

:이전 코드

```
//input의 값을 그대로 change로 옮기고(여러
M->change = M->input;

//change의 값을 user_input_money의 절대 값
//M->change->edit_money(new Money(abs(user
//돈을 돌려준다.
M->optimal_change(M->change->_total);
M->return_change();
break;
```

:현재 코드

변경 위치 : main함수의 사용자 부분에서 사용자가 투입한 돈 때문에 기계의 금고가 수용량을 초과할 경우 분기

사용자가 넣은 돈이 금고에 들어갔을 때 어떤 단위의 돈이 금고의 최대 보유량을 넘어서 금고에 들어갈 수 없는 상황일 때 이전 코드에서는 이러한 상황이 금고에 이미 어느 한 단위의 돈이 꽉찬 상태에서 일어났을 때를 가정하고(예를 들어 이미 금고에 500원이 50개 꽉 찬 경우 사용자가 500원을 넣는 경우)있어서 에러가 나는 반면(금고에 500원 48개가 있을 때 사용자가 500원 3개를 넣으면 거래를 중지하긴 하지만 금고에는 500원 48개가 아닌 48에서 3을 뺀 45개가 남아있다), 현재 코드는 모든 경우(금고에 500원이 48개 있을 때, 사용자가 500원 3개를 넣어서 터진 경우)를 고려해서 설계했다.

3. 단순 UI 변경



:관리자 메뉴 중 재고 편집에서 미등록 슬롯 입력 시

```
슬롯 번호>11
!반환 레버!

잔돈 : 3800원

동전류 잔돈 출구
100원 : 3개 500원 : 1개

지폐 투입구<지류 잔돈 출구>
1000원 : 3개
```

```
슬롯 번호>11
!반환 레버!

*****
잔돈 반환 : 3800원

동전류 잔돈 출구
100원 : 3개 500원 : 1개

지폐 투입구<지류 잔돈 출구>
1000원 : 3개
*****
```

:사용자 메뉴 중 11을 눌러 잔돈을 반환할 때

```

500원 개수>3
기계에 들어갈 수 있는 500원 동전 양을 초과하였습니다.

동전류 잔돈 출구
100원 : 3개 500원 : 3개
지폐 투입구<지류 잔돈 출구>
1000원 : 0개

잔돈 : 1800원

동전류 잔돈 출구
100원 : 3개 500원 : 3개
지폐 투입구<지류 잔돈 출구>
1000원 : 0개

```

```

500원 개수>3
!기계에 들어갈 수 있는 500원 동전 양을 초과하였습니다.!

*****
***** 초과된 돈***** 그대로 반환합니다.
동전류 잔돈 출구
100원 : 3개 500원 : 3개
지폐 투입구<지류 잔돈 출구>
1000원 : 0개

*****

잔돈 반환 : 0원

동전류 잔돈 출구
100원 : 0개 500원 : 0개
지폐 투입구<지류 잔돈 출구>
1000원 : 0개
*****

```

:사용자가 넣은 돈이 금고의 최대 보유량을 초과한 경우 이전까지 거래에 쓰이고 남은 돈과 현재 넣은 돈의 배출을 구분 짓기 위해 UI를 변경했다.