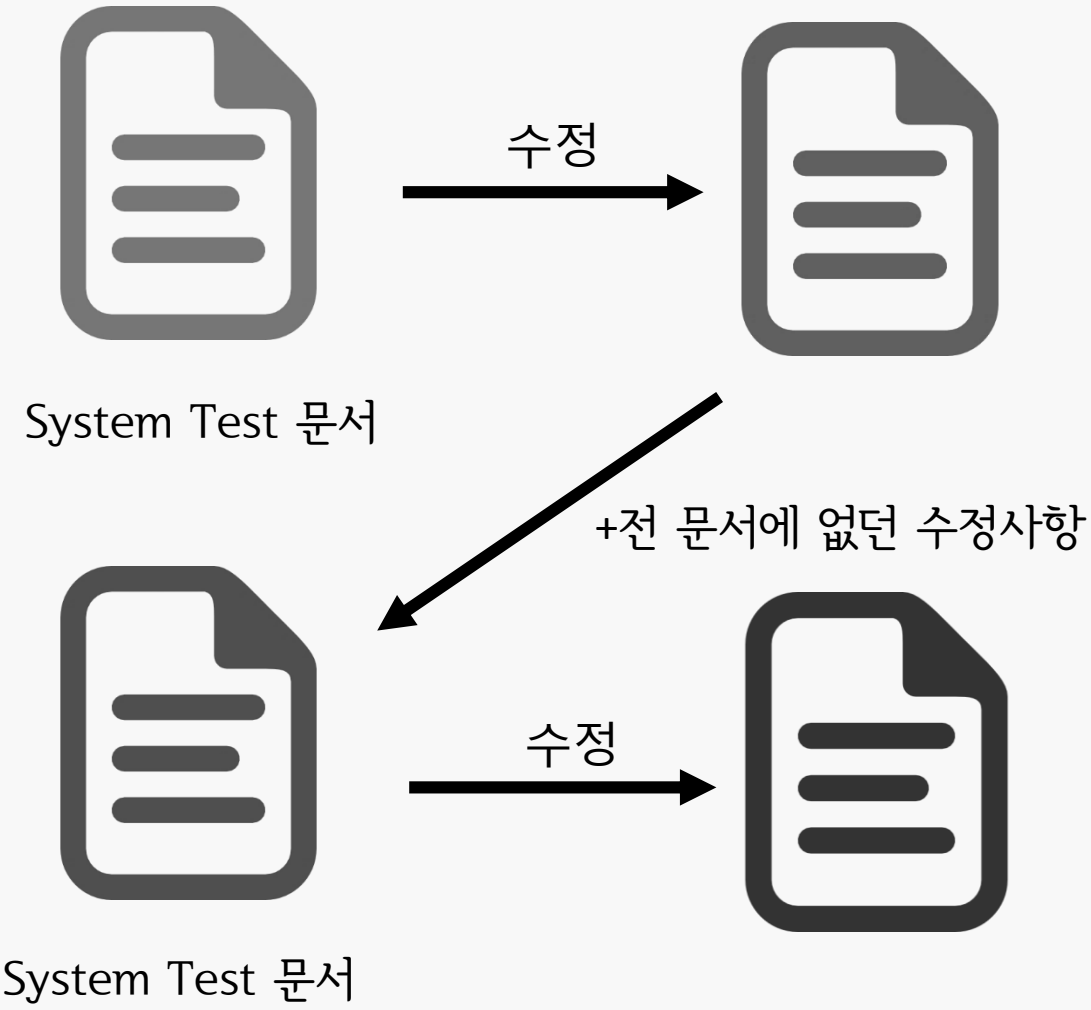
ATM

Team 1
201611293 전다윤
201311287 엄현식
201311318 최정헌

INDEX

1

문서수정


수정
System Test 문서
+전 문서에 없던 수정사항
수정
System Test 문서

문서수정

소프트웨어 검증팀의 문서대로 수정한 사항들

1008 Define Requirements

2031 Describe Use-Case
(check ,withdraw ,IssueTrafficCard)

2036 Define Operation
(selectService , insertCash)

2039 Traceability Analysis

2039 Design Class Diagram

2050 Implementation
(printReceipt , setDataRange, confirm , writeData
Add_link , selectNation)

2050 System Test 수정

일부만 수정한 /수정하지 않은 사항들

1004 RecordTerms in Glossary

1008 Business Concept Models

2031 Describe Use-case

(Update, Status Alarm)

2033 Domain Model

2034 Refine Glossary

2035 Define System seqene Diagrams

2036 Define Operation0 – end(void)

2041 Design Real Use-case

2144 Interaction diagram

(check , Deposit , withDraw , Transfer , Issue TrafficCard,

Management)

2051 Implementation clas&method

(insertCash , end , set_balance)

문서수정

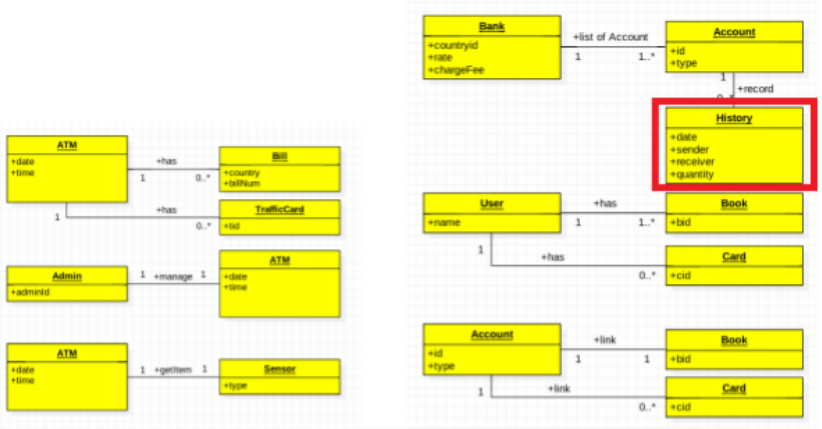
수정하지 않음 - 여기에 나온 내용이 꼭 나중에 나올 필요 없다.

1008. Define Business Concept Models

- 2030 단계에만 존재하는 심볼이 정의되어 있다. (History, Sensor)

2033. Define Domain Model

- 사용되지 않는 Sensor, Model, History를 사용한다.



