JAVA 학생관리프로그램

백앤드 개발자 부트캠프(스프링) 27주차

제출일자 : 2024년 8월 22일 (목)

김명찬

목차

A. 구조

B. 기능

A. 구조

```java
public interface StudentInformation {  1 implementation
    //기능
    void addStudent(Student std);  //학생등록  1 implementation
    void searchAll(); //전체조회  1 implementation
    void searchStudent(int studentNum); //학생조회  1 implementation
    void setInformation(int studentNum , int count , Scanner scan);//정보수정  1 implementation

    //예외처리
    int checkStudentNum(Scanner scan);  //학번 예외처리  1 implementation
    String checkPhoneNum(Scanner scan); //전화번호 예외처리  1 implementation
    String checkName(Scanner scan);     //이름 예외처리  1 implementation
    public int checkNum(Scanner scan);  //관리번호 예외처리  1 implementation
    public int checkNull(Student[] student);  1 implementation
```

StudentInformation 인터페이스

코드의 간결함과 가독성 그리고 특정 기능을 수정하거나 확장해야 할 경우 구현한 클래스의 해당 메소드만 확인하면 되기에 유지보수가 쉬워 인터페이스를 만들어 해당 프로그램에서 필수로 구현해야 하는 메소드를 정의했습니다.

Student 클래스

```
public Student(String studentName, String phoneNum, int studentNum, String major) {
    this.studentName=studentName;
    this.phoneNum=phoneNum;
    this.studentNum=studentNum;
    this.major=major;

}
```

Student 클래스 생성자

학생이름(studentName) , 전화번호(phoneNum) , 학번(studentNum) , 학과(major)을 생성자를 통해 받아 생성된 Student 객체의 필드에 각각 저장합니다.

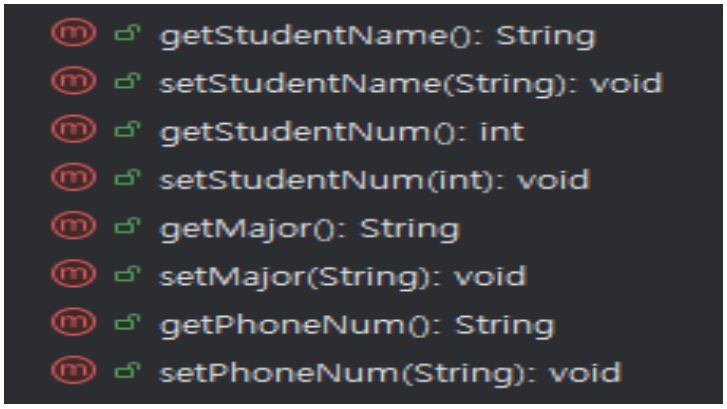
```
private String studentName;   //학생이름
private String phoneNum;  //학생 전화번호
private int studentNum;       //학번
private String major;     //학과
```

studentName : 학생이름을 담기위한 필드 (String 타입)

phoneNum : 학생 전화번호를 담기 위한 필드 (String 타입)

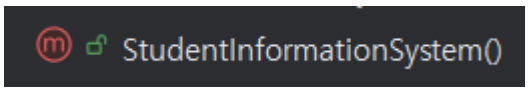
studentNum : 학번을 담기위한 필드 (int 타입)

major : 학생의 학과를 담기위한 필드 (String 타입)



필드들을 private으로 지정했기 때문에 외부에서 객체를 수정(set)하기 위한 메소드들과 필드를 얻기 위한 (get)메소드들을 구현했습니다.

StudentInformationSystem 클래스



클래스의 기본 생성자

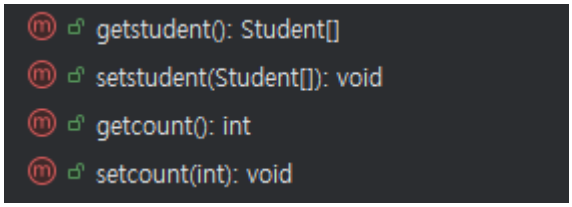
```
private Student[] student= new Student[25];  //25명으로 제한
private int count=0;  //student 배열을 위한 인덱스
```

