



System Test & Spec Review

-5팀-
이지영 (팀장)
강민호
김정연
유경원

INDEX

1. Spec Review
 2. Test Stub
 3. Category-Partition Testing
 4. Brute Force Testing
- 

01 Spec Review – Stage 1000

- 1003. Define Requirements

- Performance Requirements

- 자판기 간 통신 응답 시간이 1초 이내로 수행되어야 한다.
데이터를 확인하는 시간이 1초 이내로 수행되어야 한다.

3.3 Performance Requirements

- Network Message의 전송속도는 5초안에 이루어 지고 전송간 오류가 없어야 한다.
- 사용자가 screen에 입력 후 해당 기능이 실행되는데 걸리는 시간은 0.1s 이내여야 한다.

SRS 3.3 Performance Requirements 항목, 20p

☞ Q. 1초인가요? 0.1초인가요? SRS 문서에선 0.1초라고 되어있어 서로 말이 상충됩니다. 하나로 통일 부탁드립니다.
A. 1초로 통일하겠습니다.

OOPT 1000문서와 SRS문서 비교 통일되지 않은 기준

- 1004. Record Term in Glossary(p.7)

Verification Code	단어 설명
Stock	재고
Count	Item의 수량

💡 Stock과 Count이 문서상 같은 표현으로 사용되고 있습니다. 같은 의미를 혼용하고 있는데, Stock을 '품목(Item의 종류)'으로 고치거나 혼동을 줄이기 위해 한 가지 표현으로 통일하는 것이 좋습니다.
또한 Stock을 '재고의 수량'을 표현한 것이라면 "재고" 대신에 "재고량"으로 수정하셔야 합니다. '재고'는 물건을 나타내는 단어지 수량을 표현하는 단어가 아닙니다.

Stock / Count 의미가 혼동되는 단어 사용

01 Spec Review – Stage 1000

문장의 표현이 불명확한 부분

OOPT 1000 pg12

- Describe use cases
- 4. Show Item

Use Case	4. Show Item
Actor	None
Description	- 사용자가 현재 자판기의 재고 유무에 상관없이 구매 가능한 음료를 보여준다.

☞ "사용자가 현재 자판기의 재고 유무에 상관없이 구매 가능한 음료를 보여준다."
→ "사용자가 현재 자판기의 재고 유무에 상관없이 구매 가능한 모든 음료를 보여준다." 로 구체화하여 수정 부탁드립니다.

OOPT 1000 pg13

- 12. Inform Location

Use Case	12. Inform Location
Actor	None
Description	- 다른 자판기로부터 응답 msg가 도착한 후 실행되는 use case. - 사용자에게 선택한 음료를 판매하고 있는 자판기의 위치를 알려준다.

☞ "- 사용자에게 선택한 음료를 판매하고 있는 자판기의 위치를 알려준다" → "- 사용자에게, 선택한 음료를 판매하고 있는 모든 자판기의 위치를 알려준다" 로 수정 부탁드립니다.

OOPT 1000 pg19

Check Stock Count

legative Test Case

- 재고가 있는데 없다고 파악하는 경우, 다른 자판기의 재고 수를 요청한다

6	Check Stock Count	1. 기본: 실제 재고와 일치하는 값으로 저장해 다음단계를 진행한다.	1. 재고가 있는데 없다고 파악하는 경우: 다른 자판기의 재고 수를 요청한다.
---	-------------------	--	---

☞ Q. 이게 무슨 뜻인가요?

A. local에 재고가 없고 remote에 재고가 있는 경우를 나타냅니다.

☞ "자판기에 재고가 없지만, 다른 자판기에 재고가 있는 경우"로 수정 부탁드립니다.

문장의 표현이 모호하거나 정확하게 의미전달이 되지않는 경우 수정 요청

01 Spec Review – Stage 2030

이전문서와의 통일 및 모호한 표현 수정

- 13. Create Verification Code (p.10)

Cross Reference	System Functions: R2.10, R2.6 Use Case: "Advance Purchase"
Pre-Requisites	선택된 음료에 대한 결제가 완료됨
Typical Courses of Events	(A1): User, (A2): Manager, (A3): Other DVM, (S): System 1. (S): 새로운 선구매 코드를 생성한다.

- Pre-Requisites

💡 보다 정확하게 "선택된 음료에 대한 선결제가 완료됨."으로 수정 부탁드립니다.

- Typical Courses of Events

💡 1. (S): "새로운 선구매 코드를 생성한다." → "6자리 난수의 선구매 코드를 생성한다." 로 수정해야 합니다. OOPT 1000 문서에서 이미 언급된 내용이기 때문입니다.

현장구매와 선구매 결제의 명확한 구분 필요

- 12. Inform Location (p.9)

Cross Reference	System Functions: R2.9, R2.8, R4.1 Use Case: "Select Advance Payment", "Message Request"
Pre-Requisites	음료가 선택됨

- Pre-Requisites

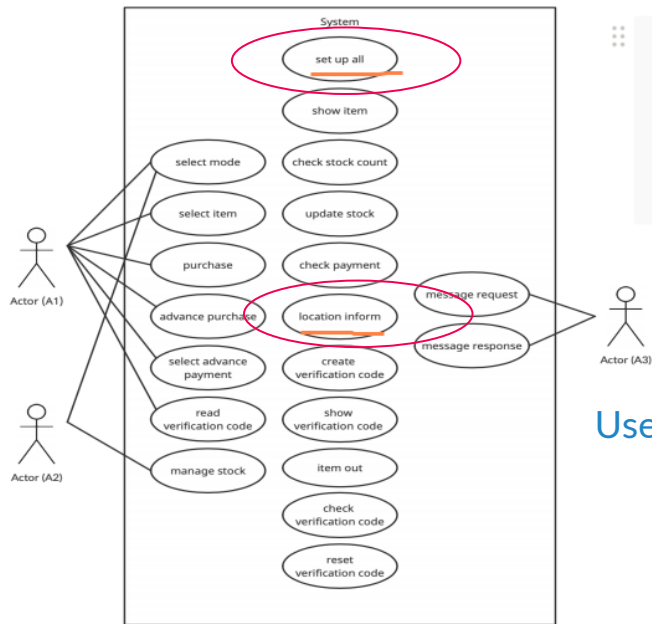
💡 위치보기 버튼을 누를 경우 실행되므로 "음료가 선택됨"이 아니라 "위치보기를 누름"으로 바뀌어야 합니다.

Pre-Requisites의 오류

01 Spec Review – Stage 2030

Use case 명칭오류

2132. Refine Use Case Diagrams



- (1) 'set up all'은 function 명입니다. 해당 function에 대응되는 UseCase 명칭은 "set up"이므로 수정 부탁드립니다.
- (2) 'location inform'은 function 명입니다. 해당 function에 대응되는 UseCase 명칭은 "inform location"이므로 수정 부탁드립니다.