문서수정

Interaction diagram (check , Deposit , withDraw)– 일부만 수정

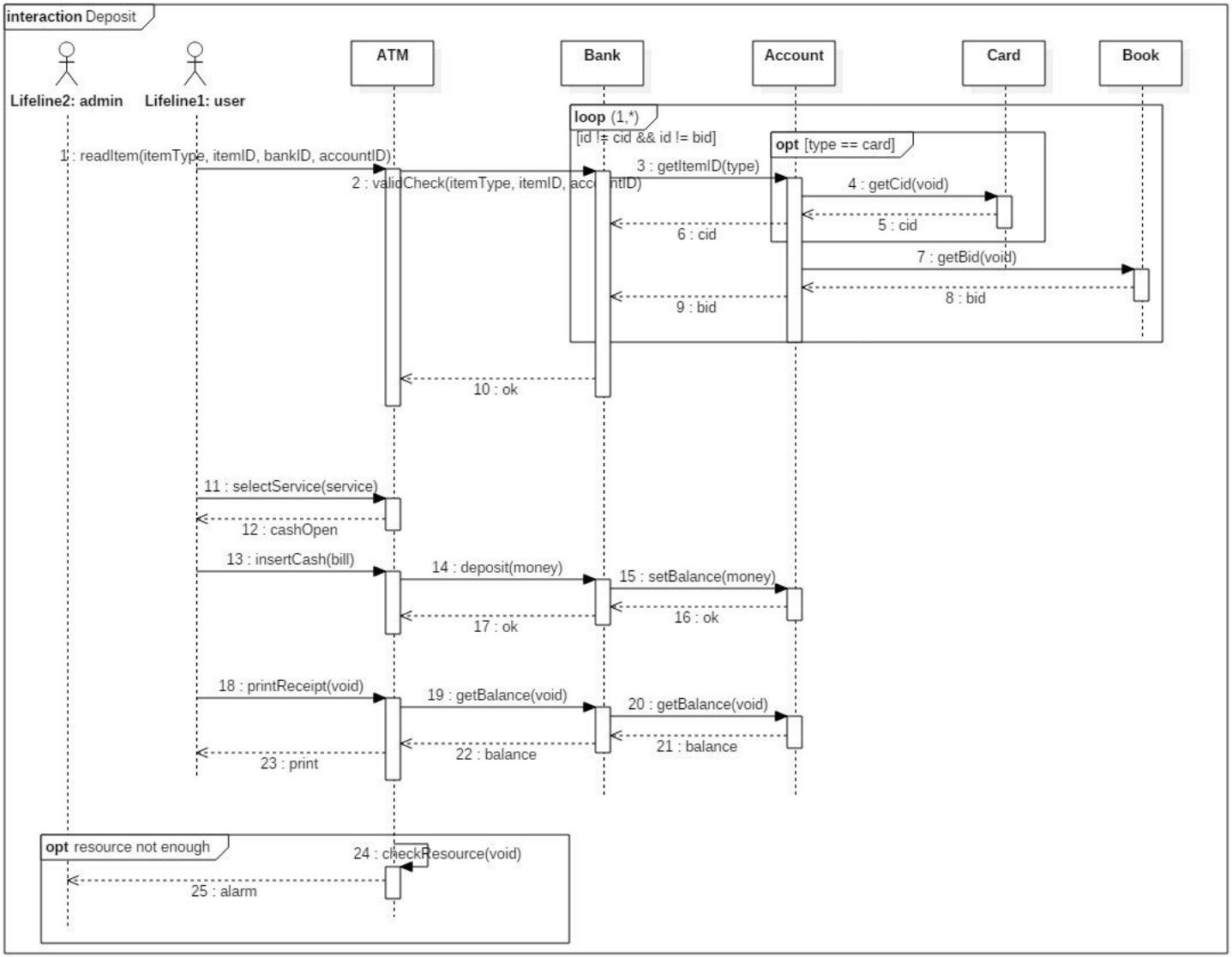
loop문 , opt 문 조건이 잘못 되었다 ⟶ 구체적으로 쓰지 않아도 된다

Sequence Diagram과 흐름이 다르다 ⟶ Sequence Digram 과 흐름이 같다

객체 값 바꾸기 ⟶ 수정

Deposit 순서가 바뀌어야 한다. ⟶ 수정

문서수정


interaction Deposit
Lifeline2: admin
Lifeline1: user
ATM
Bank
Account
Card
Book
loop (1,*)
[id != cid && id != bid]
opt [type == card]
1 : readItem(itemType, itemID, bankID, accountID)
2 : validCheck(itemType, itemID, acc ntID)
3 : getItemID(type)
4 : getCid(void)
5 : cid
6 : cid
7 : getBid(void)
8 : bid
9 : bid
10 : ok
11 : selectService(service)
12 : cashOpen
13 : insertCash(bill)
14 : deposit(money)
15 : setBalance(money)
16 : ok
17 : ok
18 : printReceipt(void)
19 : getBalance(void)
20 : getBalance(void)
21 : balance
22 : balance
23 : print
opt resource not enough
24 : checkResource(void)
25 : alarm

