5/3 스터디노트

➢ P344 Arrays 클래스

✓ 배열의 비교나 배열의 정렬, 배열의 내용을 특정 값으로 채우는 static 메서드 제공

메서드	기능
asList([배열])	매개변수의 배열을 List 객체로 리턴
binarySearch([배열], key)	이진 검색으로 arr배열에서 key를 찾아 index 값 리턴
equals([배열1], [배열2])	배열1과 배열2를 비교하여 같으면 true 리턴
fill([배열], val)	배열의 요소를 val로 모두 채움
fill([배열], 시작인덱스, 끝인덱스, val)	배열의 시작인덱스부터 끝인덱스 전까지 val 로 모두 채움
sort([배열])	배열을 오름차순으로 정렬

✓ 예제

```
import java.util.Arrays;
public class ArraysEx {
    public static void main(String[] args) {
        String[] arr = {"홍길동", "이순신", "강감찬", "김유신"};
        Arrays.fill(arr, val: "임꺽정");
        for(String n : arr) System.out.print(n + ",");
        System.out.println();
        Arrays.fill(arr, fromIndex: 1, toIndex: 3, val: "X");
        for(String n : arr) System.out.print(n + ",");
    }
}
```

```
임꺽정,임꺽정,임꺽정,임꺽정,
임꺽정,X,X,임꺽정,
```

● P345 java.text 패키지

✓ 숫자 55000 을 금액으로 표시할 때 55,000원 / 날짜 2023-05-02 를 보기좋게 2023년 5월 2일 로 표시

✓ 출력형태의 포맷을 지정할 수 있는 클래스 제공

➢ P345 DecimalFormat 클래스

✓ 숫자포멧지정

```
DecimalFormat df1 = new DecimalFormat( pattern: "###,###.##");
DecimalFormat df2 = new DecimalFormat( pattern: "000,000.00");

System.out.println(df1.format( number: 5500));
System.out.println(df2.format( number: 5500));
```

```
5,500
005,500.00
```

➔ # 은 해당 자리에 숫자가 있으면 표시하고, , 와 . 이 있어야 할 자리 지정

➔ 0 은 해당 자리에 숫자가 없어도 0을 표시하고, , 와 . 이 있어야 할 자리 지정

➢ P346 SimpleDateFormat 클래스 : Date 클래스나 Calendar 클래스에서 포멧 지정할 수 있음.

✓ 최근 LocatDate, LocalTime, LocalDateTime 의 포멧은 java.time.format 패키지의 DateTimeFormatter 클래스를 사용함 (5/2스터디노트)

● P349 java.util.regex 패키지

✓ 자바에서 정규 표현식을 사용하기 위한 클래스를 모아 놓은 패키지

✓ 정규 표현식 : 데이터가 아주 많고 복잡할 때 정규식을 이용하면 짧은 코드로 세부적인 조건을 걸어서 원하는 데이터를 추출할 수 있음.

✓ 예시: 분량이 A4 용지 3장 이상되는 긴 글에서

1) "과메기"라는 단어를 모두 검색할 건데, 그 중에서 "포항"이라는 단어와 함께 쓰인 "과메기"만 검색
2) 알파벳 un 으로 시작하는 단어 모두 검색
...

➔ 파이썬에서 자세히 배우겠습니다.

- 선생님 Array 예제

```java
import java.util.Arrays;

public class ArraysEx_hyunkoo {
    public static void main(String[] args) {
        int[] iValues = {5,2,9,1,5,6};
        String[] names = {"홍길동", "이영희", "박철수", "정지영", "김민수", "최민호", "한지영"};

        //숫자 배열 정렬하기
        Arrays.sort(iValues);
        //정렬된 배열 출력하기
        System.out.println("* 숫자 타입 배열값 정렬 결과");
        for (int iValue: iValues){ System.out.print(iValue+" "); }

        //문자열 배열 정렬하기
        Arrays.sort(names);
        //정렬된 매열 출력하기
        System.out.println("* 문자 타입 배열값 정렬 결과");
        for (String name:names){ System.out.print(name+" "); }
    }
}
```

```
* 숫자 타입 배열값 정렬 결과
1 2 5 5 6 9

* 문자 타입 배열값 정렬 결과
김민수 박철수 이영희 정지영 최민호 한지영 홍길동
```