Use case 명칭과 function의 이름이 혼동되어 사용되고 있는 부분

* (A1): User, (A2): Manager, (A3): Other DVMS

01 Spec Review – Stage 2040

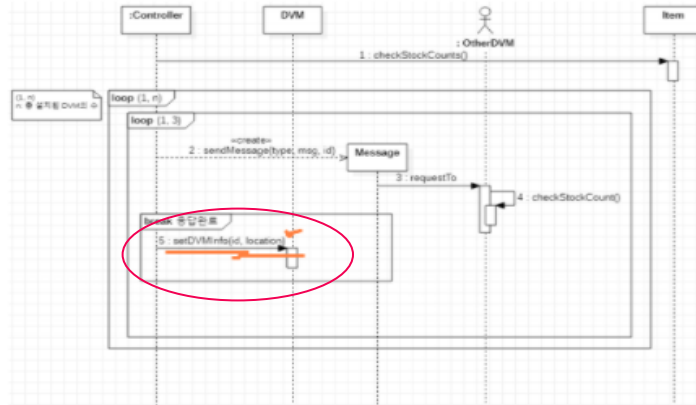
Interaction Diagram

2143. Define Interaction Diagrams

- (1) Set up(pg 19)

2143. Define Interaction Diagrams

(1) Set Up



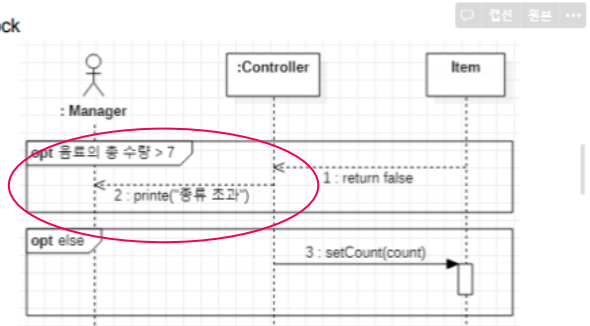
💡 "setDVMInfo" 오퍼레이션은 DVM의 id, location 그리고 item의 count 항목을 업데이트하는 것이므로 매개변수로 item도 같이 넣어야 합니다.

⇒ "setDVMInfo(id, location)" → setDVMInfo(id, location, item)

매개변수 누락

- (7) Update Stock (pg 22)

(7) Update Stock



🔴 Q. 음료의 총 수량이 7초과일경우 종류 초과 라는 에러 메시지를 생성한다고 되어있는데 "음료의 총 수량"이라는 의미는 음료의 종류가 7가지를 넘어간 경우인가요/음료의 재고의 수가 7개를 넘어간 경우인가요?

A. 음료의 종류의 가짓수가 7개를 초과한 경우입니다.

💡 음료의 종류의 가짓수가 7개 초과한 경우라고 의미가 분명하게 나타나게 수정바랍니다.

의미전달이 정확하지 않은 부분

01 Spec Review

3학년 개발팀 피드백

기타 의문

- Q. 선결제 코드는 어떻게 관리되나요?
A. 특정 시점에 한가지 음료에 대해서 선결제 코드가 여러개가 있는 경우도 존재하며 선결제 코드는 배열에 들어가 있습니다.
- Q. User는 1명으로 가정하나요, 여러 명으로 가정하나요?
A. User는 여러명이며 식별할 필요는 없습니다.
- Q. PFR에는 VM당 음료의 종류의 개수가 7개로 고정인 것 같은데 B1팀은 유동적으로 음료의 종류가 변할 수 있도록 할 것인가요?
A. 네 음료의 종류는 변합니다. 판매되는 음료의 종류의 가짓수가 7개 이하인 경우 새로운 음료가 추가될 수 있습니다.
- Q. VM ID 관리는 어떻게 하시나요?
A. 현재 로컬 자판기의 ID는 0으로 고정되었고 VM ID가 IP주소를 대체합니다.
- Q. 네트워크 통신 라이브러리를 사용해서 실제로 분산 시스템을 만들 것인가요, 아니면 네트워크 통신 없이 분산 시스템처럼 보이는 단일 프로그램을 만들 것인가요?
A. 네트워크 통신을 사용하지 않습니다.
- Q. 자판기는 총 몇 대인가요?
A. 총 5대입니다.
- Q. COPT 1000 FR Test Case에서 어떤경우 positive/negative 가 되나요?
A. use case의 목적에 맞으면 positive, 아니면 negative test case 가 됩니다

Spec Review를 진행하면서 생기는 의문에
대해서
3학년팀과 피드백을 주고받고 답변을 얻음

01 Spec Review - Trello

The image shows a Trello board interface. On the left is a board titled '이슈 현황' (Issue Status) with a blue header and a list of cards. The cards are: '이슈 현황' (UPCOMING), '[작성 예시] 화면이 여러개 출력되는 문제 디버깅' (May 20), '1000. Spec Review', '2030. Spec Review', and '2040. Spec Review'. On the right is a detailed view of the '1000. Spec Review' card. The card description reads: '1000 OOAD 문서 Spec Review 입니다. 하단 링크를 확인하시고 수정 진행하시면 될 것 같습니다.' Below the description is an attachment titled 'spec Review 통합본'. A checklist is visible with items: '1001. Non Functional Requirements 수정', '1003. Define Requirements 수정', '1004. Record Term in Glossary 수정', '1006. Define Business Use Case Diagram 동기화', '1006. Define Business Use Case - Identify use case 수정', '1006. Describe use cases 수정', '1006. 플로우 차트 다시 추가', '1009. Refine Plan - Functional Requirement Test Case 수정', '1009. JDK 버전 기입하기', and '1009. User Interface requirements 수정'. The right sidebar contains various Trello features like 'SUGGESTED', 'ADD TO CARD', 'POWER-UPS', 'BUTLER', and 'ACTIONS'.