문서수정

Interaction diagram (Tranfer ,issueTrafficCard)– 일부만 수정

inputPassword를 통해 mode를 설정 하는 부분 X → Confirm method 존재

사용자가 명세표 출력을 선택하는 부분이 누락되어 있다. → printReceipt 존재

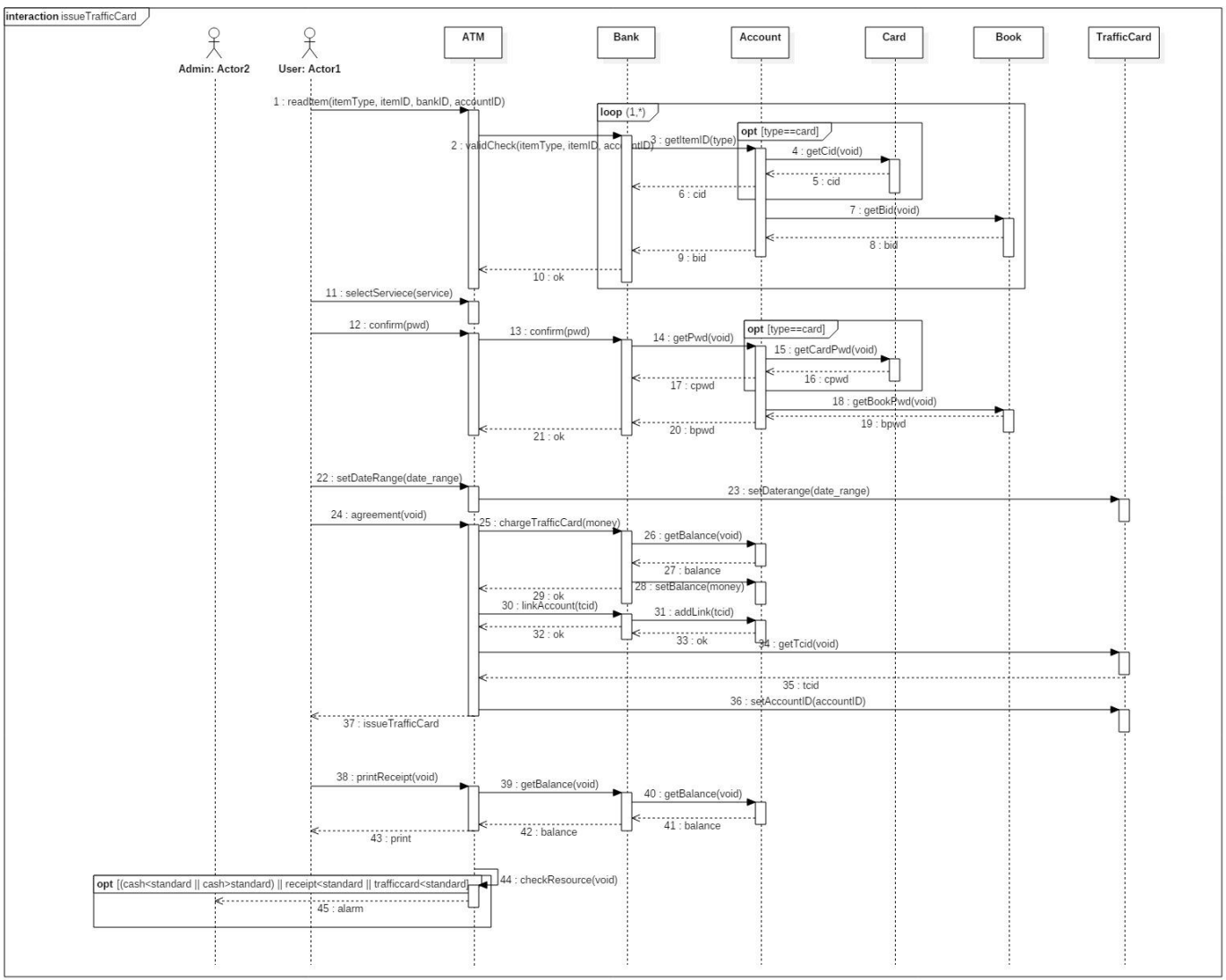
소프트웨어 검증팀 보고서에는 없었지만 , 전 diagram과
비슷한 형식이어서 , 수정한 부분을 똑같이 수정함

Interaction diagram (Management)– 수정 X

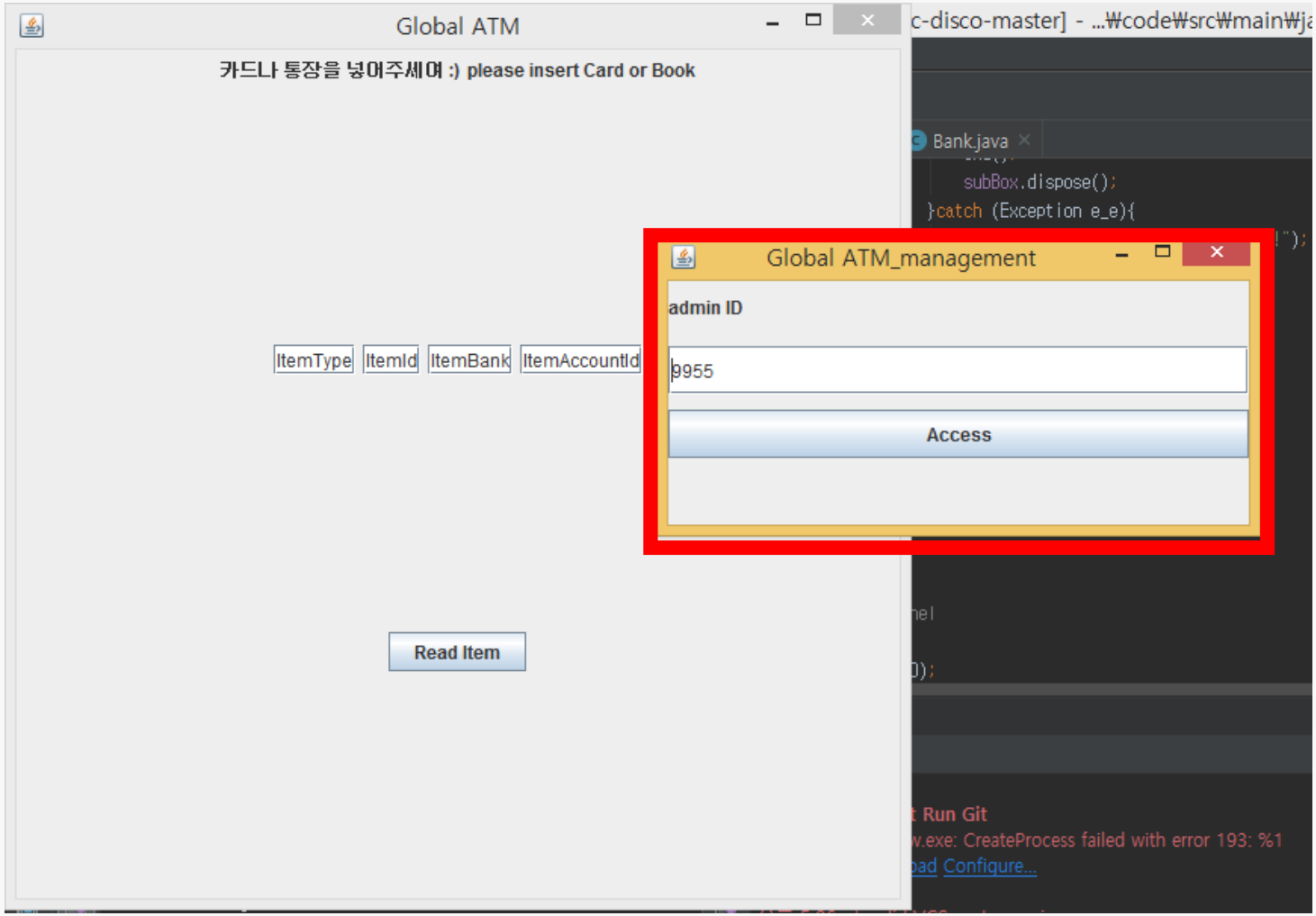
- 프로그램 상에서 존재하지 않는다. → 존재한다.

이부분에서 test하지 않은 것으로 보여진다.