- P355 chapter 12 연습문제

✓ 1번 : 모든 클래스의 최상위 클래스는 ? Object

✓ 2번 : "현대자동차:그랜져" 가 출력될 수 있도록 Car 클래스에 코드를 작성하세요

```
class Car {
    String name;
    String company;

    @Override
    public String toString(){
        return company+":"+name;
    }
}
public class Excercise2 {
    public static void main(String[] args) {
        Car car = new Car();
        car.name = "그랜져";
        car.company = "현대자동차";

        System.out.println(car);
    }
}
```

➔ Car 클래스의 객체를 출력했을 때 company 변수와 name 변수값이 나오도록 toString( ) 메서드를 오버라이딩

✓ 3번 : 두 개의 문자열 변수의 합계를 구하기 위해 숫자로 변환하여 덧셈 연산 값을 출력하는 코드 작성

```
public class Excercise3 {
    public static void main(String[] args) {
        String num1 = "100";
        String num2 = "200";

        System.out.print(num1+"+"+num2+"="+(Integer.parseInt(num1)+Integer.parseInt(num2)));
    }
}
```

```
100+200=300
```

✓ 4번 : 다음 예제는 문자열 배열에 "아이디, 이름, 나이" 형태로 저장되어 있고, 데이터는 콤마(,)로 구분되어 있다.
이 데이터 중 이름만 출력되도록 코드를 작성하고, 전체 데이터의 평균 나이를 출력하는 코드를 작성하시오.

```
public static void main(String[] args) {
    String[] member = {
            "hong,홍길동,30",
            "lee,이순신,40",
            "kim,김유신,50"
    };
    //이름만 출력하기
    for (int i = 0; i < member.length; i++) {
        System.out.println(member[i].split( regex: ",")[1]);
    }

    //평균 나이 출력하기
    int ageSum=0;
    for (int i = 0; i < member.length; i++) {
        ageSum += Integer.valueOf(member[i].split( regex: ",")[2]);
    }
    System.out.println("평균 나이: "+(ageSum/member.length));

}
```

정답

```
//평균 나이 출력하기
int ageSum=0;
for (int i = 0; i < member.length; i++) {
    ageSum += Integer.parseInt(member[i].split( regex: ",")[2]);
}
System.out.println("평균 나이: "+(double)(ageSum/member.length));
```

```
홍길동
이순신
김유신
평균 나이: 40
```

Chapter 13 컬렉션 프레임 워크

- ✓ 지금까지 데이터를 저장할 때 한가지 자료형으로만 저장했음.
- ✓ 예를 들면 name 변수를 String 으로 선언하여 name에는 문자열 자료형만 저장하고, (String name = "홍길동";)
  korean 변수를 int 로 선언하여 korean에는 정수형 자료형만 저장했음. (Int korean = 90;)
  name 변수와 korean 변수를 Student 라는 클래스에 선언함으로써 여러가지 자료형 데이터를 저장할 수 있음.
  한명의 데이터는 Student s1 = new Student("홍길동", 90); 으로 Student 클래스에 객체를 생성해서 저장했고
  다수의 데이터는 배열을 이용해서 저장했음 (String[] names = {"홍길동", "김철수", "이영희"};)

  → 이렇게 데이터를 처리하는 시스템(자료구조)를 레파지토리(Repository) 라고 부른다. 그중 배열로 처리하는 것은 "메모리 레파지토리"라고 함.
  상용레파지토리를 처리하는 시스템을 DBMS 라고 부르며 대표적인 DBMS 제품은 ORACLE 이 있음.

