[ JAVA 프로그래밍 ]

[ 컬렉션 클래스 활용 ]

담당 교수 : 지정희 교수님

학 번 : 202311264

이 름 : 김민홍

제출일 : 2023년 11월 24일

|  |  |
| --- | --- |
| 1 | **문제정의 및 분석** |

**객관식 퀴즈를 지원하는 영어 단어장 프로그램을 작성하는 것이 목표.**

**프로그램의 전체적인 조건**

1. 단어는 Word 객체를 생성해서 저장할 것
2. 프로그램이 실행 중 런타임 에러로 프로그램이 종료되지 않도록 예외처리를 해줄 것
3. 사용자의 실수도 포함해서 예외 처리 해줄 것

1)주어진 quiz.txt 와 동일한 형식의 단어와 뜻으로 나열된 형식이면 읽어서 수행 가능 해야함

2)메뉴를 출력하고, 사용자가 5번을 입력할 때까지 메뉴를 반복수행(사용자가 1~5숫자 이외에 다른 문자나 값을 입력하더라도 반복 수행해야 한다

3)주관식 퀴즈-조건

1. 퀴즈는 한글 문제로 출제(사용자가 영단어를 입력해야함)

2. 서로 다른 10문제를 연속적으로 출제함.

3. 퀴즈를 모두 푼 다음에는 맞춘 점수와 퀴즈 푼 시간을 출력

4. 같은 뜻을 가진 여러 단어가 있는 경우, 여러 단어 중 어떤 단어를 입력해도 정답처리 해야함

5. 위와 같이 같은 뜻을 가진 여러 단어가 있는 경우, 같은 뜻을 가진 모든 영단어의 출제 빈도가 증가하며, 오답시 모든 영단어의 오답횟수도 증가해야 한다.

4)객관식 퀴즈-조건

1. 퀴즈는 4지선다형으로 10문제를 연속적으로 출제한다.

2. 저장된 단어 중 랜덤하게 4개 선정해서 4지 선다형으로 구성, 단 4개로 선정되어 퀴즈로 출제되었던 단어는 다음 퀴즈 문제 선정 시 다시 선택될 수 있다.

3. 4지 선다형으로 구성할 때, 같은 뜻이 여러 개 구성되면 안된다.

4. 퀴즈로 출제된 문제의 정답의 위치 번호가 고정되어 있으면 안된다.

5. 답을 입력할 때, 범위를 초과하거나 문자를 입력하였을 때는 오답으로 채점 한다.

6. 퀴즈로 출제된 단어의 출제빈도와 오답만 증가 시킨다.

7. 10문제를 다 풀었을 경우 다음과 같이 정답 횟수와 퀴즈를 푸는데 소요한 시간을 출력한다.

5)오답노트

1. 오답 횟수가 많은 단어부터 내림차순으로 출력
2. 오답은 1번 이상 틀린 경우를 말하며, 틀린 문제가 없을 경우 “틀린 문제가 없습니다”를 출력한다.

6)단어 검색

1.사용자가 입력한 단어를 등록된 단어장에서 찾아 검색하고, 만약 없다면 “단어장에 등록된 단어가 아닙니다.” 를 출력한다.

7)종료

1.프로그램을 종료시킨다.

|  |  |
| --- | --- |
| 2 | **주요 소스코드 설명** |

String eng;

String kor;

int frequent;

int wrong;

public Word(String eng, String kor) {

super();

this.eng = eng;

this.kor = kor;

}

Word 클래스의 멤버 구성

Eng : 영단어

Kor : 단어의 한글 뜻

Frequent : 출제 횟수

Wrong : 오답 횟수

String userName;

ArrayList <Word> voc = new ArrayList<>();

static Scanner *scan* = new Scanner(System.***in***);

static Random *rand* = new Random();

vocManager의 멤버 구성

username : 이름

voc : 단어장의 단어들을 저장할 리스트 변수

scan : 사용자로부터 입력을 받는 변수

rand : 랜덤 값을 할당하는 변수

public VocManager(String userName) {

super();

this.userName = userName;

}

void addWord(Word w) {

voc.add(w);

}

vocManager의 생성자와 voc 리스트에 w객체를 추가해주는 함수 addWord함수

void makeVoc(String fileName) {

try(Scanner file = new Scanner(new File(fileName));) {

while(file.hasNextLine()) {

String str = file.nextLine();

String[] result = str.split("\t");

this.addWord(new Word(result[0].trim(), result[1].trim()));