문서수정


interaction issueTrafficCard
Admin: Actor2
User: Actor1
ATM
Bank
Account
Card
Book
TrafficCard
1 : readItem(itemType, itemID, bankID, accountID)
loop (1,*)
2 : validCheck(itemType, itemID, accountID)
3 : getItemID(type)
opt [type==card]
4 : getCid(void)
5 : cid
6 : cid
7 : getBid(void)
8 : bid
9 : bid
10 : ok
11 : selectServiece(service)
12 : confirm(pwd)
13 : confirm(pwd)
14 : getPwd(void)
opt [type==card]
15 : getCardPwd(void)
16 : cpwd
17 : cpwd
18 : getBookPwd(void)
19 : bpwd
20 : bpwd
21 : ok
22 : setDateRange(date_range)
23 : setDaterange(date_range)
24 : agreement(void)
25 : chargeTrafficCard(money)
26 : getBalance(void)
27 : balance
28 : setBalance(money)
29 : ok
30 : linkAccount(tcid)
31 : addLink(tcid)
32 : ok
33 : ok
34 : getTcid(void)
35 : tcid
36 : setAccountID(accountID)
37 : issueTrafficCard
38 : printReceipt(void)
39 : getBalance(void)
40 : getBalance(void)
41 : balance
42 : balance
43 : print
opt [(cash<standard || cash>standard) || receipt<standard || trafficcard<standard]
44 : checkResource(void)
45 : alarm

Management 부분 존재



문서수정

Analysis Traceability 수정

Operation in Sequence diagram	Operation in interaction diagram	Method	Class
1. readItem(account)	readItem(itemType, item ID, accountID)	readItem(itemType : int , itemID : int, bankID : int , accountID : int): i	ATM
2. selectService(service)	validCheck(itemType,itemID,accountID)	selectService(service : int): void	ATM
3. confirm(password)	getItemID(type)	selectNation(nation : int):int	ATM
4. insertCash(type,amount)	getCid(void)	confirm(itemType : int, pwd : int):int	ATM
5. selectMoneyType(type)	getBid(void)	insertCash(bill : String[1...*]):int	ATM
6. enterAmounts(amounts)	seleceService(service)	enterAmount(money : int):int	ATM
7. destAccount(bank,account)	confirm(itemtype ,pwd)	printReceipt(wants : boolean) : int	ATM
8. setDataRange(data_range)	confirm(pwd)	setDataRanger(data_range : int):void	ATM
9. agrrement(approval)	getPwd(void)	agreement() : boolean	ATM
10. printReceipt(print)	getCardPwd(void)	destAccount(bankID : int, accountID : int) : int	ATM
11. readManagementItem(id)	getBookPwd(void)	getAdminID():int	ATM
12. end()	insertCash(bill)	checkResource() : int	ATM
	deposit(money)	getBalance() : int	AYM
	setBalance(money)	end(): void	ATM
	selectNation(nation)		
	enterAmount(money)	validCheck(itemType : int,itemType : intemID : int) : int	Bank
	withDraw(money,accountID)	confirm(pwd:int) : int	Bank
	getBalance(void)	getBalance(tcid :int) : int	Bank
	setBalance(money)	checkAccount(bankID : int , accountID : int) : String	Bank
	destAccount(bankID,accountID)	linkAccount(tcid : int):String	Bank
	checkAccount(bankID,accountID)	transFer(money :int , accountDest : int ,accountSend : int) : int	Bank
	getName(void)	withDraw(money : int . accountID : int) : int	Bank
	setDateRange(date_range)	deposit (money : int , accountID : int) : int	Bank
	setDateRange(date_range)	getTcid() : int	TrafficCarc
	getTcid(void)	setDataRange(end_date : int) : void	TrafficCarc
	linkAccount(tcid)	setAccointID(accountID : int) :void	TrafficCarc
	addLink(tcid)	getCid() :int	Card
	chargeTrafficCard(money)	getCardPwd() :int	Card
	getBalance(void)	getBid () : int	Book
	setBalance(money)	getBookPwd() : int	Book
	setAccountID(accountID)	getitemID(itemType : int) : int	Account
	printReceipt(wants)	getPwd(itemType : int) :int	Account
	getBalance(void)	getBalance() : int	Account
	getAdminID(void)	setBalance(money : int) : void	Account
	end(void)	addLink(tcid : int): int	Account
	getBalance(void)	getName() : String	Account
	checkResource(void)		
	agreement()		

2

System Test Review

System Test Review

1. Path 설정 오류

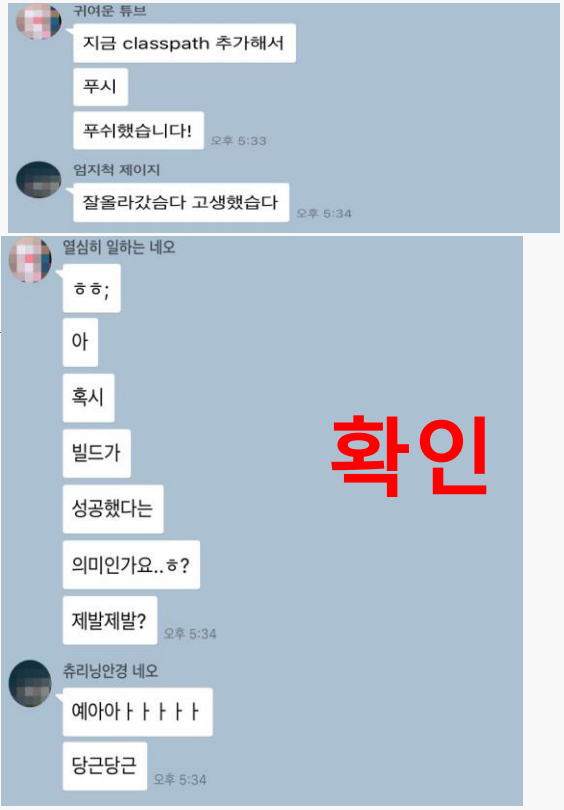
2. 알람 기능 오류

3. 예외처리

1. Path 설정 오류 - 지난 Test 에서도 나온 오류

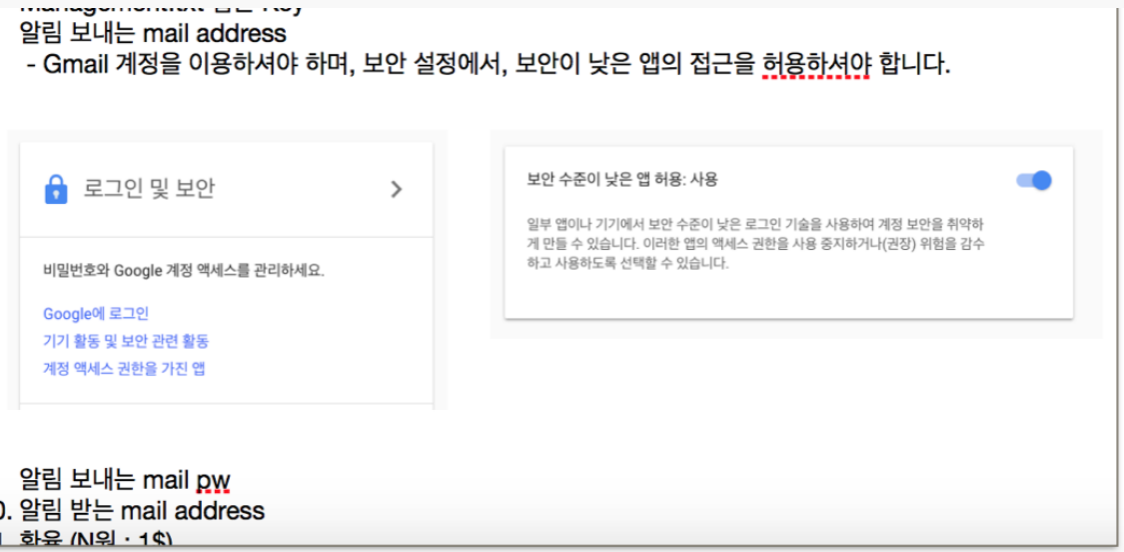
수정 완료

```
private boolean path = true;
//path_1 : IDE
private String path_1 = "code/src/main/java/ATM/management.txt";
//path_2 : .jar
private String path_2 = "management.txt";
//atm system
private ATM atm;
```