- ✓ 데이터의 성격에 따라 레파지토리 기능을 확장한 것이 컬렉션(Collection) 이라는 인터페이스.
- ✓ 컬렉션을 사용하면 DBMS 제품을 사용하지 않고 자바에서 배열보다 편하게 다수의 데이터를 다룰 수 있음
- ✓ 컬렉션 하위에 2가지의 인터페이스가 있고(List / Set), 그 외에 Map이라는 인터페이스도 있음.(Map 은 컬렉션 인터페이스의 하위 X)

- ✓ Collection 인터페이스의 주요 메서드
  (collection 인터페이스 하위의 List 와 Set 인터페이스를 구현하는 클래스들(ArrayList, Stack, HashSet, TreeSet 등)은 해당 메서드를 모두 구현함.)
- ✓ P361 아래 표 : add 추가 / clear 모든 객체 삭제 / contains 매개변수 포함여부 / equals 동등비교 / isEmpty 비어있는지 확인 / 등...

- **P362 List 인터페이스 :** [배열과 Collection 프레임워크의 List 비교해보기]

배열로 데이터를 생성하면 생성시에 `String[] names = new String[5];` 배열의 사이즈를 미리 정해야하며

```
String[] names = {"홍길동", "김철수", "이영희", "박종수", "최지수"};
int[] koreanScores = {80, 90, 70, 85, 95};
int[] englishScores = {85, 85, 75, 95, 90};
int[] mathScores = {90, 70, 80, 95, 85};

for (int i = 0; i < names.length; i++) {
    students[i] = new Student(names[i], koreanScores[i], englishScores[i], mathScores[i]);
    numStudents++;
}
```

데이터 박민지, 90, 84, 75 를 추가하고 싶으면 추가할 위치의 인덱스를 알아야 추가할 수 있음 (names[5] = "박민지";)

Collection 인터페이스 하위의 **ArrayList** 클래스에서는 미리 사이즈 정하지 않아도 되며, add 메서드를 이용하여 인덱스를 몰라도 데이터 추가 가능함.

```
import java.util.List;
```

```
//List list = new ArrayList();  -> List인터페이스로 객체 선언
ArrayList list = new ArrayList(); //List인터페이스를 구현한 ArrayList클래스로 객체선언 (권장)
list.add(1);
list.add(2);
list.add(3);
list.add(4);
list.add(5);
list.add(6);
System.out.println(list);
```

`[1, 2, 3, 4, 5, 6]`

사이즈를 조절할 필요 없이 그냥 `list.add(7);` 을 추가로 입력하면 됨.

배열이었으면 사이즈 조절하지 않고 추가하면 배열 범위 넘는 예외가 발생할 것

✓ P364 Vector 클래스 예제

Collection 프레임워크 이전에 사용되었던 클래스로, 전에 쓰던 메서드와 List 인터페이스의 구현메서드가 혼합되어 있음.

ArrayList, Vector 클래스 모두 동일한 자료구조, 메서드를 가지고 있으며 성능은 Multi Thread환경에서 Vector 클래스가 더 좋다.

그 외에 대부분의 경우에서 ArrayList 로 구현한다.

- List 인터페이스 교재 예제 (P365-P368)

```java
//추상클래스 Shape
abstract class Shape {
//필드
    int x, y;
//생성자
    Shape() { this( x: 0, y: 0); }
    Shape(int x, int y) {
        this.x = x;
        this.y = y;
    }
//추상메서드
    abstract double area();
    abstract double length();
//메서드
    public String getLocation(){
        return "x:"+x+"y:"+y;
    }
}
```

```java
//추상클래스인 Shape 를 상속받는 Circle 클래스
public class Circle extends Shape{
//필드
    double r; //반지름
//생성자
    Circle(){ this( r: 1); }
    Circle(double r){ this.r = r; }
//Shape의 추상메서드 오버라이딩(재정의)
    @Override
    double area(){ return (r*r)*Math.PI; }  //넓이
    @Override
    double length(){ return(r*2)*Math.PI; } //둘레
}
```

```java
//추상클래스 Shape를 상속받는 Rectangle 클래스
public class Rectangle extends Shape{
//필드
    int w,h;  //가로,세로
//생성자
    Rectangle(){ this( w: 1, h: 1); }
    Rectangle(int w, int h){ this.w = w; this.h = h; }
//Shape의 추상메서드 오버라이딩(재정의)
    @Override
    double area(){ return (w*h); }  //넓이
    @Override
    double length(){ return (w+h)*2; }  //둘레
}
```

```java
import java.util.ArrayList;
import java.util.List;
public class ListEx3 {
    public static void main(String[] args) {
        ArrayList list = new ArrayList();
        list.add(new Circle( r: 3.0));
        list.add(new Rectangle( w: 3,  h: 4));
        list.add(new Circle( r: 5));
        list.add(new Rectangle( w: 5,  h: 6));

        System.out.println("전체 도형의 면적의 합 : "+sumArea(list));
        System.out.println("전체 도형의 둘레의 합 : "+sumLength(list));

    }

    private static double sumLength(List list){
        double sumlength=0;
        for (int i = 0; i < list.size(); i++) {
            sumlength += ((Shape)list.get(i)).length();
        } return sumlength;
    }
    private static double sumArea(List list){
        double sumArea=0;
        for (int i = 0; i < list.size(); i++) {
            sumArea += ((Shape)list.get(i)).area();
        }
        return sumArea;
    }
}
```

```
전체 도형의 면적의 합 : 148.81415022205297
전체 도형의 둘레의 합 : 86.26548245743669
```

```java
for (int i = 0; i < list.size(); i++) {
    sumArea += ((Shape)list.get(i)).area();
}
```