}

if(voc.size()>=10) {

System.***out***.println(this.userName+"의 단어장이 생성되었습니다.\n");

menu();

}else {

System.***out***.println("단어장이 올바르지 않습니다.");

System.*exit*(1);

}

} catch (FileNotFoundException e) {

System.***out***.println("파일을 찾을 수 없습니다.");

}

}

파일 경로를 입력 받고, 파일 스캐너를 try-catch로 시도해 만약 경로에 맞는 파일이 없을 경우 “파일을 찾을 수 없습니다.” 를 출력하게 하고 만약 파일이 있는 경우라면 while(hasNextLine)을 통해 한줄마다 단어와 뜻을 result 배열에 저장해 addWord함수를 통해 voc리스트에 단어와 뜻을 넣어 저장한다. 하지만 만약 단어장 파일의 단어 개수가 10개 미만이라면 퀴즈를 생성 할 수 없으므로 예외처리를 해주었다. 그런 다음 meun();를 통해 메인 메뉴를 호출하였다.

private void menu() {

while(true) {

System.***out***.println("------ "+userName+"의 영단어 퀴즈 ------");

System.***out***.println("1)주관식 퀴즈 2)객관식 퀴즈 3)오답노트 4)단어검색 5)종료");

System.***out***.print("메뉴를 선택하세요 : ");

int choice;

try{

choice = *scan*.nextInt();

*scan*.nextLine();

}catch (InputMismatchException e) {

choice = -1;

*scan*.nextLine();

}

switch(choice) {

case 1 -> subjectivityQuiz();

case 2 -> multipleChoiceQuiz();

case 3 -> answerNote();

case 4 -> searchVoc();

case 5 -> {

System.***out***.println("\n"+this.userName+"의 단어장 프로그램을 종료합니다."); return;}

}

}

}

무한 반복문을 통해 사용자가 5를 입력할 때 까지 프로그램이 종료되지 않도록 구현했고, try-catch를 통해 사용자가 실수로 숫자 대신 문자를 입력했더라도 예외처리를 해 프로그램이 중간에 에러가 나지 않도록 구현하였다 사용자가 입력한 번호에 따라서 각 함수별 호출을 통해 기능을 수행할 수 있도록 해주었다.

private void subjectivityQuiz() {

// **TODO** Auto-generated method stub

ArrayList <Word>checkList = new ArrayList<>();

int win = 0;

long beforeTime = System.*currentTimeMillis*();

for(int i=0;i<10;i++) {

System.***out***.println("------ 주관식 퀴즈 "+(i+1)+"번 ------");

int randIndex;

while(true) {

randIndex=*rand*.nextInt(voc.size());

if(!(checkList.contains(voc.get(randIndex)))) {

checkList.add(voc.get(randIndex));

break;

}

}

for(Word w : voc) {

if(voc.get(randIndex).kor.equals(w.kor)) {

w.frequent++;

}

}

System.***out***.println("\""+voc.get(randIndex).kor+"\"뜻을 가진 영어 단어는 무엇일까요?");

System.***out***.print("답을 입력하세요 : ");

String answer = *scan*.nextLine();

if(voc.contains(new Word(answer, voc.get(randIndex).kor))) {

System.***out***.println("정답입니다.");

win++;

}else {

System.***out***.println("틀렸습니다. 정답은 "+voc.get(randIndex).eng+"입니다.");

for(Word w : voc) {

if(voc.get(randIndex).kor.equals(w.kor)) {

w.wrong++;

}

}

}

}

long totalTime = (System.*currentTimeMillis*() - beforeTime) / 1000;

System.***out***.println(userName+"님 10문제 중 "+win+"개 맞추셨고, 총"+totalTime+"초 소요되었습니다.");

}

우선 10개의 문제에서 서로 단어 뜻이 중복되지 않아야 되므로 checkList라는 리스트를 만들어 중복을 체크해주고, int형 win이라는 변수와 beforetime이라는 게임 시작 전 시작을 체크해주는 변수를 만들어준다. for문을 통해 문제를 10번 출력해주고, while과 if문을 통해 랜덤 word 객체를 뽑고 현재까지 저장된 checkList의 중복여부를 체크해 문제들이 겹치지 않도록 해준다. For-each구문을 통해 같은 한국어 뜻이 중복되는 단어의 출제 횟수를 올려주고, 사용자가 영단어를 입력하면 새로운 word 객체를 만들어 contains통해 검사한다. 틀렸을 경우에는 출제 횟수를 for-each 구문을 통해서 같은 뜻을 가진 단어가 있으면 검사해 오답 횟수를 전부다 올려준다.