학생들의 정보를 담기 위한 Student 타입 배열

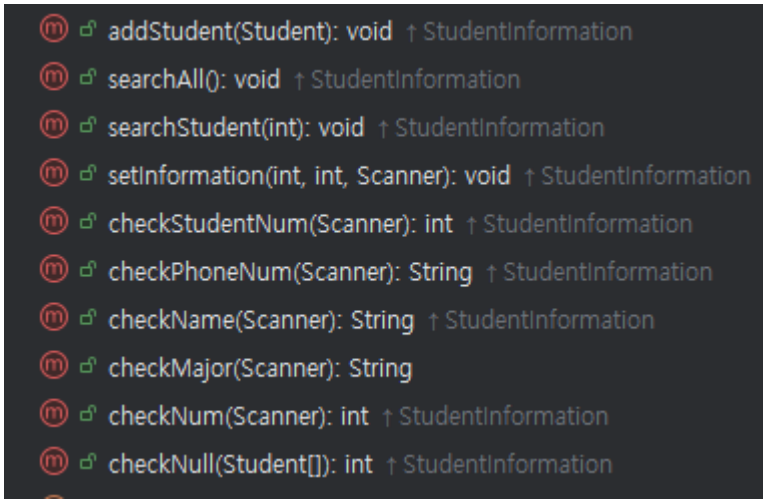
Student 타입 배열의 인덱스를 위한 count 필드

```
Scanner scan = new Scanner(System.in);
```

입력을 위한 Scanner 객체생성



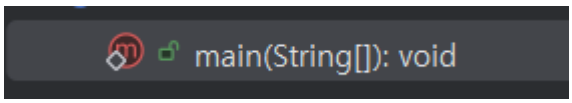
배열과 필드를 private로 지정했기 때문에 외부에서 접근하기 위한 메소드



요구하는 기능들 (1.학생등록 , 2.전체조회 , 3.학생조회 , 4.정보수정)에 대한 메소드

예외처리를 위한 메소드(5.학번 예외처리 , 6.전화번호 예외처리 , 7.이름 예외처리 , 8.학과 예외처리 , 9. 관리번호 예외처리 10. 등록된 학생이 없을 때 예외처리)

StudentInformationMain 클래스



구현했던 기능들을 통합하여 실행하는 메인 클래스

B. 기능

1. 학생등록

public void addStudent(Student student)

학번 , 이름 , 학과 , 전화번호를 입력받아 Student 객체의 생성자 매개변수로 받아 Student 객체로 만든 후 Student 타입의 배열에 객체 저장

2. 전체 조회

```
**** 학생 관리 프로그램 ****
1. 학생 등록
2. 전체 출력
3. 학생 조회
4. 정보 수정
5. 프로그램 종료
관리 번호를 입력하세요. : 2
======전체 학생 출력======
학 번 : 20241111
이 름 : 홍길동
학 과 : 빅데이터
연락처 : 010-1111-2222
=====================================
학 번 : 20242222
이 름 : 김자바
학 과 : 빅데이터
연락처 : 010-2222-3333
=====================================
```

public void searchAll()

Student 타입의 배열이 null 이 아니면 향상된 반복문을 통해 배열에 있는 모든 학생들의 정보들을 전부 출력

3. 학생 조회

```
**** 학생 관리 프로그램 ****
1. 학생 등록
2. 전체 출력
3. 학생 조회
4. 정보 수정
5. 프로그램 종료
관리 번호를 입력하세요. : 3
[학생을 조회 합니다.]
학번을 입력하십시오. : 20241111
학   번 : 20241111
이   름 : 홍길동
학   과 : 빅데이터
연락처   : 010-1111-2222
```

public void searchStudent(int studentNum)

학번을 매개변수로 받아 향상된 반복문을 통해서 입력받은 학번과 배열 내부에 있는 Student 객체의 학번이 일치하면 해당 학생의 정보 전부 출력

4. 정보 수정

```
**** 학생 관리 프로그램 ****
1. 학생 등록
2. 전체 출력
3. 학생 조회
4. 정보 수정
5. 프로그램 종료
관리 번호를 입력하세요. : 4
[학생 정보를 수정합니다.]
학번을 입력하십시오. : 20241111
학   번 : 20241111
이   름 : 홍길동
학   과 입력 : 컴퓨터공학과
전화번호 입력 : 010-1234-5678
```

setInformation(int studentNum , int count , Scanner scan)

학번(studentNum) , 수정하려고 하는 학생이 저장되어 있는 배열의 인덱스(count) , 입력받기 위한 Scanner 객체)를 매개변수로 받아 해당 학생의 학과 전화번호 수정(setMajor , setPhoneNum 사용

5. 프로그램 종료

```
**** 학생 관리 프로그램 ****
1. 학생 등록
2. 전체 출력
3. 학생 조회
4. 정보 수정
5. 프로그램 종료
관리 번호를 입력하세요. : 5
프로그램을 종료하시겠습니까? (y/n) : y
[프로그램을 종료합니다.]

Process finished with exit code 0
```

```
if(a == 5) {
    System.out.print("프로그램을 종료하시겠습니까? (y/n) : ");

    String sys = scan.nextLine();

    if("y".equals(sys.trim())) {  //trim()은 앞 뒤 공백을 지워줌.
        System.out.println("[프로그램을 종료합니다.]");
        System.exit( status: 0);
    }

    else {
        continue;
    }
}
```