spec review 진행 후
수정이 필요한 부분
트렐로에 issue로 등록하여
체크리스트로 진행 사항을 관리

02 Test Stub

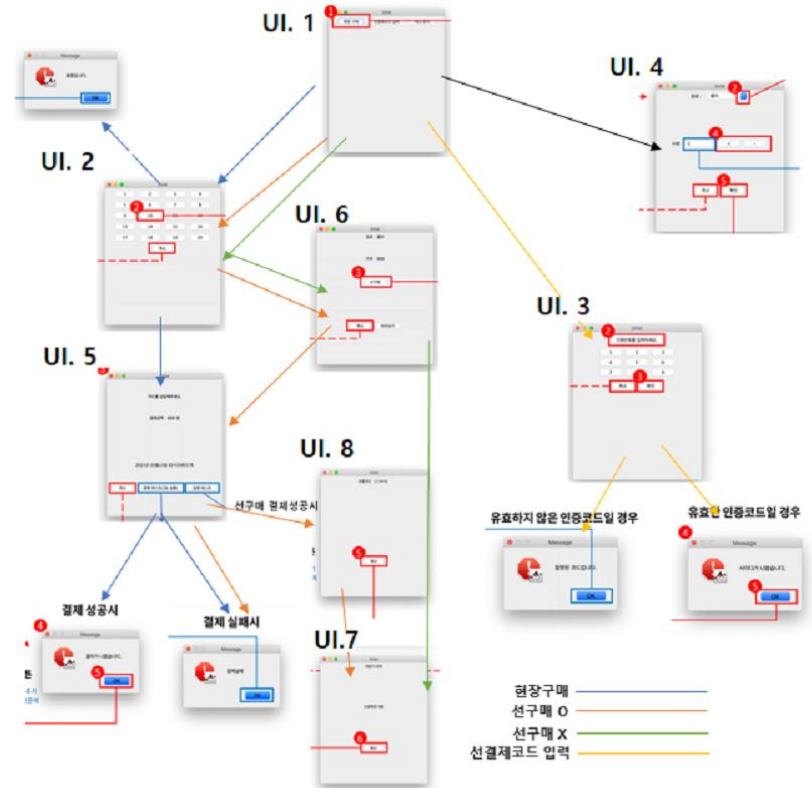
가상 Stub(매크로) 제작 방법

- 다음의 2가지 input event를 자동화 시키는 프로그램 작성

1. 마우스로 버튼 클릭
2. 콤보박스에서 마우스로 특정 항목 선택

- 결과적으로 다음 5가지 flow가 자동화 됨

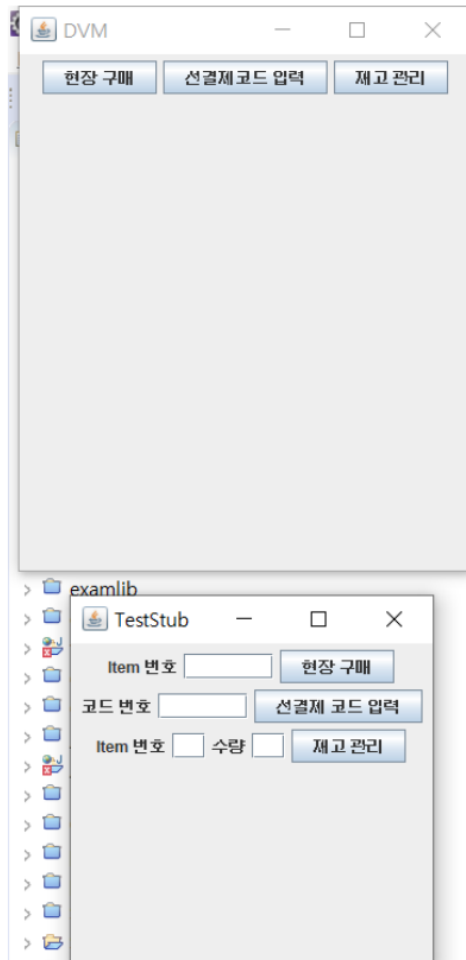
1. 파랑색 화살표에 해당하는 flow (현장구매)
2. 빨간색 화살표에 해당하는 flow (선구매)
3. 노란색 화살표에 해당하는 flow (선구매 코드입력)
4. 아래 그림에는 없는 flow (품질)
5. 아래 그림에는 없는 flow (재고 관리)



02 Test Stub

실행 방법

1. 현장 구매 버튼
 - 자동화 대상: 현장 구매 / 선구매 / 품절
 - 'Item 번호' input 필드에 1~20 사이의 값 입력
 - '현장 구매 버튼' 클릭
2. 선결제 코드 입력
 - 자동화 대상: 선결제 코드 입력
 - '코드 번호' input 필드에 선결제 코드 입력
 - '선결제 코드 입력' 버튼 클릭
3. 재고 관리
 - 자동화 대상: 재고 관리
 - 'Item 번호' input 필드에 1~20 사이의 값 입력
 - '수량' input 필드에 정수 입력
 - '재고 관리' 버튼 클릭