1. `list.get(i)` 로 list 에서 i번째 값을 가져와서
2. `(Shape)` 로 부모형인 Shape 클래스로 형변환
3. `area();` 메서드 시행 (자식클래스의 메서드로 시행됨)

   ➔ Circle 클래스 객체는 `r*r*PI`,

   Rectangle 클래스 객체는 `w*h`

- 선생님 예제1

```
ArrayList list = new ArrayList();
list.add("Hello");
list.add(123);
list.add(true);

for (int i = 0; i < list.size(); i++) {
    Object obj = list.get(i);
    if(obj instanceof String){
        System.out.println("String: "+obj);
    } else if (obj instanceof Integer){
        System.out.println("Integer: "+obj);
    } else if (obj instanceof Boolean){
        System.out.println("Boolean: "+obj);
    }
}
```

배열과 달리

ArrayList 는 add 메서드를 이용하여

여러가지 자료형의 데이터를 자유롭게 추가할 수 있음.

여러 자료형을 다 받기 위해 Object 클래스 이용

```
String: Hello
Integer: 123
Boolean: true
```

- 선생님 예제2 : 아래의 다섯가지 메뉴를 제공하는 학생정보관리 프로그램 코드 작성하기

  전체 학생정보를 출력하는 메뉴 / 평균점수가 제일 높은/낮은 학생을 찾는 메뉴 / 학생 이름으로 검색하는 메뉴 / 학생을 추가하는 메뉴

[Student 클래스]

```java
public class Student {
//필드
    private static int nextId = 1;
    private int id;
    private String name;
    private int koreanScore;
    private int englishScore;
    private int mathScore;
//생성자
    public Student(String name, int koreanScore, int englishScore, int mathScore) {
        this.id = nextId++;
        this.name = name;   this.koreanScore = koreanScore;   this.englishScore = englishScore;  this.mathScore = mathScore;
    }
//getter
    public int getId() { return id; }
    public String getName() { return name; }
    public int getKoreanScore() { return koreanScore; }
    public int getEnglishScore() { return englishScore; }
    public int getMathScore() { return mathScore; }
//메서드
    //평균점수구하기
    public double getAverage(){ return (koreanScore+englishScore+mathScore)/3.0; }
    //학생정보출력하기
    public String toString(){
        return name+"\t"+id+"\t"+koreanScore+"\t"+englishScore+"\t"+mathScore+"\t"+String.format("%.2f",getAverage());
    }
}
```

[메인 클래스 구조]

```
public static void main(String[] args) {
    initilizeStudents();
    showMenu();
}

//5명의 임의 학생 데이터 생성
private static void initilizeStudents(){...}

//메뉴 출력 및 기능 실행
private static void showMenu(){...}

//1. 전체 학생 정보 출력
private static void printAllStudents(){...}

//2. 최고 평균점수 학생 검색
private static void searchHighestAverageStudent(){...}

//3. 최저 평균점수 학생 검색
private static void searchLowestAverageStudent(){...}

//4. 학생 검색
private static void searchStudent(){...}

//5. 학생 추가
private static void addStudent(){...}
```

