




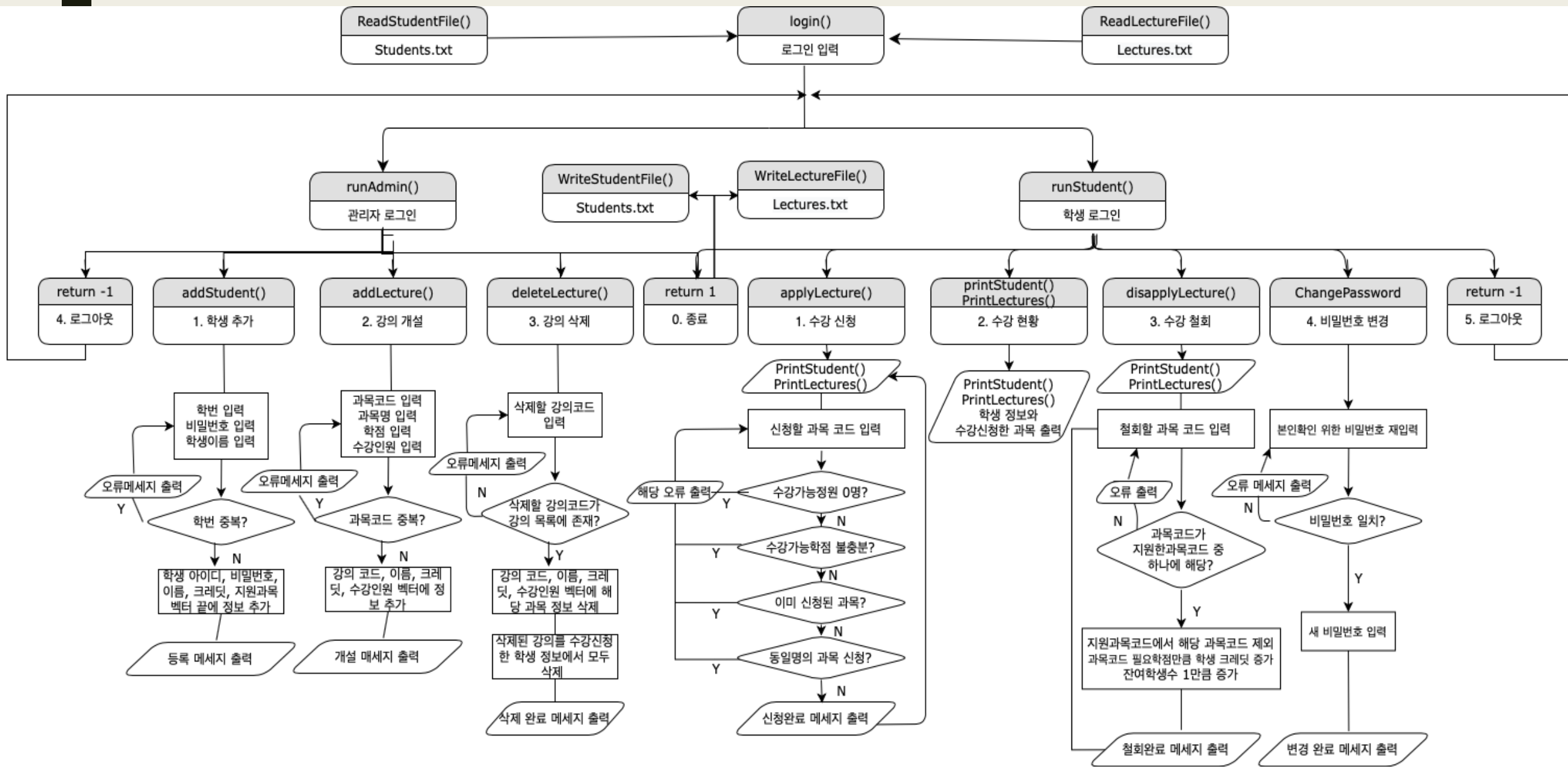
# Flowchart

객체지향프로그래밍 중간 대체 프로젝트

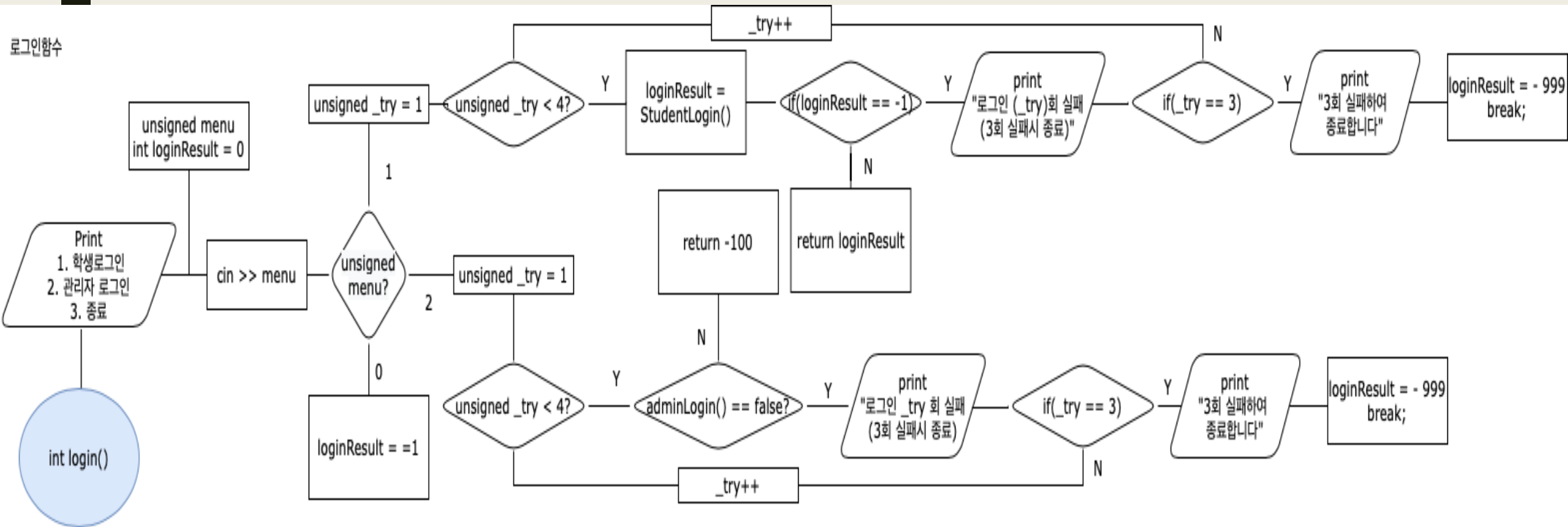
