2019 소프트웨어프로젝트 프로젝트 3

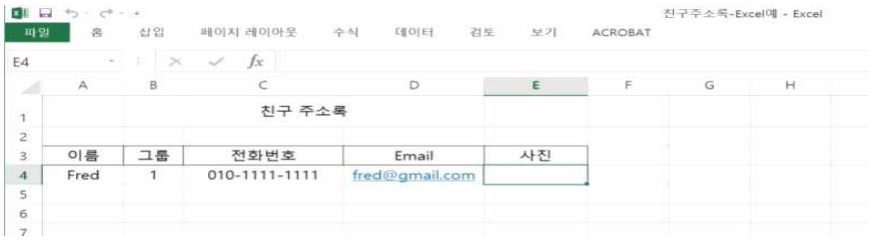
친구목록 프로그램 (FriendsList)

1. 내용

카톡이나 전화 앱에 있는 친구목록 관리 프로그램을 작성한다. 지금 단계에서는 콘솔 입출력 버전을 만들고, 향후 프로젝트에서 GUI 버전을 만들게 된다.

단순히 입출력으로 보여주는 프로그램이 아니고, 친구 정보를 내부적으로 입력/저장관리/출력하는 프로그램을 작성하는 것임. 향후 친구정보의 추가/수정/삭제로 확장이 용이하게 객체 지향 개념으로 작성하여야 한다. 따라서 다음 요구 사항에 맞게 프로그램을 작성하여야 한다.

1) Friend class
   - 프로그램에서 하나의 친구를 나타냄.
   - Excel로 친구 관리를 한다고 가정한다면, 각 행에 해당
   - 다음 정보들을 포함하여야 한다.
     이름, Group, 전화번호, email주소, 프로필 사진 (생략 가능) 5개 항목
   - 정보들을 다루는 각종 메소드들 포함
   - 하나의 친구에 대한 정보를 콘솔로 출력하여 주는 public void print() 메소드는 **필수**
2) FriendList class
   - 친구들의 묶음을 나타냄.
   - Excel로 친구 관리를 한다고 가정한다면, 테이블 전체에 해당
   - Friend 인스턴스들을 모아서 관리하는 클래스.
   - 자료구조는 array 사용 (향후, ArrayList로 갱신)
   - 목록에 있는 친구 숫자를 알려주는 public int numFriends(),
   - 목록 상, i 번째 친구 정보를 알려주는 public Friend getFriend(int i) 메소드는 **필수**



3) FriendListFile class
   - 파일에 기술된 내용을 읽어서, 파싱을 해서 항목별로 친구 정보와 친구 목록을 만드는,
   - 반대로, 친구 목록을 파일에 저장하는 메소드들의 집합.
   - 주어진 이름의 파일에서 친구목록을 만드는,
     `public FriendList readFileToList(String fileName)` 메소드는 **필수**
   - 그 외의 내부 구현 사항은 보여서는 안 됨.
   - main()을 포함하면 안 됨.
4) 친구 정보를 기술하는 파일은 다음 양식을 따른다.
   - 하나의 라인이 한 명의 친구를 기술.
     - ":"을 구분자로 하고, 친구이름, 그룹, 전화번호, email, 사진, 5개 항목으로 구성
     - 5번째 사진은 생략 가능
     - 중간의 공백문자는 무시
     - 예) Fred: 1:010-1111-1111: fred@google.com: fred.gif
     - 빈줄, 또는 처음을 //로 시작하면 comment 처리
     - 정보 사항은 모두 영어로. (즉, 한글은 사용하지 않음).
     - 파일에 나오는 친구 수는 100개 이하로 가정한다. max = 100

5) 잘못된 파일명, 잘못된 규칙으로 기술된 파일 내용 등 오류에 대해 콘솔에 보고하여야 함.
   - 즉, 잘못된 입력 경우에 대처. exception
6) main()을 포함하는 별도 Test 클래스를 만드는 것이 좋음 (뒤의 AutoTest 참조)

