객체지향프로그래밍 13 주차

제출일	2023 년 12 월 5 일	학과	컴퓨터공학과
과목	객체지향프로그래밍 01	학번	2022112701
담당교수	한인 교수님	이름	안소희

□ 실습문제 1

□ 문제분석

학생 정보를 관리하는 프로그램을 작성해야한다.

MyLinkedList<E>클래스의 add, get 메서드를 사용해야한다. 학생 정보에는 학번, 이름, 학과, 학점 평균이 있다. 기능에는 전체 학생의 학점 평균 조회, 학생 정보 등록, 학생 정보 삭제가 있다. 입출력으로 메뉴와 학생 정보를 입력하고 결과를 출력한다.

□ 프로그램 설계 및 알고리즘

먼저 MyLinkedList class 에서 노드 클래스와 사이즈, 노드를 가리키는 주소를 매개변수로 삼고 노드에는 데이터와 다음 노드의 주소를 매개변수로 갖는다. 데이터를 추가할 때 data 를 매개변수로 하는 생성자를 호출하여 newNode 를 생성하고 가장 처음일 때와 아닐 때 경우를 나누어 작성한다. 먼저 head 가 없다면 head 가 newNode 를 가리키도록 한다. 만약 존재한다면 마지막 노드까지 순회하여 그 다음에 newNode 를 추가한다. get 함수는 인덱스 위치의 요소를 반환하는 함수로 인덱스가 범위를 벗어나면 throw 로 처리한다. 그게 아니라면 해당 인덱스까지 이동하여 데이터를 반환한다. 해당 인덱스의 요소를 삭제할 때 인덱스가 0 이라면 head 가 가리키는 노드를 다음 노드로 옮기고 아니라면 해당 인덱스 전까지 이동하여 next 를 next.next 로 옮긴다. Student 객체에선 학생의 정보를 다룬다. 이름 학번 학과 학점평균 정보가 있으며 생성자는 기본 생성자와 학생 정보를 매개변수로 받는 생성자가 있다. toString 으로 멤버 변수 정보를 나타내는 String 을 반환한다. 그리고 메인에서 각각 기능에 맞는 메소드를 호출한다

□ 소스코드 및 주석

(MyLinkedList.java)

```
package week13_1;


public class MyLinkedList<T>{
	private Node<T> head;
	private int size;

	private static class Node<T>{
		T data;
		Node<T> next;

		Node(T data){//생성자
			this.data = data;
			this.next = null;
		}
		public String toString() {
			return "Node data: " + data.toString();
		}
	}
	public MyLinkedList() {//매개변수 없는 기본 생성자
		head = null;
		size = 0;
	}
```

```
        public void add(T data) {

                Node<T> newNode = new Node<T>(data);//data를 매개변수로 받는 생성자 호출
                if(head == null) {//head가 없다면
                        head = newNode;
                }else {
                        Node<T> current = head;//head 존재하면 헤드를 가리키는 current 선언
                        while(current.next != null) {//끝까지 순회
                                current = current.next;
                        }
                        current.next = newNode;//그 다음에 연결되도록 함
                }
                size++;//크기 증가

        }
        public T get(int index) {//리턴 타입이 T인 get 함수
         if (index < 0 || index >= size) {
             throw new IndexOutOfBoundsException("Index: " + index + ", Size: " +
size);
         }
         Node<T> current = head;
         for (int i = 0; i < index; i++) {
             current = current.next;
         }
         return current.data;
    }
        public int getSize() {
                return size;
        }
        public void remove(int index) {//해당 인덱스 요소 삭제
                // TODO Auto-generated method stub
                if(index == 0) {//첫 번째 요소라면 헤드가 그 다음을 가리키도록 함
                        head = head.next;
                }
                else {
                        Node<T> current = head;
                        for(int i = 0 ; i < index ; i++) {//해당 인덱스까지 이동
                                current = current.next;
                        }
                        current.next = current.next.next;
                }
                size--;

        }
}
```

(Student.java)

```
package week13_1;

class Student {
        int studentID;
    String name;//이름
    String department;//학과
    double averageGrade;//학점 평균
```

```
    public Student(int studentID, String name, String department, double
averageGrade) {//위 멤버 변수를 초기화 하는 생성자
        this.studentID = studentID;
        this.name = name;
        this.department = department;
        this.averageGrade = averageGrade;
    }
    public Student() {//기본 생성자
        studentID  = 0;
        name = null;
        department = null;
        averageGrade = 0;
    }
    @Override
    public String toString() {
        return "Student{" +
                "studentID='" + studentID + '\'' +
                ", name='" + name + '\'' +
                ", department='" + department + '\'' +
                ", averageGrade=" + averageGrade +
                '}';
    }
}
```