프로그램 실행시
먼저 실행되는 메서드

각 메뉴의 기능을 수행하는
메서드 작성

➔ **배열을 이용할때와 ArrayList 클래스를 이용할 때 코드가 어떻게 달라지는 지 확인해보기**

[배열을 사용하여 프로그램 작성]

- 메인 메서드 시행전 배열 관련 필드 선언

```java
public class StudentManagement1 {
    private static final int MAX_STUDENTS = 100;
    private static Student[] students = new Student[MAX_STUDENTS];
    private static int numStudents = 0;

    public static void main(String[] args) {
        initilizeStudents();
        showMenu();
    }
```

배열은 사이즈가 미리 정해져야 하므로 상수 MAX_STUDENTS 를 선언한다.

-> 100으로 선언했기 때문에 101명째부터는 학생 추가 불가능.

- 메인 메서드 첫번째 시행 메서드 : 학생 5명 임의 데이터 생성

```java
private static void initilizeStudents(){
    String[] names = {"홍길동", "김철수", "이영희", "박종수", "최지수"};
    int[] koreanScores = {80,90,70,85,95};
    int[] englishScores = {85, 85, 75, 95, 90};
    int[] mathScores = {90, 70, 80, 95, 85};

    for (int i = 0; i < names.length; i++) {
        students[i] = new Student(names[i], koreanScores[i], englishScores[i], mathScores[i]);
        numStudents++;
    }
}
```

for 문 사용하여 인덱스 0부터 names 배열의 길이 (5명) 까지

students 배열에 학생 정보 넣고, 학생수 변수 (numStudents)를 +1 해줌

[ArrayList 를 이용하여 프로그램 작성]

- 메인 메서드 시행전 ArrayList 관련 필드 선언

```java
import java.util.ArrayList;
public class StudentManagement2 {
    private static ArrayList studentList = new ArrayList();

    public static void main(String[] args) {
        initializeStudents();
        showMenu();
    }
```

ArrayList 는 사이즈를 미리 정하지 않아도 되므로 객체만 생성해줌

: studentList

- 메인 메서드 첫번째 시행 메서드 : 학생 5명 임의 데이터 생성

```java
private  static void initializeStudents() {
    String[] names = {"홍길동", "김철수", "이영희", "박종수", "최지수"};
    int[] koreanScores = {80, 90, 70, 85, 95};
    int[] englishScores = {85, 85, 75, 95, 90};
    int[] mathScores = {90, 70, 80, 95, 85};

    for (int i = 0; i < names.length; i++) {
        studentList.add(new Student(names[i], koreanScores[i], englishScores[i], mathScores[i]));
    }
}
```

for문 사용하여 names 배열 길이 만큼 (5명) add 메서드 사용해서

studentList 객체에 학생정보 넣기

- 메인 메서드 두번째 시행 메서드 : 메뉴 출력

```
private static void showMenu(){
    Scanner scanner = new Scanner(System.in);

    while(true){
        System.out.println("\n================================");
        System.out.println("1. 전체 학생 정보");
        System.out.println("2. 최고 평균 점수 학생 검색");
        System.out.println("3. 최저 평균 점수 학생 검색");
        System.out.println("4. 학생 검색");
        System.out.println("5. 학생 추가");
        System.out.println("6. 종료");
        System.out.println("\n================================");
        System.out.print("원하는 메뉴를 선택하세요: ");

        int menu = scanner.nextInt();
        scanner.nextLine();

        switch (menu){
            case 1: printAllStudents();             break;
            case 2: searchHighestAverageStudent();  break;
            case 3: searchLowestAverageStudent();   break;
            case 4: searchStudent();                break;
            case 5: addStudent();                   break;
            case 6:
                System.out.println("프로그램을 종료합니다.");
                scanner.close();
                System.exit( status: 0);
            default:
                System.out.println("잘못된 메뉴를 선택하셨습니다. 다시 선택해주세요.");
        }
    }
}
```

- 메인 메서드 두번째 시행 메서드 : 메뉴 출력

(왼쪽과 동일)

사용자가 입력한 값을 int menu에 대입

switch 문을 사용하여

사용자가 입력한 값(menu)이 1이면

1번 case 인 printAllStudents(); 메서드를 시행하고

사용자가 입력한 값이 2이면

2번 case 인 searchHighestAverageStudent(); 메서드 시행

....