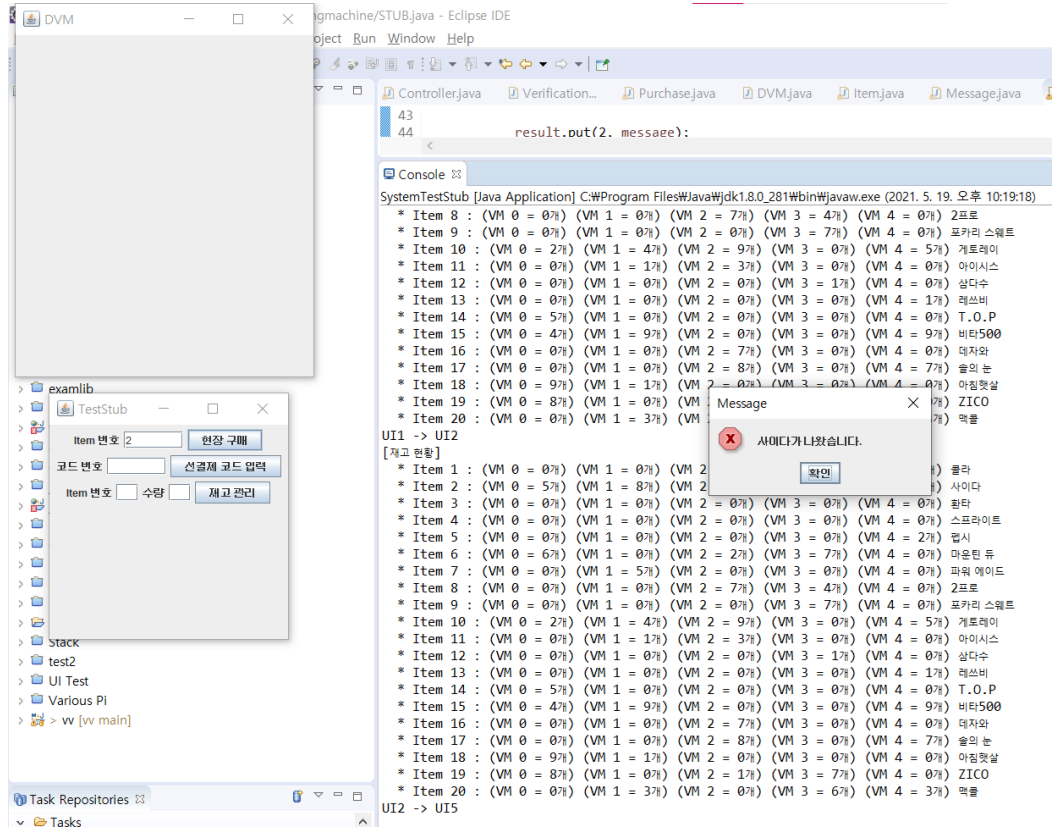


02 Test Stub

실제 실행 모습

1. 현장 구매 버튼

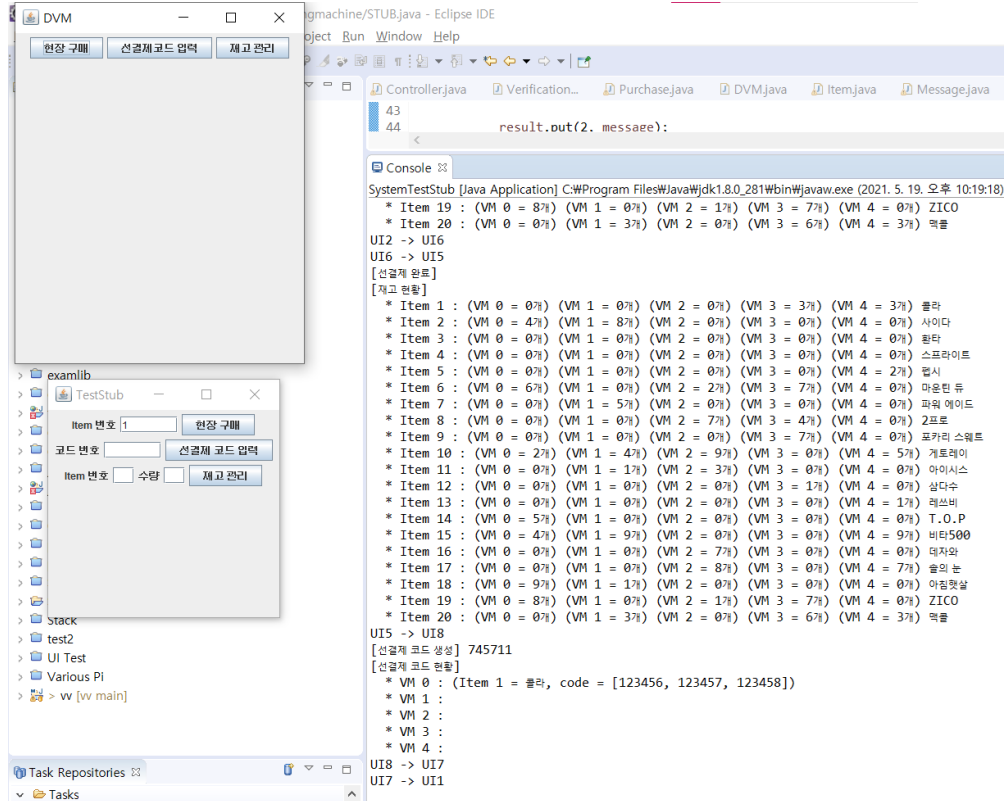
1) 현장 구매



02 Test Stub

실제 실행 모습

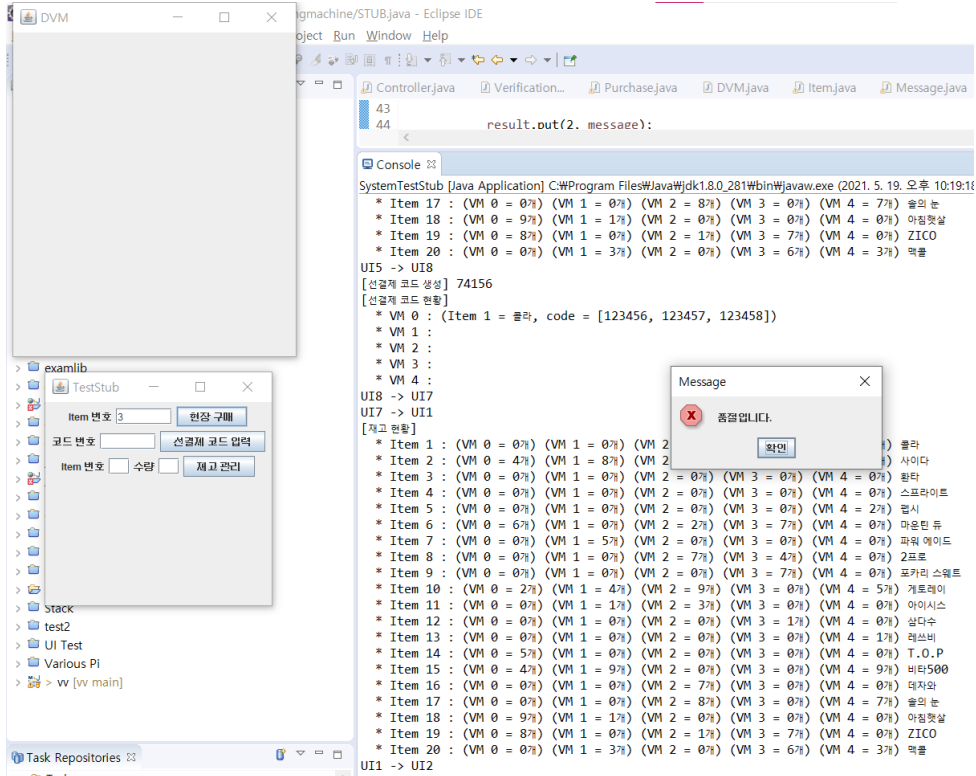
- 1. 현장 구매 버튼
- 2) 선구매



02 Test Stub

실제 실행 모습

- 1. 현장 구매 버튼
- 3) 품질



The screenshot shows the Eclipse IDE environment. The main editor displays the source code for `Verification.java`, with lines 43 and 44 highlighted. The console window shows the output of the `SystemTestStub` application, including a list of items and their verification status. A message dialog box is displayed in the foreground, showing a red 'X' icon and the text '품질입니다.' (Quality is here).