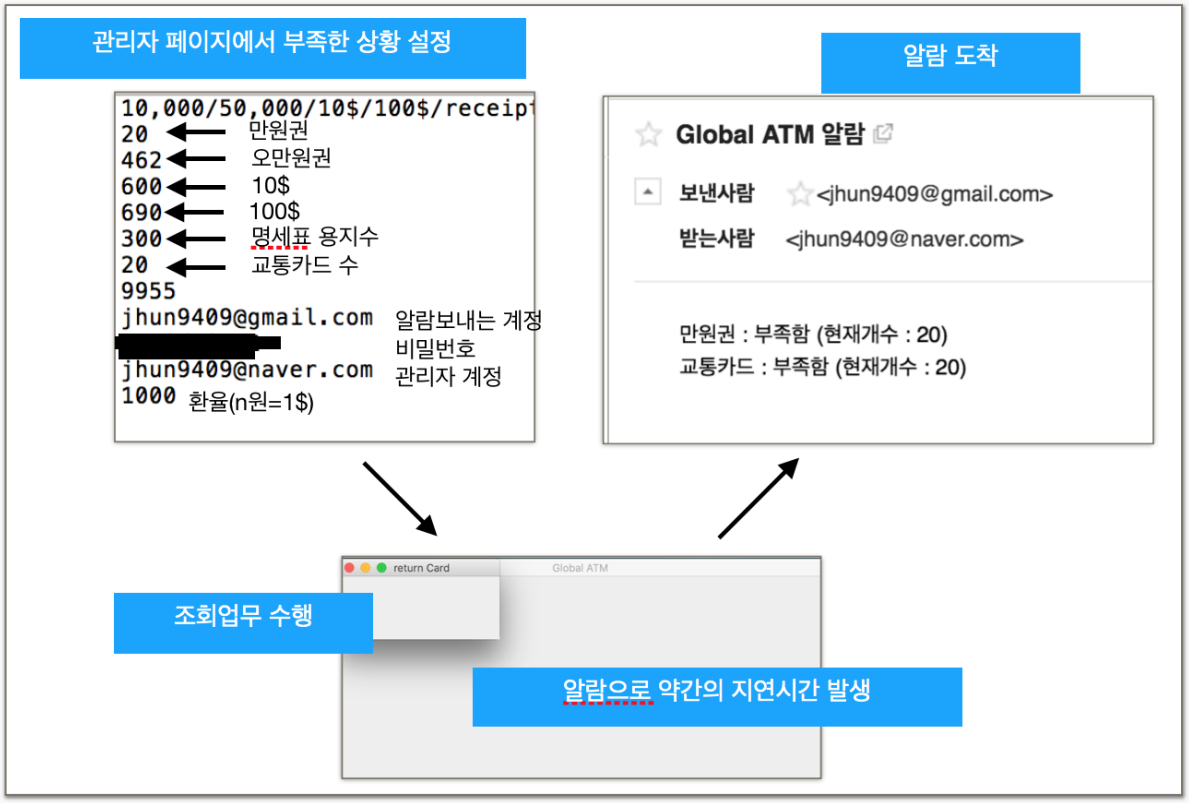
2. 알람 기능 오류 - SMTP 사용

8. 알림 보내는 mail address
   - Gmail 계정을 이용하셔야 하며, 보안 설정에서, 보안이 낮은 앱의 접근을 허용하셔야 합니다.

