5월 31일 복습&퀴즈

이태양


```java
class Market {
    private ArrayList<String> userBuyList;
    private ArrayList<Integer> userBuyListStock;
    private String[] marketSellList = {"선풍기", "키보드", "마우스", "모니터"};
    private int[] marketSellListPrice = {380000, 80000, 70000, 400000};
    private int myMoney;
    private Boolean continueShopping;
    Scanner scan;
    final int DEFAULT_IDX = 1;
    int price = 0;
    public Market () {
        userBuyList = new ArrayList<String>();
        userBuyListStock = new ArrayList<Integer>();
        continueShopping = false;
        scan = new Scanner(System.in);
    }
    public void doShopping () {
        do {
            // 1. 마켓에서 판매하는 물품을 보여줌
            showMarketSellList();
            // 2. 구매할 물건을 선택하세요.
            //    어떻게 선택하지 ?
            //    키보드 입력이나 뭔가가 필요한거 같네 ?
            //    가만 보니 초기에 Scanner를 안만들었구나 추가!
            selectBuyItem();
            // 3. 구매리스트가 작성되었다면 비용 산정 진행
            doPayment();
            // 4. 계속 구매할 것인지 여부 판단
            checkContinueShopping();
            myWallet();
        } while (continueShopping);
```

필요한 변수들을 선언해주고

생성자에서 객체를생성

쇼핑을 위한함수 함수들을 순서대로 동작시킴 중재자역할을 하는 느낌? 지난 수업에 이렇게 코드를 짜고 중재자 라고 했던 말씀이 생각남


```java
private void checkContinueShopping () {
    Boolean isOK = false;
    do {
        System.out.print("쇼핑을 계속하시겠습니까 ? Y/N");
        String res = scan.nextLine();
        if (res.equals("Y")) {
            isOK = false;
            continueShopping = true;
        } else if (res.equals("N")) {
            isOK = false;
            continueShopping = false;
        } else {
            isOK = true;
            continue;
        }
    } while (isOK);
}
private void doPayment () {
    for(int i =0; i<userBuyList.size();i++){

        if(userBuyList.get(i)=="선풍기"){
            price += userBuyListStock.get(i)*marketSellListPrice[0];
        }else if(userBuyList.get(i)=="키보드") {
            price += userBuyListStock.get(i) * marketSellListPrice[1];
        }else if(userBuyList.get(i)=="마우스") {
            price += userBuyListStock.get(i) * marketSellListPrice[2];
        }else if(userBuyList.get(i)=="모니터"){
            price += userBuyListStock.get(i)*marketSellListPrice[3];
        }
    }
    System.out.println("총 결제 금액 : "+price);
```

쇼핑을 계속하는지 여부묻는 함수

N이 입력되기 전까지 반복문을 탈출하지 않음으로 다시 계속 물어봄

결제를 위해 가격을 계산해주려고했다 원하는 상품을 입력시 그 같은 위치의 인덱스로 개수를 뽑아올 수 있었지만, marketSellListPrice는 순서대로 값이 들어가 있어서, 구입하는 물품을 뒤부터 입력할 경우 같은 위치의 인덱스로 뽑아올수가 없어서 임의로 지정하는 방식으로 코드를 짰다


```java
public void myWallet() {
    boolean isTure = true;
    System.out.println("지갑에 넣을 금액을 입력해주세요 :");
    int num = scan.nextInt();
    while(isTure) {
        if (num < price) {
            System.out.println("잔액이 부족합니다 금액을 충전해 주세요 !");
            System.out.println("얼마를 더 충전하시겠습니까? ");
            num += scan.nextInt();
            System.out.println("충전이 완료되었습니다 현재 잔액은 :" + num + " 원 입니다.");
            if(num>price){
                isTure = false;
                System.out.println("결제 완료하였습니다 : 남은 금액은 : " + (num-price)+"원 입니다");
            }
        }
    }

}
```

지갑기능을 만들어본다고 시작했다

지갑에 넣은 금액이 부족하면 더 충전하게끔

더 충전해서 지갑금액이 결제금액보다 커지면 결제가 되는 느낌을 내봤다.