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 19 오후 10:19:11)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개) 물의 눈
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 아침햇살
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개) ZICO
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개) 맥콜

UI5 -> UI8
[선결제 코드 생성] 74156
[선결제 코드 현황]
* VM 0 : (Item 1 = 클라, code = [123456, 123457, 123458])
* VM 1 :
* VM 2 :
* VM 3 :
* VM 4 :

UI8 -> UI7
UI7 -> UI1
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 클라
* Item 2 : (VM 0 = 4개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 사이타
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 활약
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 스프라이트
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개) 뎀시
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 2개) (VM 3 = 7개) (VM 4 = 0개) 마운틴 뷰
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 파워 에이트
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개) 2로보
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개) 포카리 스위트
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개) 게트레이
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개) 아이시스
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개) 삼다수
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개) 레쓰비
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) T.O.P
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개) 비타500
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개) 더자와
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개) 물의 눈
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 아침햇살
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개) ZICO
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개) 맥콜

UI1 -> UI2
```

02 Test Stub

실제 실행 모습

2. 선결제 코드 입력

1) 빈칸 입력

The screenshot shows the Eclipse IDE environment. On the left, there are two windows: 'DVM' and 'TestStub'. The 'DVM' window contains a numeric keypad with buttons for digits 1-9, 0, and buttons for '취소' (Cancel) and '확인' (OK). The 'TestStub' window contains input fields for 'Item 번호', '수량', and '재고 관리', along with buttons for '현장 구매', '선결제 코드 입력', and '재고 관리'. The main IDE window shows the 'Controller.java' file with a line of code: `result.out(2, message);`. The 'Console' window displays the output of the application, showing a list of items and their prices. A 'Message' dialog box is overlaid on the console, displaying the text '코드를 입력해주세요.' (Please enter the code.) and a '확인' (OK) button.

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 19)
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개)
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
UI2 -> UI1 [품질]
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 2 : (VM 0 = 4개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개)
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 2개) (VM 3 = 7개) (VM 4 = 0개)
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개)
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)
```

02 Test Stub

실제 실행 모습

2. 선결제 코드 입력

2) 숫자 외의 문자 입력

The screenshot displays the Eclipse IDE environment. The main window shows a Java application titled 'DVM' with a numeric keypad and a text input field containing '12345a'. Below the keypad are '취소' (Cancel) and '확인' (OK) buttons. The IDE's console shows the following output:

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 1'  
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)  
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)  
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)  
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)  
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)  
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)  
UI2 -> UI1 [종료]  
[재고 현황]  
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 2 : (VM 0 = 4개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 2개) (VM 3 = 7개) (VM 4 = 0개)  
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개)  
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개)  
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)  
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)  
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)  
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)  
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
```

A message dialog box is shown with the text '숫자만 입력 가능합니다.' (Only numbers can be entered.) and a '확인' (OK) button.

02 Test Stub

실제 실행 모습

2. 선결제 코드 입력

3) 유효하지 않은 코드

The screenshot shows the Eclipse IDE environment. The main window displays a Java application titled "DVM" with a numeric keypad interface. The keypad has buttons for digits 1-9 and 0, along with "취소" (Cancel) and "확인" (OK) buttons. The text "123450" is entered in the input field. Below the keypad, there are two buttons: "취소" and "확인".

The Eclipse IDE interface shows the "Console" window with the following output:

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 11)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)
UI2 -> UI1 [종결]
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 3개) (VM 4 = 3개)
* Item 2 : (VM 0 = 4개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
```

A message dialog box is displayed with the text "잘못된 코드입니다." (Invalid code) and a "확인" (OK) button.

02 Test Stub

실제 실행 모습

2. 선결제 코드 입력

4) 유효한 코드

The screenshot displays the Eclipse IDE environment. The main editor shows the source code for `Controller.java`, with lines 43 and 44 visible, indicating a call to `result.out(2, message):`. The console window shows the output of the application, listing 20 items with their respective VM counts. A message dialog box is displayed in the foreground, titled "Message", with the text "클라가 나왔습니다." (The clerk is here.) and a "확인" (OK) button.

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 1)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 17개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)
UI2 -> UI1 [품질]
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 3개) (VM 4 = 3개)
* Item 2 : (VM 0 = 4개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개)
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 17개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)
```

02 Test Stub

실제 실행 모습

3. 재고 관리

1) 정상 처리

The screenshot shows the Eclipse IDE environment. The main window is titled 'DVM' and contains three buttons: '현장 구매', '선택제 코드 입력', and '재고 관리'. Below this window, the IDE's project explorer shows a folder structure including 'examlib', 'TestStub', 'Stack', 'test2', 'UI Test', and 'Various Pi'. The 'TestStub' folder is expanded, showing a file named 'wv [wv main]'. The IDE's editor shows 'Controller.java' with the following code snippet:

```
Controller dvm = Controller.getInstance();
```

The console output shows the following items and their VM counts:

- * Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개) 포카리 스위트
- * Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개) 게토레이
- * Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개) 아이시스
- * Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개) 삼다수
- * Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개) 레쓰비
- * Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) T.O.P
- * Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개) 비타500
- * Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개) 데자와
- * Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개) 슬의 눈
- * Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 아침햇살
- * Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개) ZICO
- * Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개) 맥클

The console also shows the following commands and output:

```
UI4 -> UI1
UI1 -> UI4
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 3개) (VM 4 = 3개) 콜라
* Item 2 : (VM 0 = 8개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 사이다
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 활타
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 스프라이트
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개) 펄시
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 2개) (VM 3 = 7개) (VM 4 = 0개) 마운틴 듀
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 파워 예이드
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개) 2프로
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 포카리 스위트
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개) 게토레이
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개) 아이시스
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개) 삼다수
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개) 레쓰비
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) T.O.P
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개) 비타500
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개) 데자와
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개) 슬의 눈
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개) 아침햇살
* Item 19 : (VM 0 = 12개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개) ZICO
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개) 맥클
UI4 -> UI1
```

02 Test Stub

실제 실행 모습

3. 재고 관리

2) 예상 음료 종류가 7가지
지를 초과하는 경우

The screenshot shows the Eclipse IDE environment. On the left, a window titled 'DVM' displays a simple UI with a dropdown menu for '음료' (Beverage) set to '콜라' (Cola) and a quantity input field set to '2'. Below it are '취소' (Cancel) and '확인' (OK) buttons. In the background, another window titled 'examlib' shows a more complex UI with fields for 'Item 번호' (Item No.), '코드 번호' (Code No.), and '수량' (Quantity), along with buttons for '현장 구매' (Purchase on-site), '선결제 코드 입력' (Enter pre-payment code), and '재고 관리' (Inventory Management). The main Eclipse IDE window shows the source code for 'Controller.java' with a call to `Controller.getInstance()`. The Console window displays the output of the application, listing 20 items with their VM counts. A 'Message' dialog box is overlaid on the console, displaying an error message: '7종류의 음료만 판매할 수 있습니다.' (Only 7 types of beverages can be sold.) with an '확인' (OK) button.