2016101407 경영학과 한석준



# 전체적인 시스템 플로우차트



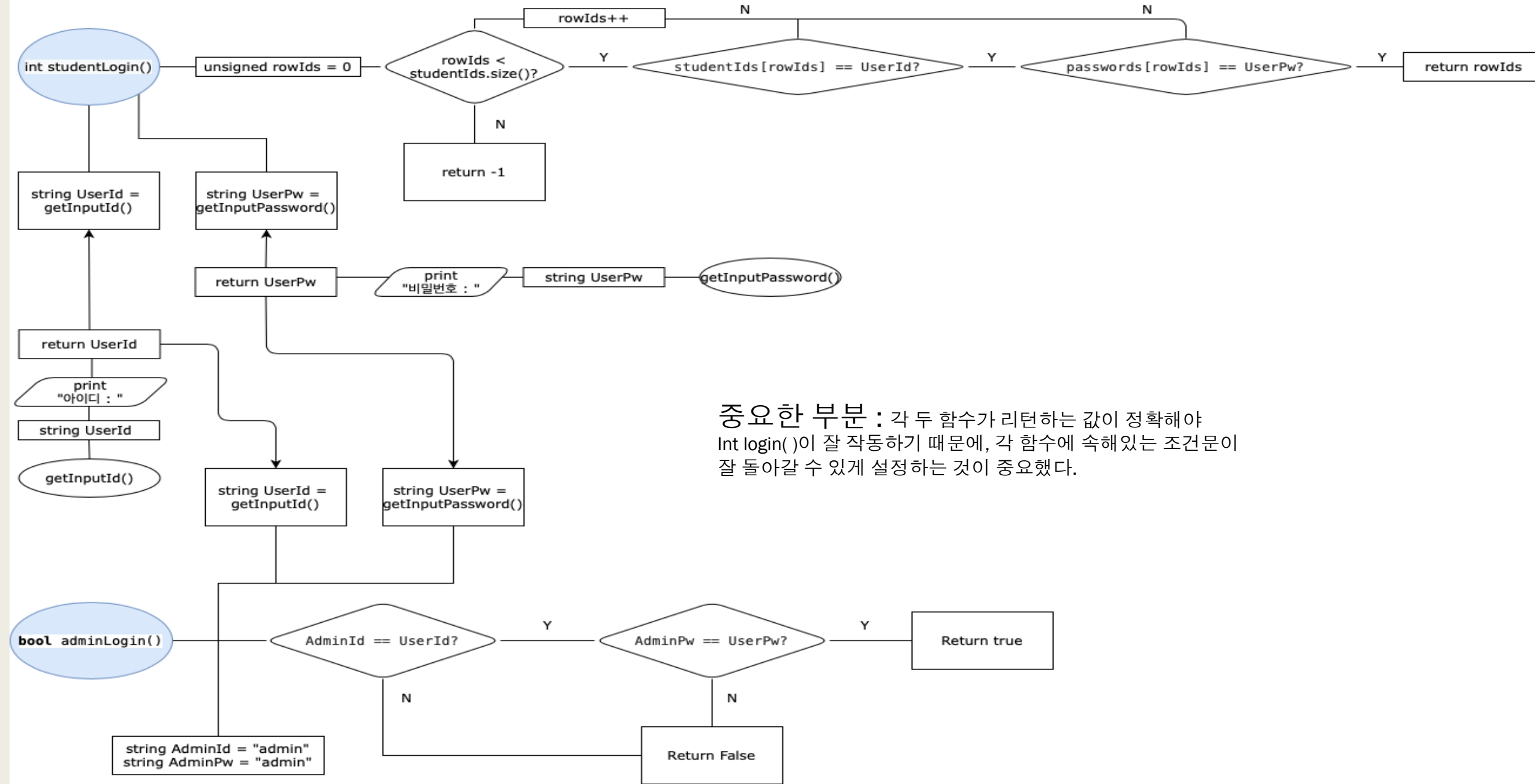
# 로그인 함수 플로우차트 (int login( ))