입력받은 값이 문자열 y 또는 Y와 일치했을 때 프로그램 종료 문구와 함께 종료

n 입력 시 프로그램 그대로 실행

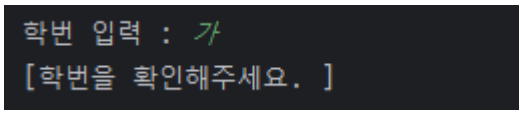
6. 예외처리

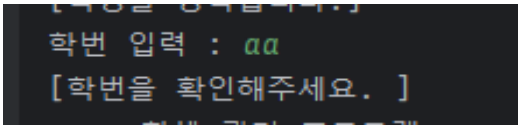
학번 예외처리

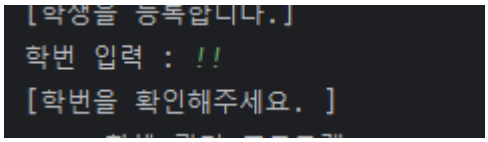
public int checkStudentNum(Scanner scan)

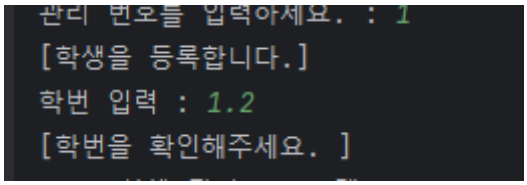
```
String strnum = Integer.toString(studentNum);  //학번을 문자열로 변환해서 길이 확인
if(strnum.length() !=  8 || (studentNum < 0 )) { //학번(8자리) 입력이 아니거나 음수인 경우
    return -1;
}

return studentNum;
```

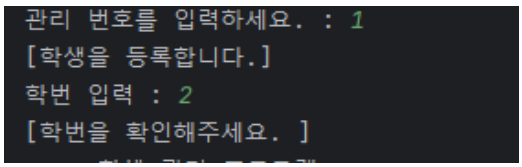
```
catch(InputMismatchException e) { //int 타입과 다른 타입이 입력된 경우
    scan.nextLine();
    return -1;
}
```

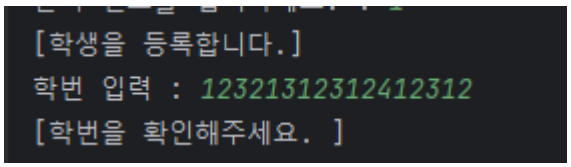












학번 8자리 입력이 되지 않았을 때와 음수가 입력되었을 때 학번 타입(int) 과 다른 타입이 입력이 되었을 때 예외처리

전화번호 예외처리

public String checkPhoneNum(Scanner scan)

```
전화번호 입력 : 010-1111-2
[ 잘못된 전화번호 입력 ]
```

```
//정규 표현식에서 \d 는 숫자이지만 자바에서는 이스케이프 문자 때문에 \\d 써야 숫자로 인식됨

String PhoneNum = scan.nextLine();

String pattern="^\\d{3}-\\d{4}-\\d{4}$"; //정규 표현식 패턴 설정
//{3자리}-{4자리}-{4자리}
// ^ : 문자열로 시작한다는 의미 \d : 숫자
//Pattern 은 정적메소드를 제공하기 때문에 따로 객체 생성없이 matches 메소드를 사용할 수 있음.
 if(!Pattern.matches(pattern, PhoneNum)) { //정규 표현식(pattern)과 일치하지 않으면
     return null;
 }
```

```
if(phonenum == null) {
    System.out.println("[ 잘못된 전화번호 입력 ]");
    continue;
}
```

```
전화번호 입력 : 가
[ 잘못된 전화번호 입력 ]
```

```
전화번호 입력 : aa
[ 잘못된 전화번호 입력 ]
**** 학생 관리 프로그램 ****
```

```
전화번호 입력 : !!
[전화번호를 확인해주세요.]
```

```
전화번호 입력 : 1.2
[전화번호를 확인해주세요.]
**** 학생 관리 프로그램 ****
```

```
전화번호 입력 : 2
[전화번호를 확인해주세요.]
```

```
전화번호 입력 :
[ 잘못된 전화번호 입력 ]
```