```
SystemTestStub [Java Application] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (2021. 5. 1)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)

UI4 -> UI1
UI1 -> UI4
[재고 현황]
* Item 1 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 2 : (VM 0 = 8개) (VM 1 = 8개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개)
* Item 3 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 4 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 5 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 2개)
* Item 6 : (VM 0 = 6개) (VM 1 = 0개) (VM 2 = 2개) (VM 3 = 7개) (VM 4 = 0개)
* Item 7 : (VM 0 = 0개) (VM 1 = 5개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 8 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 4개) (VM 4 = 0개)
* Item 9 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 7개) (VM 4 = 0개)
* Item 10 : (VM 0 = 2개) (VM 1 = 4개) (VM 2 = 9개) (VM 3 = 0개) (VM 4 = 5개)
* Item 11 : (VM 0 = 0개) (VM 1 = 1개) (VM 2 = 3개) (VM 3 = 0개) (VM 4 = 0개)
* Item 12 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 1개) (VM 4 = 0개)
* Item 13 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 1개)
* Item 14 : (VM 0 = 5개) (VM 1 = 0개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 15 : (VM 0 = 4개) (VM 1 = 9개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 9개)
* Item 16 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 7개) (VM 3 = 0개) (VM 4 = 0개)
* Item 17 : (VM 0 = 0개) (VM 1 = 0개) (VM 2 = 8개) (VM 3 = 0개) (VM 4 = 7개)
* Item 18 : (VM 0 = 9개) (VM 1 = 1개) (VM 2 = 0개) (VM 3 = 0개) (VM 4 = 0개)
* Item 19 : (VM 0 = 8개) (VM 1 = 0개) (VM 2 = 1개) (VM 3 = 7개) (VM 4 = 0개)
* Item 20 : (VM 0 = 0개) (VM 1 = 3개) (VM 2 = 0개) (VM 3 = 6개) (VM 4 = 3개)

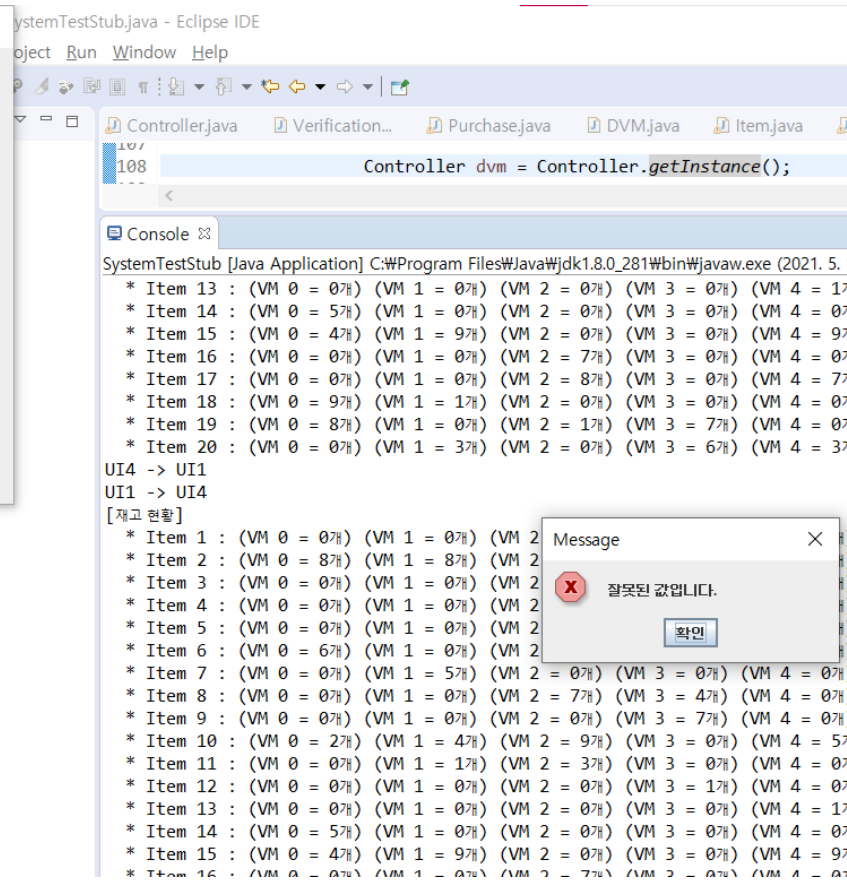
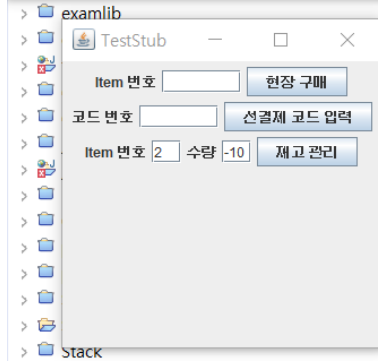
Message
7종류의 음료만 판매할 수 있습니다.
확인
```

02 Test Stub

실제 실행 모습

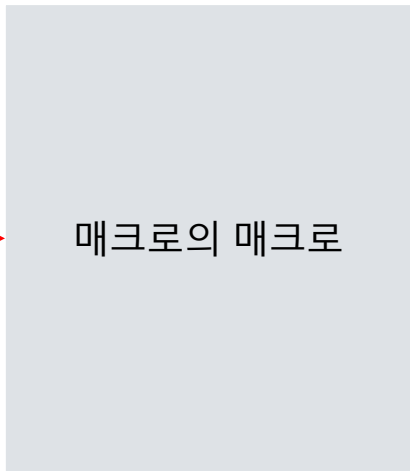
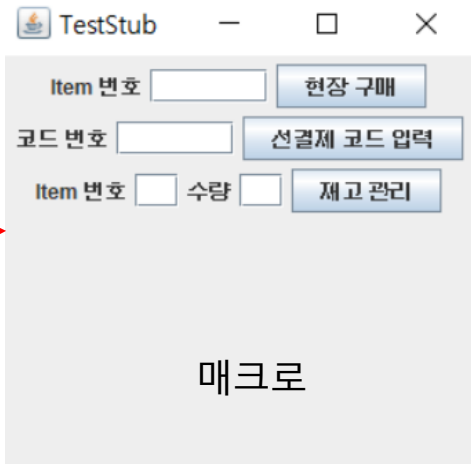
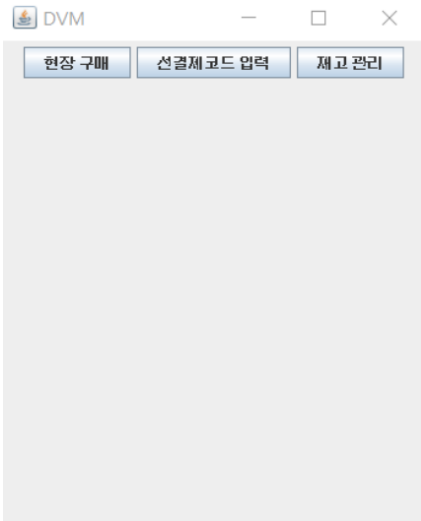
3. 재고 관리

3) 예상 재고가 음수가 되는 경우



02 Test Stub

매크로의 매크로



02 Test Stub

매크로의 매크로로 테스트하기 어려운 이유

- 현장구매 하나만 가지고도 item out 순서에 따른 조합이 매우 많음
 - o 예를 들어 특정 샘플에 대해 조합의 개수를 계산해보면
 - o $15 \times 4 \times 8 \times 9 \times 4 \times 7 \times 19 \times 5 \times 5 \times 21 \times 8 \times 9 \times 11 \times 18 \times 12 \times 12 \times 9 \times 7$
 - o 즉, **10,403,133,021,388,800** 개의 조합을 테스트해야 함
 - o 버튼을 클릭하여 연산이 수행되고 페이지가 넘어가는 delay를 무시한다고 해도 시간적으로 테스트 불가능
 - o 샘플을 단순화 시켜서 item당 재고가 최대 2개만 가능하다고 해도 1,048,576개의 조합 생성
- 현장구매/선결제코드입력/재고관리의 수행 순서가 섞이는 경우
 - o 그런데 그 1,048,576개의 조합이 '선결제코드입력'과 '재고관리'와 **또 다시 조합**을 이루게 되면, 역시 테스트 불가능한 개수가 나옴

03 Category-Partition Testing