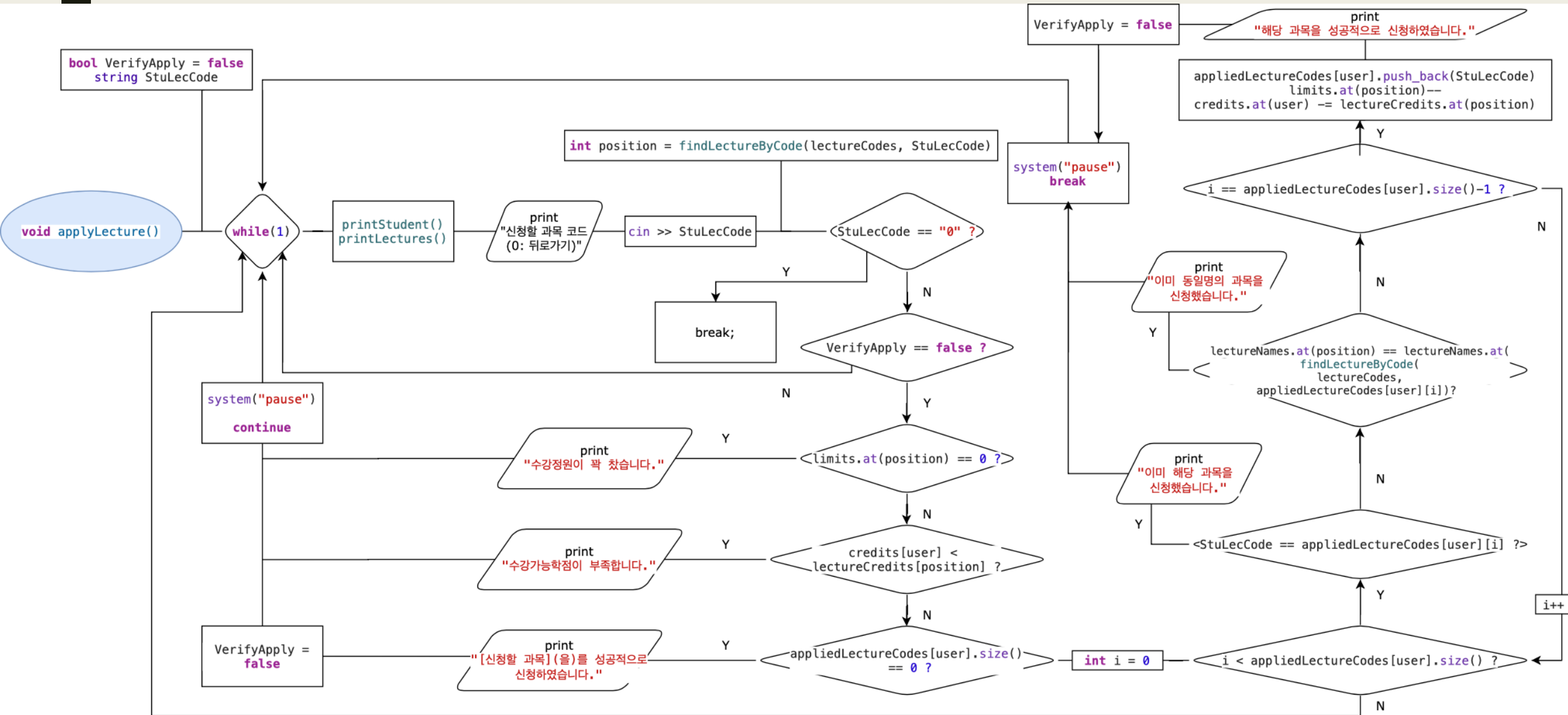
중요했던 부분 : 각 상황에 맞는 리턴값을 리턴해줄 수 있게 조건문을 구성하는 것.