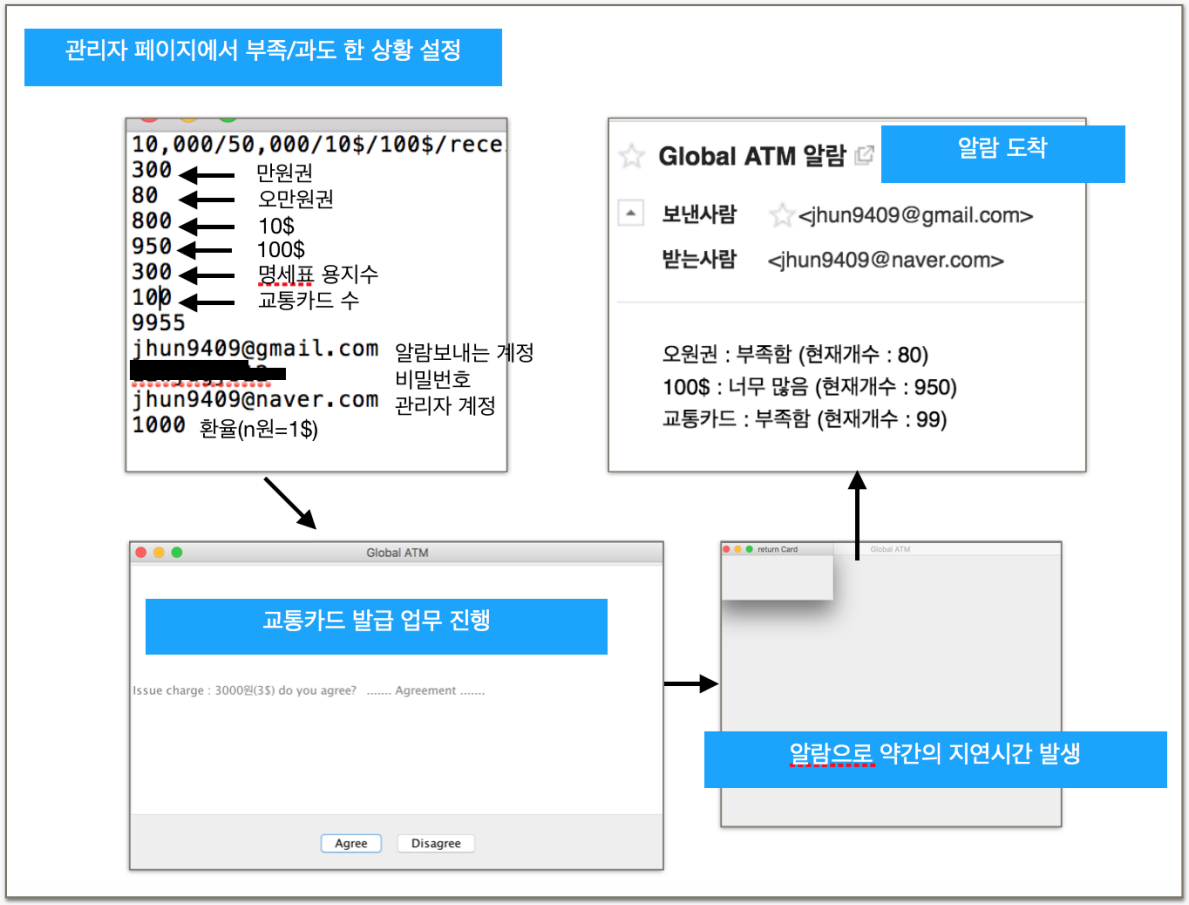


9. 알림 보내는 mail pw
10. 알림 받는 mail address
11. 활용 (N원 : 1$)

2. 알람 기능 오류 – 실행 결과



2. 알람 기능 오류 – 실행 결과



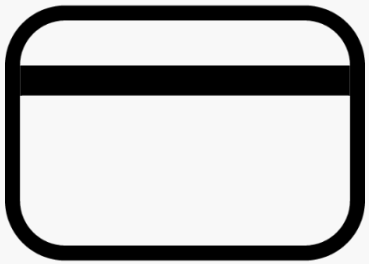
3. 예외 처리

- GUI 화면 출력 오류
- 여러 화폐가 동시에 입금되지 않는 오류
- 숫자 입력 시 불편함
- 만원 이하의 돈 , 10달러 이하의 돈은 처음부터 출금할 수 없다
- 카드발급 수수료 오류 (0.3$)
- 금액 인출

3. 예외 처리

- 카드발급 수수료 오류

  - 카드발급 수수료는 지난 문서에 의하면 0.3$ 여야 한다



3000원 => 3 $

3. 예외 처리

- 금액 인출
  - 인출 시 GUI에 보이는 금액 단위가 다르다

3

Static Analysis Review

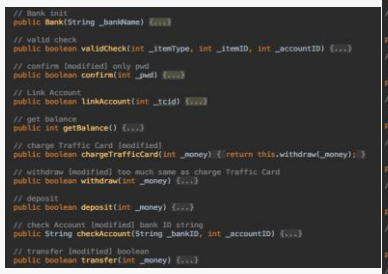
Static Analysis Review

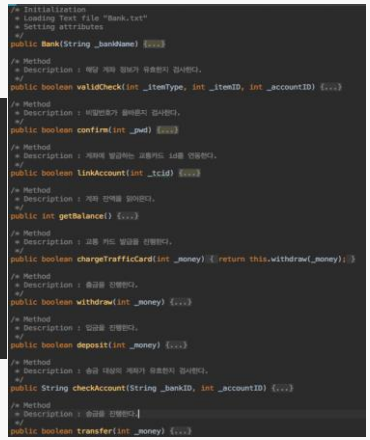
1. 주석 처리
2. Exception
3. 숫자 상수 사용
4. 스타일 개선
5. 스타일 개선
6. “*”를 이용한 import문
7. GUI

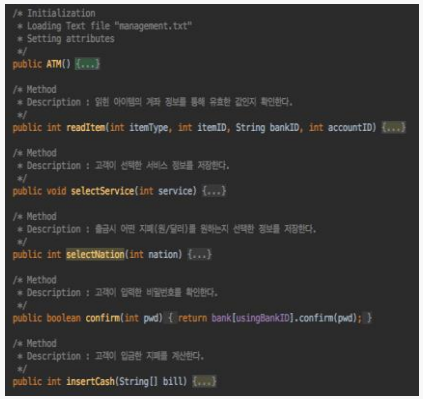
1. 주석 처리 - Class 주석

```
/* Book.java
 * : 통장 객체
 * : Account 마다 1개의 객체를 가지고 있음
 * : 은행정보와 계좌정보를 인자값으로 생성자메소드를 지니고 있음
 * : txt파일로 미리 저장된 정보를 읽고, 전달한다.
 */
```

1. 주석 처리 - Method 주석







2. Exceoption

catch 절에서 exception(java.io.FileNotFoundException)이 그대로 출력됨.	JAVA_21	높음	ATM.java	96	ATM.ATM()	ATM
메서드 (printStackTrace)가 사용됨.	JAVA_44	매우 높음	Book.java	50	Book.Book(String, int)	Item

모두 printStackTrace()로 대체

(오류 확인을 위한 콘솔 메시지 출력 유지)

3. 숫자 상수 사용

```
//input password
private char [] input_pw = new char[4];
private int pw_i;

//insert cash
private char [] input_cash = new char[3];
private int cash_i;
private String ptn_s;

//input Transfer accountid
private char [] input_id = new char[4];
private int id_i;

//input Date Range
private char [] input_date = new char[2];
private int date_i;
```

```
//input password
private char [] input_pw = new char[PWD_SIZE];
private int pw_i;

//insert cash
private char [] input_cash = new char[CASH_STRING];
private int cash_i;
private String ptn_s;

//input Transfer accountid
private char [] input_id = new char[AID_SIZE];
private int id_i;

//input Date Range
private char [] input_date = new char[TC_DATE_SIZE];
private int date_i;
```

```
//Size
private final int PWD_SIZE = 4;
private final int CASH_STRING = 3;
private final int AID_SIZE = 4;
private final int TC_DATE_SIZE = 2;
```

```
//parsing index Accountid  Bookid   Bookpwd
private final int PARSING_AID = 4;
private final int TAB_SIZE = 2;
private final int ID_SIZE = 4;
private final int PWD_SIZE = 4;
private final int PARSING_BID = PARSING_AID + TAB_SIZE + ID_SIZE;
private final int PARSING_BPWD = PARSING_BID + TAB_SIZE + PWD_SIZE;
```

```
//withdraw Cash type
private final int CASH_KOR = 0;
private final int CASH_ENG = 1;
private int cashNation;

//Fee
private final int FEE_KOR = 1000;
private final int FEE_ENG = 1;

//$ Type
private final int TEN_DOLLAR = 10;
private final int HUNDRED_DOLLAR = 100;
//won Type
private final int MAN_WON = 10000;
private final int OMAN_WON = 50000;
```

```
//Bank Index
private final int BANK_TYPE_NUM = 4;
private final int BANK_KB = 0;
private final int BANK_SH = 1;
private final int BANK_CT = 2;
private final int BANK_BA = 3;
private Bank[] bank = new Bank[BANK_TYPE_NUM];
private int usingBankID;

//ATM Item "Traffic Card"
private TrafficCard tCard;
private int trafficCardAmount;

//ATM Item "Cash"
private final int MAX_CASH = 1000;
private final int CASH_TYPE_NUM = 4;
private final int CASH_TYPE_MAN_WON = 0;
private final int CASH_TYPE_O_MAN_WON = 1;
private final int CASH_TYPE_TEN_DOLLAR = 2;
private final int CASH_TYPE_HUNDRED_DOLLAR = 3;
private int[] cashAmount = new int[CASH_TYPE_NUM];
```

```
//Index Num (Bank.txt)
private final int NAME_INDEX = 0;
private final int AID_INDEX = 1;
private final int BALANCE_INDEX = 2;
private final int TCID_INDEX = 3;
```


```
//Index Num (Bank.txt)
private final int NAME_INDEX = 0;
private final int AID_INDEX = 1;
private final int BALANCE_INDEX = 2;
private final int TCID_INDEX = 3;
```