마지막으로 10번의 문제가 출력되고 현재시간-처음시작한 시간을 빼서 게임의 총 시간과 win 변수에 담긴 총 맞춘 개수를 출력해준다.

private void multipleChoiceQuiz() {

// **TODO** Auto-generated method stub

int win=0;

long beforeTime = System.*currentTimeMillis*();

for(int i=0;i<10;i++) {

ArrayList<Word> quizList = new ArrayList<Word>();

System.***out***.println("------ 객관식 퀴즈 "+(i+1)+"번 ------");

while(quizList.size() < 4) {

int tmp = 0; //같은 한글 뜻이 확인하는 변수

int randIndex = *rand*.nextInt(voc.size());

if(!quizList.contains(voc.get(randIndex))) {

for(Word w : quizList) {

if(w.kor.equals(voc.get(randIndex).kor)) {

tmp = 1;

}

}

if(tmp == 0) {

quizList.add(voc.get(randIndex));

}

}

}

int answer = *rand*.nextInt(4);

System.***out***.println(quizList.get(answer).eng+"의 뜻은 무엇일까요?");

quizList.get(answer).frequent++;

for(int j =1;j<=4;j++) {

System.***out***.println(j+") "+quizList.get(j-1).kor);

}

System.***out***.print("답을 입력하세요 : ");

int input;

try {

input = *scan*.nextInt();

*scan*.nextLine();

}catch(InputMismatchException e) {

input = -1;

*scan*.nextLine();

}

if(input-1 == answer) {

System.***out***.println("정답입니다.");

win++;

}else {

System.***out***.println("틀렸습니다. 정답은 "+(answer+1)+"번입니다.");

quizList.get(answer).wrong++;

}

}

long totalTime = (System.*currentTimeMillis*() - beforeTime) / 1000;

System.***out***.println(userName+"님 10문제 중 "+win+"개 맞추셨고, 총"+totalTime+"초 소요되었습니다.");

}

객관식 문제도 마찬가지로 함수로 구현해, 우선 맞춘개수를 담은 변수와 게임 시작한 시간을 담은 long타입 변수를 선언해주고 객관식 문제 10문제를 출력해야 하므로 for문을 통해 10번 반복해준다. 그 다음 for문안에 출제될 word객체를 담은 리스트를 선언해 주었는데 그 이유는 객관식 4문항으로 선정된 단어는 다음 문제에서도 출제될 수 있기 때문에 결국 4문항이 같은 영단어가 아니고, 같은 뜻이 겹치면 안되도록 if문으로 객체 검사를 하고, 그 안에서 for-each구문을 통해 뜻이 겹치지 않도록 검사해줬다. 이후 정답이 계속 같은 위치에 있으면 안되므로 random정수를 이용해 정답을 바꿔주었다. 사용자가 답을 입력할 때 1~4번을 제외한 정수나 문자열을 입력했을 때 문자열이 나오면 try-catch로 input에 -1값을 넣고 if문을 통해 answer에 담긴 값과 비교하여 정답 처리를 해주었다.

private void answerNote() {

// **TODO** Auto-generated method stub

ArrayList <Word> wrongList = new ArrayList<Word>();

for(Word w : voc) {

if(w.wrong>0) {

wrongList.add(w);

}

}

if(!wrongList.isEmpty()) {

for(Word w:wrongList) {

System.***out***.println(w+"\n-------------------");

}

}else {

System.***out***.println("\n틀린 문제가 없습니다.");

}

}

오답 노트에서는 우선 하나라도 틀린 word객체가 있는지 검사해야되므로 wrongList라는 리스트를 만들고 하나라도 틀린 word객체가 있다면 이 리스트에 추가했다. 만약 리스트가 비었다면 틀린 문제가 없다고 출력되고 비어있지 않다면 word객체를 출력하도록 구현하였다.

private void searchVoc() {

System.***out***.println("------ 단어 검색 ------");

System.***out***.print("검색할 영단어를 입력하세요 (영단어) : ");

String word = *scan*.nextLine();

word = word.trim();

boolean ch = false;

for(Word w : voc) {

if(w != null) {

if(w.eng.equals(word)) {

System.***out***.println(w);

ch = true;

break;

}

}

}

if(!ch) {

System.***out***.println("단어장에 등록된 단어가 아닙니다.");

}

System.***out***.println("---------------------\n");