프로그램 종료시에는 스캐너를 닫아주고

프로그램 정상종료 메서드 시행 (`System.exit(0);`)

사용자가 6 종료를 입력하기 전까지는 while 문으로 인해서

프로그램 무한 반복됨.

- 첫번째 메뉴 : 전체 학생 출력하기

```
private static void printAllStudents(){
    if (numStudents == 0){ System.out.println("등록된 학생이 없습니다."); }
    System.out.println("\n==================================");
    System.out.println("이름\t\tID\t국어\t영어\t수학\t평균");
    System.out.println("\n==================================");

    for (int i = 0; i < numStudents; i++) {
        System.out.println(students[i]);
    }
    System.out.println("\n==================================");

}
```

if문을 사용해서 학생수가 0이면 등록된 학생이 없다는 메시지 출력하기

for문으로 students 배열의 0번 인덱스 정보부터 학생수 인덱스정보까지 출력

- 두,세번재 메뉴 : 최고,최저평균 학생 검색

```
private static void searchHighestAverageStudent(){
    if(numStudents ==0 ){ System.out.println("등록된 학생이 없습니다."); }
    double maxAvg=students[0].getAverage();
    int maxIndex = 0;

    for (int i = 0; i < numStudents; i++) {
        if (students[i].getAverage() > maxAvg) {
            maxAvg = students[i].getAverage();
            maxIndex = i;
        }
    }
    System.out.println("최고 평균 점수 학생: "+students[maxIndex].getName());
}
```

maxAvg 변수를 첫번째 인덱스의 평균점수로 선언 후

for문 사용해서 두번째 인덱스부터 끝인덱스까지 비교해서

큰값을 maxAvg 값으로 대입하고 해당 인덱스를 maxIndex 에 대입.

- 첫번째 메뉴 : 전체 학생 출력하기

```
private static void printAllStudents(){
    if(studentList.isEmpty()){ System.out.println("등록된 학생이 없습니다."); }
    System.out.println("\n==================================");
    System.out.println("이름\t\tID\t국어\t영어\t수학\t평균");
    System.out.println("\n==================================");

    for(Object student: studentList){
        System.out.println((Student)student);
    }
    System.out.println("\n==================================");
}
```

if문을 사용해서 studentList 가 isEmpty(비어있다면) 학생 없음 메시지 출력

for문으로 studentList 의 요소를 Object 클래스 객체로 받아서 출력하기

(String 자료형, int 자료형, double 자료형이 혼재되어 있으므로 object 로 받음)

- 두,세번재 메뉴 : 최고,최저평균 학생 검색

```
private static void searchHighestAverageStudent(){
    if(studentList.isEmpty()){ System.out.println("등록된 학생이 없습니다."); }
    double maxAvg=((Student)studentList.get(0)).getAverage();
    int maxIndex = 0;

    for (int i = 0; i < studentList.size(); i++) {
        if (((Student)studentList.get(i)).getAverage() > maxAvg) {
            maxAvg = ((Student)studentList.get(i)).getAverage();
            maxIndex = i;
        }
    }
    System.out.println("최고 평균 점수 학생: "
            +((Student)studentList.get(maxIndex)).getName());
}
```

왼쪽과 동일하나 배열->List 형태로 바꾸어서 코드 작성

- numStudents -> studentList.size()
- students[i] -> studentList.get(i)

- 네번째 메뉴 : 학생 검색

```
private static void searchStudent(){
    Scanner scanner = new Scanner(System.in);
    System.out.print("검색할 학생 이름을 입력하세요: ");
    String name = scanner.nextLine();
    boolean found = false;

    for (int i = 0; i < numStudents; i++) {
        if (students[i].getName().equals(name)) {
            System.out.println(students[i]);
            found = true;
        }
    }
    if (!found){
        System.out.println("해당 학생을 찾을 수 없습니다.");
    }
}
```

for 문을 이용해서 인덱스 0부터 끝인덱스(numStudents) 까지
사용자가 입력한 이름(name)과 배열의 이름값(students[i].getName())을 비교해서
같으면 (equals) 해당 학생 정보를 출력하고 found 변수값을 true로 바꿈.