```
#  
# Test specification for DVM  
#  
  
Environments:  
  
Current UI:  
  initial page.  
    [property ui1]  
  item select.      [property ui2]  
  input verification code. [property ui3]  
  modify stock.     [property ui4]  
  payment.          [property ui5]  
  inform prepurchase. [property ui6]  
  inform location.  [property ui7]  
  inform verification code. [property ui8]  
  
Mode:  
  purchase.          [property mode1] [if ui5]  
  prepurchase.      [property mode2] [if ui5]  
  
is Advance:  
  false.            [if ui5]  
  true.             [if ui5]  
  
selected item stock:  
  local stock zero and remote stock zero.  
    [if ui2]  
  local stock zero and valid remote stock.  
    [if ui2]  
  valid local stock. [if ui2]  
  
expected changed local stock:  
  less than zero.   [if ui4]  
  more than zero.   [if ui4]  
  
expected changed local item type:  
  less than seven. [if ui4]
```

more than seven. [if ui4]

Parameters:

Page move button:

initial page-purchase button.

[if ui1]

initial page-prepurcahse button.

[if ui1]

initial page-stock management button.

[if ui1]

select item button. [if ui2]

cancel button. [if ui2 || ui3 || ui4 || ui5 || ui6]

confirm button. [if ui3 || ui4 || ui7 || ui8]

1% payment fail. [if ui5]

100% payment fail. [if ui5]

location button. [if ui6]

Manage Stock:

select item zero. [if ui4]

select item less than zero. [if ui4]

select item more than zero. [if ui4]

Verification Code:

valid verification code. [if ui3]

invalud verification code. [if ui3]

03 Category-Partition Testing

TSL Generator 사용하여 Test Case 생성