3. 숫자 상수 사용

```
//withdraw Cash type
private final int CASH_KOR = 0;
private final int CASH_ENG = 1;
private int cashNation;

//Fee
private final int FEE_KOR = 1000;
private final int FEE_ENG = 1;

//$ Type
private final int TEN_DOLLAR = 10;
private final int HUNDRED_DOLLAR = 100;
//won Type
private final int MAN_WON = 10000;
private final int OMAN_WON = 50000;
```

```
//Bank Index
private final int BANK_TYPE_NUM = 4;
private final int BANK_KB = 0;
private final int BANK_SH = 1;
private final int BANK_CT = 2;
private final int BANK_BA = 3;
private Bank[] bank = new Bank[BANK_TYPE_NUM];
private int usingBankID;

//ATM Item "Traffic Card"
private TrafficCard tCard;
private int trafficCardAmount;

//ATM Item "Cash"
private final int MAX_CASH = 1000;
private final int CASH_TYPE_NUM = 4;
private final int CASH_TYPE_MAN_WON = 0;
private final int CASH_TYPE_O_MAN_WON = 1;
private final int CASH_TYPE_TEN_DOLLAR = 2;
private final int CASH_TYPE_HUNDRED_DOLLAR = 3;
private int[] cashAmount = new int[CASH_TYPE_NUM];
```

```
//parsing index Accountid  Bookid   Bookpwd
private final int PARSING_AID = 4;
private final int TAB_SIZE = 2;
private final int ID_SIZE = 5;
private final int PWD_SIZE = 4;
private final int PARSING_CID = PARSING_AID + TAB_SIZE + ID_SIZE; // 4 + 2 + 5 = 11
private final int PARSING_CPWD = PARSING_CID + TAB_SIZE + PWD_SIZE; // 10 + 2 + 4 = 17
```

```
//Transaction Mode
private final int READY_MODE = 0;
private final int CHECK_MODE = 1;
private final int DEPOSIT_MODE = 2;
private final int WITHDRAW_MODE = 3;
private final int TRANSFER_MODE = 4;
private final int ISSUE_MODE = 5;
private int transactionMode = READY_MODE;

//Account Nation (KOR / ENG)
private final int ACC_KOR = 0;
private final int ACC_ENG = 1;
private int accountNation;
```

4. 생성자 초기화

```
public ATM() {
    try {
        if(path){
            bootATM = new File(path_1);
        }
        else{
            bootATM = new File(path_2);
        }
        fr = new FileReader(bootATM);
        br = new BufferedReader(fr);

        //mention delete
        br.readLine();

        //declare getStr
        String getStr;
```

```
public ATM() {

    //declare getStr
    String getStr;
```

5. 스타일 개선 생성자 초기화

숫자 상수를 사용함(for loop의 counter로 - 1이나 0, 1을 사용하는 경우 제외)	Sun_24	매우 낮음	Book.java	18	Book.Book(String, int)	Item

```
//parsing index Accountid  Bookid   Bookpwd
private int PARSING_AID = 4;
private int TAB_SIZE = 2;
private int ID_SIZE = 4;
private int PWD_SIZE = 4;
private int PARSING_BID = PARSING_AID + TAB_SIZE + ID_SIZE;
private int PARSING_BPWD = PARSING_BID + TAB_SIZE + PWD_SIZE;
```

```
while((getStr = br.readLine()) != null) {
    //finding..
    if(getStr.substring(0, PARSING_AID).equals(String.valueOf(aid))) {
        bid = Integer.parseInt(getStr.substring(PARSING_AID + TAB_SIZE, PARSING_BID));
        bpwd = Integer.parseInt(getStr.substring(PARSING_BID + TAB_SIZE, PARSING_BPWD));
        break;
    }
}
```

6. “*”를 이용한 import문 사용

```
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.SwingConstants;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Desktop;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.io.File;
import java.io.IOException;
```

Summary

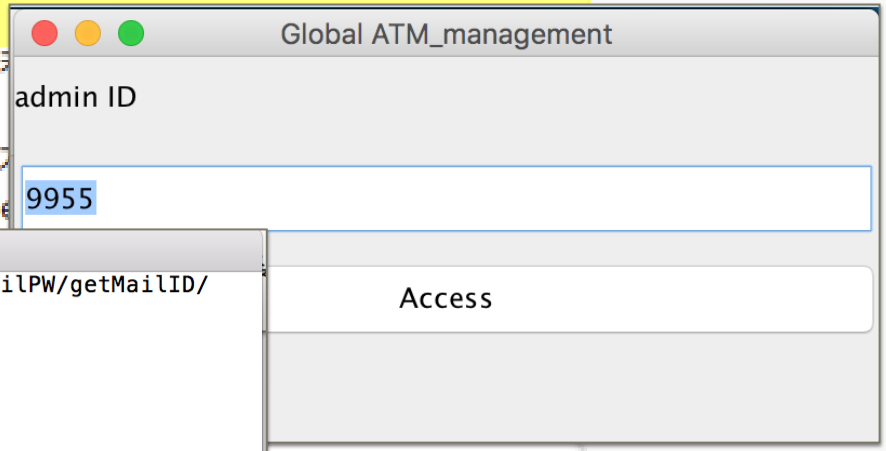
1. 문서들의 단계마다 일치하지 않는 부분들이 매우 많다.
   A. 2050단계의 경우, 모든 method의 Input과 Output칸을 비워놓는 등 최소한의 성의를 보이지 않았다.
2. 실제 구현된 코드와 문서 간에 굉장한 차이점이 존재한다.
3. 용어가 통일되지 않았다.
4. GUI class에서 상당 수의 연산을 수행해서 본래 그 연산을 수행했어야 할 class에는 빈 함수가 존재한다.
5. 객체 지향적인 설계를 하지 않았다.
6. 외화로 출금, 이체를 할 경우 자릿수가 바뀌어 연산 되는 등의 치명적인 버그들이 다수 존재한다.
7. Functional Requirement를 만족시키지 못하였다.
   A. 관리자와 같은 기능의 경우, 문서에는 존재하지만 코드상에서는 존재를 찾아볼 수 없다.
8. Non Functional Requirement를 모두 만족시키지 못하였다.
   A. 직관적이지 않은 UI
      i. 한글 깨짐 현상 발생 (파일과 코드 전체가 EUC-KR로 저장되어있다.)
      ii. PlaceHolder 처리가 되어있지 않다. (default text로 값이 들어가 있다.)
      iii. Maxlength가 지정되어 있지 않다.
      iv. Number format이 적용되어 있지 않다. (구분자가 없다.)
      v. Return Card가 닫지 않으면 계속 출력된다.
      vi. account id를 입력할 시 기존에 있던 label이 삭제되고 그 곳에 account id가 입력된다.