입력받은 값이 정규 표현식 패턴에 일치하지 않으면 null을 반환

이름 예외처리

public String checkName(Scanner scan)

```
이  름 입력 : abc
[ 잘못된 이름 입력 ]
```

```
String studentName = scan.nextLine();
String pattern = "^[가-힣]{2,5}$";  //정규 표현식 패턴 설정 한글 2자이상 5자 미만


if(!Pattern.matches(pattern , studentName)) {  //정의한 정규 표현식에 맞지 않을 때
    return null;
}
```

```
if(stdname == null) {
    System.out.println("[ 잘못된 이름 입력 ]");
    continue;
}
```

```
이  름 입력 : 가
[ 잘못된 이름 입력 ]
```

```
이  름 입력 : aa
[ 잘못된 이름 입력 ]
```

```
이  름 입력 : !!
[ 잘못된 이름 입력 ]
```

```
이  름 입력 : 1.2
[ 잘못된 이름 입력 ]
```

```
이  름 입력 : 2
[ 잘못된 이름 입력 ]
```

```
이  름 입력 : 라이언어피치
[ 잘못된 이름 입력 ]
```

```
이  름 입력 :
[ 잘못된 이름 입력 ]
```

정규 표현식 패턴에 일치하지 않으면 null 반환

학과 예외처리

public String checkMajor(Scanner scan)

```
학  과 입력 : abcdc
[ 잘못된 학과 입력 ]
```

```
String studentMajor = scan.nextLine();
String pattern = "^[가-힣]{3,}$";   // 3, -> 최소 3자 이상이면 ok


if(!Pattern.matches(pattern , studentMajor)) {
    return null;
}
```

```
if(major == null) {
    System.out.println("[ 잘못된 학과 입력 ]");
    continue;
}
```

```
학  과 입력 : 가
[학과를 확인해주세요.]
```

```
학  과 입력 : aa
[학과를 확인해주세요.]
```

```
학  과 입력 : !!
[학과를 확인해주세요.]
```

```
학  과 입력 : 1.2
[학과를 확인해주세요.]
```

```
학  과 입력 : 학과
[학과를 확인해주세요.]
```

```
학  과 입력 : 1학과
[학과를 확인해주세요.]
```

```
학  과 입력 : 빅데 이터
[학과를 확인해주세요.]
```

```
학  과 입력 :
[ 잘못된 학과 입력 ]
```

정규 표현식의 패턴과 일치하는 입력값이 아니면 null 반환

관리번호 예외처리

public int checkNum(Scanner scan)

```
관리 번호를 입력하세요. : aa
[다시 입력해주세요.]
```

```
    try{
        int studentNum = scan.nextInt();
        scan.nextLine();

        if(studentNum < 0 || studentNum > 5) {  //관리번호가 음수이거나 5보다 클 때
            return -1;
        }
        return studentNum;
    }
    catch (InputMismatchException e) {
        scan.nextLine();
        return -1;
    }
}
```

```
if(a == -1) {
    System.out.println("[다시 입력해주세요.]");
    continue;
}
```

```
관리 번호를 입력하세요. : 가
[다시 입력해주세요.]
```

```
관리 번호를 입력하세요. : aa
[다시 입력해주세요.]
```

```
관리 번호를 입력하세요. : !!
[다시 입력해주세요.]
```

```
관리 번호를 입력하세요. : 1.2
[다시 입력해주세요.]
```

```
관리 번호를 입력하세요. : -1
[다시 입력해주세요.]
```

```
관리 번호를 입력하세요. : 6
[다시 입력해주세요.]
```

관리번호가 음수이거나 5보다 클 때 -1 을 반환하고

관리번호(int 타입)와 다른 타입이 입력되었을 때 -1 반환

등록된 학생이 없을 때 예외처리

public int checkNull(Student[] student)

```
**** 학생 관리 프로그램 ****
1. 학생 등록
2. 전체 출력
3. 학생 조회
4. 정보 수정
5. 프로그램 종료
관리 번호를 입력하세요. : 2
[등록된 학생이 없습니다.]
```

```
if(nullFlag == -1) {
    System.out.println("[등록된 학생이 없습니다.]");
    continue;
}
```

(Student 타입) 학생 배열을 매개변수로 받습니다. 받은 배열을 advanced for문을 통해 전체 배열을 탐색합니다. 배열 안에 한 명이라도 저장되어 있다면 nullFlag에 1을 저장합니다. 최종적으로 return nullFlag 를 합니다.

Reference

교재 pdf

https://moonong.tistory.com/31

https://inpa.tistory.com/entry/JAVA-%E2%98%95-%EC%A0%95%EA%B7%9C%EC%8B%9DRegular-Expression-%EC%82%AC%EC%9A%A9%EB%B2%95-%EC%A0%95%EB%A6%AC