```
-----  
TSLgenerator  
(C) University of California Irvine,  
and Oregon State University, 2001  
-----  
353 test frames generated and written to dvm.ts1  
-----  
TSLgenerator  
(C) University of California Irvine,  
and Oregon State University, 2001  
-----  
87 test frames generated and written to dvm.ts1  
-----  
TSLgenerator  
(C) University of California Irvine,  
and Oregon State University, 2001  
-----  
44 test frames generated and written to dvm.ts1
```

Test Case 생성

Environment constraint 적용 -> 353개

Environment constraint 부분 삭제 후 적용 -
> 87개

Property constraint 적용 -> 44개

03 Category-Partition Testing

Constraint 적용

1.1 Categorize & Property & Constraints

Group	Category	Value	Constraint [Property prop]
Environments	Current UI	initial page.	[property ui1]
Environments	Current UI	item select.	[property ui2]
Environments	Current UI	input verification code	[property ui3]
Environments	Current UI	modify stock	[property ui4]
Environments	Current UI	payment	[property ui5]
Environments	Current UI	inform prepurchase	[property ui6]
Environments	Current UI	inform location	[property ui7]
Environments	Current UI	inform verification code	[property ui8]
Environments	Mode	purchase	[property mode1] [if ui5]
Environments	Mode	prepurchase	[property mode2] [if ui5]
Environments	is Advance	false	[if ui5]
Environments	is Advance	true	[if ui5]
Environments	selected item stock	local stock zero and remote stock zero	[if ui2]
Environments	selected item stock	local stock zero and valid remote stock	[if ui2]
Environments	selected item stock	valid local stock	[if ui2]
Environments	expected changed local stock	less than zero	[if ui4]
Environments	expected changed local stock	more than zero	[if ui4]
Environments	expected changed local item type	less than seven	[if ui4]

Group	Category	Value	Constraint [Property prop]
Environments	expected changed local item type	more than seven	[if ui4]
Parameters	Page move button	initial page-purchase button.	[if ui1]
Parameters	Page move button	initial page-prepurchase button.	[if ui1]
Parameters	Page move button	initial page-stock management button	[if ui1]
Parameters	Page move button	select item button	[if ui2]
Parameters	Page move button	cancel button	[if ui2 ui3 ui4 ui5 ui6]
Parameters	Page move button	confirm button	[if ui3 ui4 ui7 ui8]
Parameters	Page move button	location button	[if ui6]
Parameters	Page move button	1% payment fail	[if ui5]
Parameters	Page move button	100% payment fail	[if ui5]
Parameters	Manage Stock	select item zero	[if ui4]
Parameters	Manage Stock	select item less than zero	[if ui4]
Parameters	Manage Stock	select item more than zero	[if ui4]
Parameters	Verification Code	valid verification code	[if ui3]
Parameters	Verification Code	invalid verification code	[if ui3]

Constraint(single, property)를 적용하여 test case 44개로 감소

03 Category-Partition Testing

CPT Result

<u>Aa</u> TestCase	<u>≡</u> Result	<u>≡</u> Description
<u>1</u>	Pass	
<u>2</u>	Pass	
<u>3</u>	Pass	
<u>4</u>	Pass	
<u>5</u>	Pass	
<u>6</u>	Pass	

<u>7</u>	Pass	
<u>8</u>	Pass	
<u>9</u>	Pass	
<u>10</u>	Pass	
<u>11</u>	Pass	
<u>12</u>	Pass	
<u>13</u>	Pass	
<u>14</u>	Pass	
<u>15</u>	Pass	
<u>16</u>	Pass	

<u>17</u>	Pass	
<u>18</u>	Pass	
<u>19</u>	-	
<u>20</u>	-	
<u>21</u>	Pass	
<u>22</u>	-	
<u>23</u>	-	
<u>24</u>	Pass	
<u>25</u>	Pass	
<u>26</u>	Pass	
<u>27</u>	Pass	
<u>28</u>	Pass	
<u>29</u>	Pass	
<u>30</u>	Pass	
<u>31</u>	Pass	
<u>32</u>	Pass	
<u>33</u>	Pass	
<u>34</u>	Pass	
<u>35</u>	Pass	

<u>36</u>	Pass	
<u>37</u>	Pass	
<u>38</u>	Pass	

<u>39</u>	Pass	
<u>40</u>	Pass	
<u>41</u>	Pass	
<u>42</u>	Pass	
<u>43</u>	Pass	
<u>44</u>	Pass	
<u>45</u>	Pass	

TestCase 19, 20, 22, 23은 logical contradictory 한 테스트 케이스라서 제외함. 40 / 40 = 100%

04 Brute Force Testing

3학년 개발팀(B1)이 정의한 기능(OOPT 2063)을 구체화하고 실제 프로그램 실행시 기능이 잘 동작하는지 확인

Use case	Number	Test case	Test Result	없는 상황 발생	있어 테스트 불가	황 발생
Set up	1-1	시스템이 시작되었을 때 다른 DVM의 ID를 받아 있는지 확인	PASS	구매가 성공했을 때 유료 재고를 맞게 줄어 있는지 확인	PASS	구매 결정할 유료의 재고가 없다면 품질에 시지를 보여주고, "Select Mode"로 들어가는지 확인
	1-2	시스템이 시작되었을 때 다른 DVM의 재고를 받아왔는지 확인	PASS	구매가 성공했을 때 해당 음료를 배출하는지 확인	PASS	유치학인 모드를 선택한 음료에 대해 "Message Request"로 재고를 받아오는지 확인
	1-3	시스템이 시작되었을 때 자판기의 재고가 업데이트 되었는지 확인	PASS	구매 결정 실패했을 때 결제 실패 메시지를 보여주는지 확인	PASS	유치학인 모드를 선택한 유료의 재고가 남아 있다면 해당 음료에 대해 "Inform Location"을 실행하는지 확인
Select mode	2-1	USER가 선택한 모드에 따라 정해진 메뉴가 실행되는지 확인	PASS	선택한 구매코드를 받아오는지 확인	PASS	유치학인 모드를 선택한 유료의 재고가 남아 있다면 해당 음료에 대해 "Inform Location"을 실행하는지 확인
	2-2	USER가 여러 개의 모드를 선택했을 때 항상 input으로 한번에 여러 개의 모드를 선택하는 경우는 불가능	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Manage Stock	3-1	Manager가 경신한 재고 정보가 적용되는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	3-2	Manager가 총 8가지 이상의 음료를 경신할 경우 에러가 나오는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	PASS	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Show item	4-1	모든 음료가 나오는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	PASS	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	4-2	User가 선택한 유료가 선택되는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Check stock count	6-1	User가 선택한 음료에 대해 현재 자판기에서 판매 가능한지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	6-2	현재 자판기에서 판매 가능하면 Purchase를 정상적으로 실행하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	6-3	통신을 통해 다른 자판기로부터 재고를 받아오는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	6-4	다른 자판기로부터 재고가 있다는 응답을 받으면, "Select Advance Payment"를 정상적으로 실행하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Update stock	7-1	음료 재고정보를 주어진 값에 맞게 최신화 하는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Purchase	8-1	구매가 성공했을 때 유료 재고를 맞게 줄어 있는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	8-2	구매가 성공했을 때 해당 음료를 배출하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	8-3	구매 결정 실패했을 때 결제 실패 메시지를 보여주는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Advance purchase	9-1	선택한 구매코드를 받아오는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	9-2	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	9-3	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Check payment	10-1	결제 결과를 받아오는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	10-2	정상적인 결제 카드가 아니라면 "잘못된 카드"라는 에러 메시지를 보여주는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	10-3	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Select advance payment	11-1	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	11-2	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	11-3	재고를 감소시킨 후 "Advance Purchase"를 실행하는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Inform location	12-1	유료를 갖고있는 자판기에 대해 위치를 보여 주는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	13-1	음료가 선택되었을 때, 새로운 구매코드를 생성하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	13-2	"Create Verification Code"를 통해 생성된 구매코드를 보여주는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Create verification code	13-3	선택한 구매코드를 여러 종류의 음료에 중복하여 생성되는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	13-4	선택한 구매코드를 여러 종류의 음료에 중복하여 생성되는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Item out	15-1	유료 선택되어 있고, 결제가 완료되었거나 유효한 구매 코드를 입력했을 때만 음료를 제공하는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	15-2	현재 자판기에 재고가 없을 때는, 구매 코드를 입력해도 item out이 되지 않아야 함	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	16-1	User가 입력한 구매 코드를 "Check Verification Code"에 전달하고 그 결과를 받는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Read verification code	16-2	기존된 코드가 유효하다면 "Item Out"을 실행하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	16-3	"Item Out"을 실행한 후에 "Reset Verification Code"를 실행하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Check verification code	17-1	시스템에 입력된 코드가 존재하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	18-1	시스템에서 입력된 코드를 정상적으로 삭제하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	19-1	보내고자하는 메시지 종류를 수신 대상에게 잘보내지는지 확인	FAIL	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
Message request	19-2	수신된 메시지에 따라 명시된 응답을 전송한 DVM에 정상적으로 보내지는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인
	19-3	알 수 없는 종류에 메시지를 수신한 경우 알 수 없음을 응답하는지 확인	PASS	선택한 구매코드를 받아온 선택구매코드를 "Message Request" 통해 "Broadcast" 하는지 확인	FAIL	유치학인 모드를 선택한 유료의 재고가 남아 있지 않다면 품질메시지를 보여주고, "Select Mode"로 들어가는지 확인

전체 Test Case : 50 개
(35 / 50) = 70% Pass

50개의 test case 중에서 35개의 test case pass => 70% Pass

04 Brute Force Testing - Trello

BFT 결과 trello에 등록

The screenshot shows a Trello card with the following content:

- Title:** [Select Mode] 불가능한 테스트 케이스 수정 요청
- Description:** Select mode- USER가 여러 개의 모드를 선택했을 때 현상구 매모드가 실행되는지 확인
FAIL- -> 버튼 입력을 통한 input으로 한번에 여러 개의 모드를 선택하는 경우는 불가능
- Text:** 버튼 입력 형태로 프로그램이 작성되어있기 때문에 해당 테스트 케이스는 성립하지 않습니다. 삭제 혹은 다른 테스트 케이스로 수정 부탁드립니다.
- Attachments:** 20210519_234118.png (Added 18 minutes ago)

The screenshot shows a Trello board titled "이슈 현황" with four cards:

- Card 1:** [Update Stock] 선구매 코드를 이용하여 item out 실행 후 다른 자판기에서 재고가 update되지 않는 상황 발생
- Card 2:** [Advance Purchase] 선구매 코드가 다른 자판기에 저장되지 않음.
- Card 3:** [Select advance payment] 선구매 진행을 위한 결제 시 다른 자판기들의 재고가 바로 줄어들지 않음.
- Card 4:** [Check verification code] 서로 다른 종류의 용료에 같은 선구매 코드 할당

BFT 진행 결과 FAIL된 Test case trello에 issue로 등록하여 3학년 개발팀에게 전달하여 관리

감사합니다