배열에 입력한 이름이 없으면 for문에서 끝 인덱스까지 모두 비교 후
if 조건(equals)에 해당하지 않으므로 if문의 실행문을 실행하지 않고
for문을 빠져나와서 if(!found) 의 "학생을 찾을 수 없습니다." 을 출력함.

- 네번째 메뉴 : 학생 검색

```
private static void searchStudent(){
    Scanner scanner = new Scanner(System.in);
    System.out.print("검색할 학생 이름을 입력하세요: ");
    String name = scanner.nextLine();
    boolean found = false;

    for (Object student:studentList) {
        if (((Student)student).getName().equals(name)) {
            System.out.println((Student)student);
            found = true;
        }
    }
    if (!found){
        System.out.println("해당 학생을 찾을 수 없습니다.");
    }
}
```

왼쪽과 동일하나 배열->List 형태로 바꾸어서 코드 작성

- numStudents -> studentList.size()
- students[i] -> studentList.get(i)

- 다섯번째 메뉴 : 학생 추가

```
private static void addStudent(){
    Scanner scanner = new Scanner(System.in);
    System.out.print("추가할 학생 이름을 입력하세요: ");
    String name = scanner.nextLine();
    System.out.print("국어 점수를 입력하세요: ");
    int koreanScore = scanner.nextInt();
    System.out.print("영어 점수를 입력하세요: ");
    int englishScore = scanner.nextInt();
    System.out.print("수학 점수를 입력하세요: ");
    int mathScore = scanner.nextInt();
    scanner.nextLine();

    Student newStudent = new Student(name,
                koreanScore, englishScore, mathScore);
    students[numStudents] = newStudent;
    numStudents++;

    System.out.println("학생이 추가되었습니다.");
}
```

사용자가 입력한 새 학생의 정보로

Student 클래스에 newStudent 객체를 생성하고

students 배열에서

5명 임의 데이터 생성시에 마지막에 학생수보다 +1 되었던

numStudents 인덱스자리에 newStudent 객체를 넣고

다음에 또 학생이 추가될 수 있도록 numStudents 값을 +1

- 다섯번째 메뉴 : 학생 추가

```
private static void addStudent(){
    Scanner scanner = new Scanner(System.in);
    System.out.print("추가할 학생 이름을 입력하세요: ");
    String name = scanner.nextLine();
    System.out.print("국어 점수를 입력하세요: ");
    int koreanScore = scanner.nextInt();
    System.out.print("영어 점수를 입력하세요: ");
    int englishScore = scanner.nextInt();
    System.out.print("수학 점수를 입력하세요: ");
    int mathScore = scanner.nextInt();
    scanner.nextLine();

    Student newStudent = new Student(name,
                koreanScore, englishScore, mathScore);
    studentList.add(newStudent);

    System.out.println("학생이 추가되었습니다.");
}
```

사용자가 입력한 새 학생 정보로

Student 클래스에 newStudent 객체를 생성하고

List 의 add 메서드를 이용해서 학생 정보 추가

(배열과 달리 학생수 인덱스 필요없음)

● P368 LinkedList

✓ ArrayList 와 마찬가지로 List 인터페이스를 구현한 클래스이며 사용방법도 거의 같음.

✓ 차이점은 ArrayList 는 배열형태로 인덱스를 기반으로 데이터가 연속적으로 존재,
LinkedList 는 데이터가 서로 주소값을 통해 참조함(원형 구조: 마지막 데이터가 첫번째 데이터 참조).

✓ ArrayList 는 데이터의 개수가 변하지 않는 경우, LinkedList는 데이터의 추가/삭제가 빈번한 경우에 각각 성능이 좋다

✓ 보통 데이터베이스의 데이터를 처리하는 경우에 데이터 추가/삭제보다 데이터 출력/연산하는 경우가 더 많아서 ArrayList 를 더 자주 사용함.