프로그램이 종료되면 잔액은 사라진다,,


```java
private void selectBuyItemStock (String selectItem) {
    Boolean isntErrorAmount = true;
    int amount;
    do {
        System.out.print("구매할 수량을 선택하세요: ");
        amount = scan.nextInt();
        if (amount <= 0) {
            System.out.println("잘못된 수량이니 다시 입력해주세요!");
            continue;
        }
        isntErrorAmount = false;
    } while (isntErrorAmount);
    createNonDuplicateBuyList(selectItem, amount);
}
```

1개 미만의 수량을 입력시 다시 입력하게 하기위한 함수

```java
private void selectBuyItem () {
    Boolean continueBuying = true;
    do {
        System.out.print("구매할 물건의 번호를 누르세요(결제진행: 0): ");
        int itemNum = scan.nextInt();
        if (itemNum > 4) {
            System.out.println("잘못된 물품을 선택하셨습니다!");
            continue;
        } else if (itemNum < 0) {
            System.out.println("잘못된 물품을 선택하셨습니다!");
            continue;
        } else if (itemNum == 0) {
            continueBuying = false;
            continue;
        }
        // 실제 물건의 구매 수량을 결정하기 전에 해당 물품을 구매하므로 ArrayList 설정이 필요하다.
        // 이제 해당 작업을 여기에 추가해봅시다 ~
        // 현재 케이스에서는 중복에 대한 대처가 진행되지 않고 있음
        // 그러므로 중복을 감지하여 리스팅을 할 수 있는 매서드를 만들 필요가 있다!
        //userBuyList.add(marketSellList[itemNum - DEFAULT_IDX]);
        //System.out.println(userBuyList);
        // 현재 createNonDuplicateBuyList()도 stock을 처리하고
        // 아래쪽의 selectBuyItemStock()도 stock을 처리한다.
        // 이렇게 혼동이 발생하는 경우에는 누가 더 우선권을 가져야 하는지 분석이 필요하다.
        // cNDBL(줄여서)은 실제 물건의 구매에 있어서 중복이 있는지 검사한다.
        // createNonDuplicateBuyList(marketSellList[itemNum - DEFAULT_IDX]);
        // 물품을 모두 선택하고 몇 개 구할지 결정하는 매서드
        selectBuyItemStock(marketSellList[itemNum - DEFAULT_IDX]);
        System.out.println(userBuyList);
        System.out.println(userBuyListStock);
    } while (continueBuying);
}
```

번호 잘못입력시 예외처리와 물품 구매시 출력화면 제공


```
    private void createNonDuplicateBuyList (String target, int amount) {
        // 실제 중복이 되었다면 인덱스 값이 나올 것이고
        // 중복이 없으면 -1이 나오게 될 것이다.
        int idx = userBuyList.indexOf(target);
        if (idx == -1) {    // 중복 없음
            userBuyList.add(target);
            userBuyListStock.add(amount);
        } else {            // idx가 중복된 요소를 알려줌
            // set(idx, 데이터)는 특정 인덱스의 값을 update(갱신)함
            // add(idx, 데이터) + remove(idx + 1)과 동일한 역할을 함
            userBuyListStock.set(idx, userBuyListStock.get(idx) + amount);
        }
    }
    private void showMarketSellList () {
        int length = marketSellList.length;
        System.out.println("우리 마켓에서 판매하는 물품을 리스팅 합니다!");
        for (int i = 0; i < length; i++) {
            System.out.printf("%d. %s: %d\n", i + 1, marketSellList[i], marketSellListPrice[i]);
        }
    }
}
public class Prob51 {
    public static void main(String[] args) {
        Market m = new Market();
        m.doShopping();
    }
}
```

주석참고..

판매물품을 보여주는 함수

객체생성 후 호출


```
우리 마켓에서 판매하는 물품을 리스팅 합니다!
1. 선풍기: 380000
2. 키보드: 80000
3. 마우스: 70000
4. 모니터: 400000
구매할 물건의 번호를 누르세요(결제진행: 0): 1
구매할 수량을 선택하세요: 1
[선풍기]
[1]
구매할 물건의 번호를 누르세요(결제진행: 0): 2
구매할 수량을 선택하세요: 1
[선풍기, 키보드]
[1, 1]
구매할 물건의 번호를 누르세요(결제진행: 0): 3
구매할 수량을 선택하세요: 4
[선풍기, 키보드, 마우스]
[1, 1, 4]
구매할 물건의 번호를 누르세요(결제진행: 0): 0
총 결제 금액 : 740000
쇼핑을 계속하시겠습니까 ? Y/N쇼핑을 계속하시겠습니까 ? Y/NN
지갑에 넣을 금액을 입력해주세요 :
700000
잔액이 부족합니다 금액을 충전해 주세요 !
얼마를 더 충전하시겠습니까?
50000
충전이 완료되었습니다 현재 잔액은 :750000 원 입니다.
결제 완료하였습니다 : 남은 금액은 : 10000원 입니다
```

이런식으로 프로그램이 진행되었습니다!