}

}

우선 사용자로부터 영단어를 입력 받고 이를 for-each로 w객체에 담긴 영어와 비교하여 같다면 출력하고 ch변수를 true로 바꾼 다음 멈추게 했다 이후 ch가 false라면 단어장에 등록된 단어가 아니라고 출력했다.

|  |  |
| --- | --- |
| 3 | **실행결과** |

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

다른 단어장 사용하였을 때

텍스트, 스크린샷, 폰트, 메뉴이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디스플레이이(가) 표시된 사진

자동 생성된 설명

|  |  |
| --- | --- |
| 4 | **느낀점 및 토의사항** |

ArrayList의 사용법을 익힐 수 있었고, contains와 equals를 활용해 객체를 검사할 수 있다는 법도 배울 수 있었다. For-each구문을 활용해야 뜻만 검사하거나 영단어 자체만 검사할 수 있다는 점이 조금 불편하였다. 배열 대신에 여러 ArrayList같은 이미 자바에 구현해 놓은 자료구조를 활용해 더 편리하게 할 수 있다는 점도 배웠다. 또한 파일을 읽어 형식만 같다면 다양한 단어장을 불러와 활용할 수 있다는 점도 배웠다.

|  |  |
| --- | --- |
| 5 | **과제 수행 완료 체크 리스트** |

<체크리스트>

1. 학번 이름은 출력하였는가? **예**

2. 사용자 입력 실수, 파일 오류 등을 고려한 예외처리를 수행하였는가? **예**

3. 텍스트 파일을 읽어서 Word 객체를 생성한 후, 컬렉션 클래스의 저장구조에 잘 저장하였는가? **예**

4. 파일에 저장된 패턴(영어와 단어가 나열된 형식)이 동일한 파일도 읽어서 처리 할 수 있는가? **예**

5. 메뉴는 5번 종료 메뉴를 수행할 때까지 반복 수행되는가? **예**

6. 1번 주관식 퀴즈 메뉴 관련

A. 퀴즈 10문제가 중복되지 않도록 처리하였는가? **예**

B. 정답을 정확하게 체크하는가? **예**

C. 뜻이 같은 단어에 대한 정답처리를 잘 하였는가? **예**

D. 10문제가 연속으로 출제되는가? **예**

E. 10문제의 퀴즈를 푼 시간을 측정하여 출력하였는가? **예**

F. 10문제 퀴즈 채점한 결과가 출력되었는가? **예**

G. 퀴즈 출제된 단어의 출제회수가 누적되는가? **예**

H. 오답시 오답회수가 누적되는가? **예**

I. 뜻이 같은 단어에 대한 출제회수 처리는 잘 하였는가? **예**

J. 뜻이 같은 단어에 대한 오답회수 처리를 잘 하였는가? **예**

7. 2번 객관식 퀴즈 메뉴 관련

A. 퀴즈의 선지로 선택된 4개의 영어 단어에는 중복이 없는가? **예**

B. 선지를 구성하는 4개의 영어 단어의 뜻도 중복체크 하였는가? **예**

C. 정답의 위치는 랜덤하게 설정하였는가? **예**

D. 사용자가 답을 입력하면 정확하게 정답과 오답을 출력하는가? **예**

E. 10문제가 연속으로 출제되는가? **예**

F. 10문제의 퀴즈를 푼 시간을 측정하여 출력하였는가? **예**

G. 10문제 퀴즈 채점한 결과가 출력되었는가? **예**

H. 퀴즈 출제된 단어의 출제회수가 누적되는가? **예**

I. 퀴즈에서 틀린 단어에 대한 오답처리가 제대로 되는가? **예**

J. 오답 회수는 누적되는가? **예**

8. 3번 오답노트 메뉴 관련

A. 퀴즈에서 틀린 문제는 오답노트에 잘 저장하였는가? **예**

B. 오답회수가 가장 높은 단어순으로 출력되는가? **예**

C. 틀린 단어가 없는 경우에 대한 처리도 하였는가? **예**

9. 4번 단어 검색 메뉴 관련

A. 입력된 단어를 찾은 경우 단어의 뜻을 보여주는가? **예**

B. 단어장에 등록되지 않은 단어의 경우 처리를 하였는가? **예**

10.보고서 관련

A. 모든 실행결과를 잘 보여줄 수 있도록 체크리스트에 해당하는 모든 상황에 대한 실행결과를 보고서에 담았는가? **예**

B. 보고서에는 체크리스트가 작성되어 있는가? **예**