ex) 조건을 판별하여 학생 메뉴의 경우 로그인 3회 미만 실패시 loginResult = -1, 3회 실패시 loginResult = -999, 로그인 성공시 loginResult = studentLogin()  
관리자 메뉴의 경우 adminlogin() == false일시 실패 프린트. 3회 실패시 loginResult = -999, 성공시 return -100.

# 로그인 함수 플로우차트 (int studentLogin( ), bool adminLogin( ))



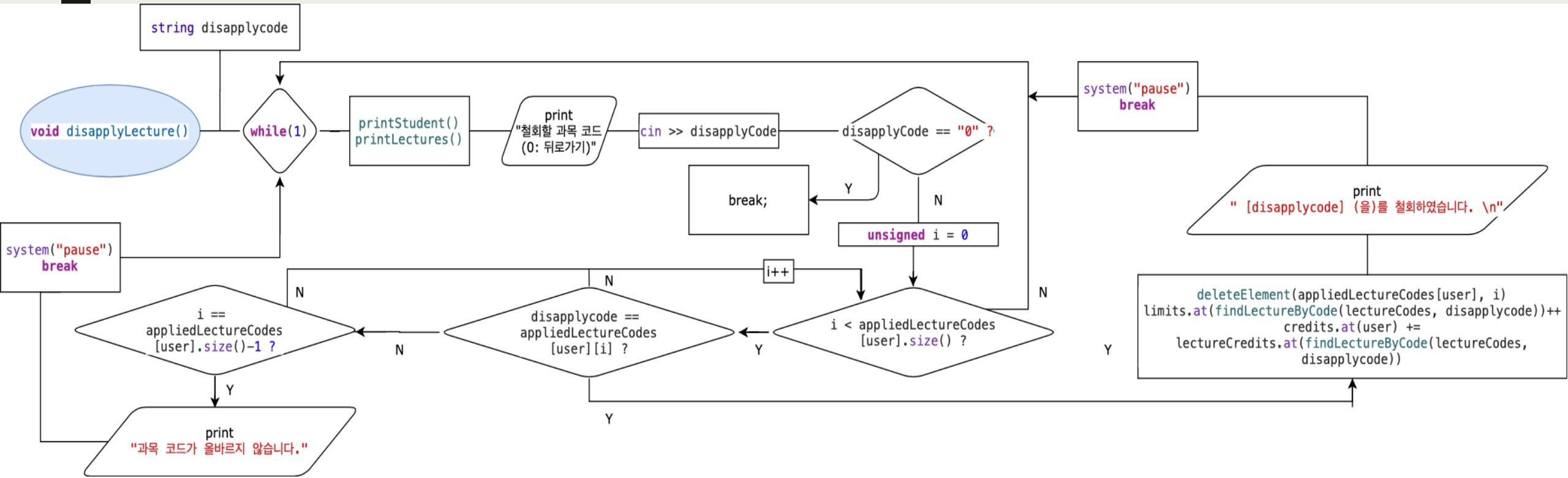
# 학생메뉴 함수 플로우차트(void applyLecture( ))



**어려웠던 부분** : void applyLecture()은 조건을 구성하는 것들이 너무 많아 어려웠다. 특히 동일명의 과목을 신청했는 지에 대한 판별 조건을 구성하는 것이 어려웠다.

StuLecCode의 인덱스랑 같은 LectureNames와, 이미 지원한 과목코드를 과목 목록에서 찾아 그에 해당하는 인덱스를 LectureNames에서 찾는 조건식을 구성하는데 있어 어려움이 많았다.

# 학생메뉴 함수 플로우차트 (void disapplyLecture( ))



**중요한 부분 :** disapplycode가 appliedLectureCodes 안에 존재할 시, 처리할 동작들을 제대로 구현하는게 중요했다.  
 대상 코드의 인덱스를 정확히 찾아 limits, lectureCredits 벡터 안에서 제거해야했고, 로그인한 유저의 appliedLectureCodes에서 삭제하는 메소드를 구현하는게 중요했다.

# 관리자메뉴 함수 플로우차트 (void deleteLecture())