2. 목적
   - Java String 구조 사용법 연습
   - File I/O 사용 연습
   - 인터페이스 정의를 통한 프로그램 배분/검사 등 분업화된 프로그래밍 맛보기

3. 추진 방법
   1) 문제 정의에 입각해서 프로그램 구조 설계. 즉, 사용할 클래스들 정의.
      - Friend, FriendList, FriendListFile
   2) 친구 정보 기술방법 ("위 1-4)")이해
   3) String이 제공하는 method들 공부

   --- 정상 입력 파일에 대한 동작 완성 ----------
   4) 1단계로 파일에서 가장 간단한 형태 한 줄을 읽어서 정해진 필드를 분리하는 파싱 메소드를 작성하고, 그 결과에 해당하는 **하나의 친구 정보**(class Friend) **인스턴스/객체를** 만드는 작업을 완성.
   5) print()를 코딩하여 제대로 파싱을 하는지 확인
   6) 여러 줄의 파일 입력, 즉, 여러 제품 정보들의 처리 코딩.
      -- 즉, FriendList 코딩

   --- 비정상 입력 파일에 대응하도록 확장 ----------
   7) comment 라인을 처리하는 문장을 추가해서 파일 읽기 메소드를 완성
   8) 제품 ID의 충돌을 점검하고 출력해서 알려주는 기능 추가
   9) 각종 정상/비정상 입력에 대해 동작 점검 및 프로그램 수정
   ----

   10) **구조 > 기능 > 성능**
   11) 자동 실행검사 (AutoTest) 이용해서 최종 점검 main이 있는 class 빼고 jar 파일 만들기
   12) 리포트 작성

4. AutoTest를 이용한 구현 검증
   1) 주어진 문제의 요구사항을 만족하는지를 웹을 통해 자동 검사
      - 인터페이스 요구 사항, 즉, 앞의 1-1), 1-2), 1-3) 규정을 준수
      - 기능적 요구 사항 만족. 즉, 앞의 1-4)부터 5)까지 준수
   2) 검사 방법
      a. 구현한 프로그램들에서, 테스트용 main()이 포함된 **class를 제외한** 나머지 class들을 "jar"를 이용하여 FriendList.jar로 만든다.
      b. 165.194.35.156에 접속하여 준비한 FriendList.jar를 upload하고 테스트 결과를 확인한다.
      c. 먼저, 정상 입력 case에 집중해서 AutoTest를 수행하여 완성한다.
      d. 비정상 모든 항목의 결과가 "Correct Answer"와 유사하게 나오면, 구현이 올바르게 된 것이므로 구현을 마치고, 그렇지 않은 경우 구현을 수정하여 다시 AutoTest를 수행한다.

5. 평가항목

- 프로그램의 동작 여부 (70%)
  * 동작 여부를 확연히 알아볼 수 있는 결과. 즉, AutoTest 결과 화면
  * 본인이 작성한 프로그램의 진위 여부를 판단하는데 도움 되는 인증 자료 포함
- 리포트 적정성 (30%)
  * 설계 및 구현 시 고민 했던 사항과 결정 내용을 간략히 기술
    + 본인이 직접 프로그램을 했음을 자연스럽게 입증
  * 평가자 입장에서, 평가사항을 찾기 쉽게 작성하였는가?

6. 리포트 제출

1) 기한 : 4/17(수) 강의시간 전까지
   - 최종 테스트 과정 화면, 설계 노트, 소스 프로그램, 자체평가표
   - 하나로 묶은 리포트를 eClass로 제출

<참고> file 입력 예

```
try {
        input = new Scanner(file);  // file is an instance of File
}
catch(Exception e) {
        // print error message and return;
}

while (input.hasNext()) {
        String line = input.nextLine();
        // 친구 정보 분해(파싱)
}
```