● P371 Set 인터페이스

✓ List 처럼 Collection 인터페이스의 하위 인터페이스이고 데이터를 다루지만, List와 달리 **중복을 허용하지 않고 저장 순서가 유지되지 않음**.

✓ 예를 들면 {"홍길동", "김철수", "김철수", 1, 2, "1", "2", "3"} 을 set 으로 출력하면 [홍길동, 김철수, 1, 2, 1, 2, 3] 으로 출력됨

✓ (1,2 는 int, "1","2" 는 String 이라서 다른 데이터 이므로 각각 출력되었고,

✓ "김철수", "김철수"는 자료형도 String으로 동일하고 값도 동일하므로 하나만 출력됨) : 중복 비허용

```java
import java.util.HashSet;
import java.util.Set;
public class HashSetEx {
    public static void main(String[] args) {
        Object[] arr = {"홍길동", "이순신", "홍길동", "이순신",1,2,"1","2"};
        Set set = new HashSet();
        for (int i = 0; i < arr.length; i++) {
            set.add(arr[i]);
        }
        System.out.println(set);
    }
}
```

-> Object 타입으로 배열 생성
(String, int 등 다양한 자료형 받음)

-> HashSet 의 객체 "set" 생성 후
"set" 에 arr의 값 하나씩 추가하기 (add메서드)

-> 출력해보면 중복되는 값은 1번만 출력되는 것을 확인할 수 있음

```
[1, 1, 2, 2, 홍길동, 이순신]
```

-> 출력되는 순서도 입력순서와 다름.

- HashSet 선생님 예제 _ HK편의점

✓ 중복되는 데이터값을 빼고 데이터 처리하기 : Set 인터페이스 활용

```
public class HashSetEx3 {
    public static void main(String[] args) {
        String[] websiteData =  {
                "초코파이", "다이제", "칸쵸", "틴캔커피", "진라면", "스크류바", "삼각김밥", "포카칩", "칠성사이다", "파워에이드",
                "콜라", "솔의눈", "핫식스", "허니버터칩", "불닭볶음면", "새우깡", "쌈장", "홈런볼", "미닛메이드", "크랜베리쥬스",
                "호상정"  "종가집김치"  "초코빙스"  "에이스초밥"  "참이슬"  "지즈소주"  "블루베리요거트"  "양파링"  "오레오크키"  "몽쉘"
```

-> 다양한 제품명을 websiteData 배열에 넣기

```
HashSet uniqueDateTypes = new HashSet();

for(String data : websiteData){
    uniqueDateTypes.add(data);
}

LocalDateTime now = LocalDateTime.now();

//DateTimeFormatter 생성
DateTimeFormatter formatter = DateTimeFormatter.ofPattern( pattern: "yyyy년 M월 d일 EEEE a h시간대", Locale.KOREA);

//포맷 적용하여 문자열로 변환
String salesTime = now.format(formatter);

System.out.println(" * HK편의점 판매 제품 정보 ("+salesTime+")기준");
System.out.println(" - 총 데이터 개수: "+websiteData.length);
System.out.println(" - 데이터 종류 개수: "+uniqueDateTypes.size());
System.out.println(" - 종류별 데이터 ");
int count = 1;
for(Object product:uniqueDateTypes){
    System.out.print(" "+(String)product);
    count++;
    if(count==15){
        System.out.println("");
        count=1;
    }
}
```

-> uniqueDateTypes 라는 HashSet 객체 생성 후

websiteData 배열의 요소를 uniqueDateTypes 객체에 넣기 : HashSet 클래스이므로 중복데이터는 1개만 저장됨

-> 총 데이터의 개수는 전체 상품명 개수인 websiteData배열의 길이(length)

데이터 종류 개수는 중복 상품명은 1개로 셀 수 있게 uniqueDateTypes 의 size()

-> 중복된 상품명은 1개로 출력하기

uniqueDateTypes 요소를 모두 출력하는데 요소 하나를 출력할때마다 count 변수가 1씩 더해짐.(count++)

count 변수가 15가 되면 줄바꿈 (println(“”);) -> 14번째 상품명 출력 후 줄바꿈하고 그 다음 상품명 출력