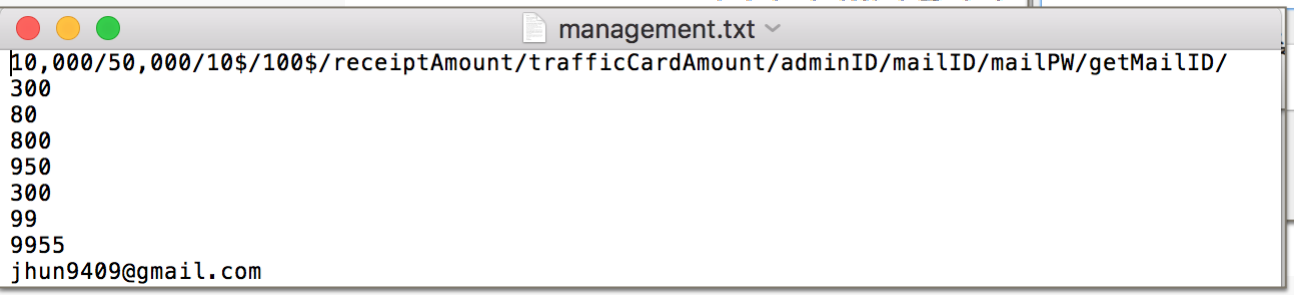
Summary

1. 문서들의 단계마다 일치하지 않는 부분들이 매우 많다.
   A. 2050단계의 경우, 모든 method의 Input과 Output칸을 비워놓는 등 최소한의 성의를 보이지 않았다.
2. 실제 구현된 코드와 문서 간에 굉장한 차이점이 존재한다.
3. 용어가 통일되지 않았다.
4. GUI class에서 상당 수의 연산을 수행해서 본래 그 연산을 수행했어야 할 class에는 빈 함수가 존재한다.
5. 객체 지향적인 설계를 하지 않았다.
6. 외화로 출금, 이체를 할 경우 자릿수가 바뀌어 연산 되는 등의 치명적인 버그들이 다수 존재한다.
7. Functional Requirement를 만족시키지 못하였다.
   A. 관리자와 같은 기능의 경우, 문서에는 존재하지만 코드상에서는 존재를 찾아볼 수 없다.
8. Non Functional Requirement를 모두 만족시키지 못하였다.
   A. 직관적이지 않은 UI
      i. 한글 깨짐 현상 발생 (파일과 코드 전체가 EUC-KR로 저장되어있다.)
      ii. PlaceHolder 처리가 되어있지 않다. (default text로 값이 들어가 있다.)
      iii. Maxlength가 지정되어 있지 않다.
      iv. Number format이 적용되어 있지 않다. (구분자가 없다.)
      v. Return Card가 닫지 않으면 계속 출력된다.
      vi. account id를 입력할 시 기존에 있던 label이 삭제되고 그 곳에 account id가 입력된다.

Summary

1. 문서들의 단계마다 일치하지 않는 부분들이 매우 많다.
   A. 2050단계의 경우, 모든 method의 Input과 Output칸을 비워놓는 등 최소한의 성의를 보이지 않았다.
2. 실제 구현된 코드와 문서 간에 굉장한 차이점이 존재한다.
3. 용어가 통일되지 않았다.
4. GUI class에서 상당 수의 연산을 수행해서 본래 그 연산을 수행했어야 할 class에는 빈 함수가 존재한다.
5. 객체 지향적인 설계를 하지 않았다.
6. 외화로 출금, 이체를 할 경우 자릿수가 바뀌어 연산 되는 등의 치명적인 버그들이 다수 존재한다.
7. Functional Requirement를 만족시키지 못하였다.
   A. 관리자와 같은 기능의 경우, 문서에는 존재하지만 코드상에서는 존재를 찾아볼 수 없다.
8. Non Functional Requirement를 모두 만족시
   A. 직관적이지 않은 UI
      i. 한글 깨짐 현상 발생 (파일과 코드 전체
      ii. PlaceHolder 처리가 되어있지 않다. (de





Summary

1. 문서들의 단계마다 일치하지 않는 부분들이 매우 많다.
   A. 2050단계의 경우, 모든 method의 Input과 Output칸을 비워놓는 등 최소한의 성의를 보이지 않았다.
2. 실제 구현된 코드와 문서 간에 굉장한 차이점이 존재한다.
3. 용어가 통일되지 않았다.
4. GUI class에서 상당 수의 연산을 수행해서 본래 그 연산을 수행했어야 할 class에는 빈 함수가 존재한다.
5. 객체 지향적인 설계를 하지 않았다.
6. 외화로 출금, 이체를 할 경우 자릿수가 바뀌어 연산 되는 등의 치명적인 버그들이 다수 존재한다.
7. Functional Requirement를 만족시키지 못하였다.
   A. 관리자와 같은 기능의 경우, 문서에는 존재하지만 코드상에서는 존재를 찾아볼 수 없다.
8. Non Functional Requirement를 모두 만족시키지 못하였다.
   A. 직관적이지 않은 UI
      i. 한글 깨짐 현상 발생 (파일과 코드 전체가 EUC-KR로 저장되어있다.)
      ii. PlaceHolder 처리가 되어있지 않다. (default text로 값이 들어가 있다.)
      iii. Maxlength가 지정되어 있지 않다.
      iv. Number format이 적용되어 있지 않다. (구분자가 없다.)
      v. Return Card가 닫지 않으면 계속 출력된다.
      vi. account id를 입력할 시 기존에 있던 label이 삭제되고 그 곳에 account id가 입력된다.

4

소감

감사합니다.