(StudentEx.java)

```
package week13_1;

import java.util.Scanner;

public class StudentEx {
        private static final Scanner oInDev = new Scanner(System.in);

    public static void main(String[] args) {
        MyLinkedList<Student> studentList = new MyLinkedList<>();

        while (true) {
            System.out.println("1. 전체 학생의 학점 평균 조회");
            System.out.println("2. 학생 정보 등록");
            System.out.println("3. 학생 정보 삭제");
            System.out.println("4. 학생 학점 평균 검색");
            System.out.println("5. 종료");
            System.out.print("선택: ");

            int choice = oInDev.nextInt();
            oInDev.nextLine(); // 개행 문자 처리



            switch (choice) {
                case 1:
                      displayAverageGrade(studentList);//전체 학생의 학점 평균 조회
                    break;
                case 2:
                    registerStudent(studentList);//학생 정보 등록
                    break;
                case 3:
                    deleteStudent(studentList);//학생 정보 삭제
                    break;
```

```
                case 4:
                      SearchAverageGrade(studentList);//학생의 학점 검색
                    break;
                case 5:
                    System.out.println("프로그램을 종료합니다.");
                    oInDev.close();
                    System.exit(0);
                default:
                    System.out.println("올바른 선택이 아닙니다. 다시 선택해주세요.");
            }
        }


    }
// 전체 학생의 학점 평균 조회 메서드
    public static void displayAverageGrade(MyLinkedList<Student> studentList) {

        double dTotal = 0.0;
        int iSize =  studentList.getSize();//전체 학생 수
        if(iSize > 0) {
               for (int i = 0; i < iSize; i++) {
                      dTotal += studentList.get(i).averageGrade;//학점 총합 구하기
               }
               double dAverage = dTotal / iSize;//나누어 평균 구함
               System.out.println("전체 학생의 학점 평균: " + dAverage);
        }
        else {
              System.out.println("등록된 학생이 없습니다");
        }
    }
    //학점 평균 검색 메서드
    public static void SearchAverageGrade(MyLinkedList<Student> studentList){

        int i;
        int ID;
        int iSize = studentList.getSize();


        if(iSize > 0 ) {
               System.out.print("학번 입력: ");
               ID = oInDev.nextInt();
               for (i = 0; i < iSize ; i++) {//순회하며 있는지 확인
                    Student student = studentList.get(i);
                    if (student.studentID == ID) {
                       System.out.println(student.averageGrade);
                       break;
                    }
               }
               if(i == iSize) {// 입력한 학번에 해당하는 학생이 없는 경우
                      System.out.println("해당하는 학생이 없습니다.");

               }
        }
        else {//총 학생수가 0 인 경우
               System.out.println("등록된 학생이 없습니다.");
```

```
        }


    }
    //학생 등록 메서드
    public static void registerStudent(MyLinkedList<Student> studentList) {

        System.out.println("학번 입력: ");
                int iID = oInDev.nextInt();
                System.out.println("이름 입력: ");
                String sName = oInDev.next();
                System.out.println("학과 입력: ");
                String sDepartment = oInDev.next();
                System.out.println("학점평균 입력: ");
                double dAverageGrade = oInDev.nextDouble();



                studentList.add(new Student(iID, sName, sDepartment,
dAverageGrade));//위의 정보를 매개변수로 하는 객체 생성 후 리스트에 저장

    }
    //학생 정보 삭제
    public static void deleteStudent(MyLinkedList<Student> studentList) {
        int i ;
        int ID;
        int iSize = studentList.getSize();
        if (iSize > 0) {
                System.out.print("학번 입력: ");
                ID = oInDev.nextInt();
                for (i = 0; i < iSize; i++) {
                 Student student = studentList.get(i);
                 if (student.studentID == ID) {
                     break;
                 }
             }
                if(i == iSize) {//해당하는 학생 없을 시
                        System.out.println("해당하는 학생이 없습니다.");
                }
                else {//학생 존재
                     Student removedStudent = studentList.get(i);//해당 인덱스의 객체 반환
                     System.out.println("삭제된 학생 정보: " + removedStudent);//문자열과
연결하며 자동으로 toSring()호출
                     studentList.remove(i);
                }
         }
        else {//학생 수가 0 명일 때
                System.out.println("등록된 학생이 없습니다.");

        }
    }

}
```

□ 결과 및 결과 분석

1. 결과 화면을 캡처하여 첨부, 해당 결과가 도출된 이유와 타당성 분석
2. 핵심 내용을 중심으로 결과 분석, 필요시 다음 표를 활용하여 명확히 작성

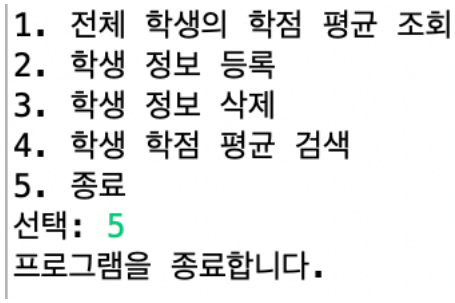
결과 화면

```
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 2
학번 입력:
2022
이름 입력:
학생1
학과 입력:
컴퓨터공학과
학점평균 입력:
4.1
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 2
학번 입력:
2020
이름 입력:
학생2
학과 입력:
```

전자전기공학부
학점평균 입력:
4.04
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 2
학번 입력:
2023
이름 입력:
학생3
학과 입력:
미디어커뮤니케이션학과
학점평균 입력:
3.98
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 1
전체 학생의 학점 평균: 4.04
1. 전체 학생의 학점 평균 조회

```
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 4
학번 입력: 2022
4.1
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 3
학번 입력: 2022
삭제된 학생 정보: Student{studentID='2022', name='학생1', department='컴퓨터공학과', averageGrade=4.1}
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 3
학번 입력: 2020
삭제된 학생 정보: Student{studentID='2020', name='학생2', department='전자전기공학부', averageGrade=4.04}
```

```
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 3
학번 입력: 2023
삭제된 학생 정보: Student{studentID='2023', name='학생3', department='미디어커뮤니케이션학과', averageGrade=3.98}
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 1
등록된 학생이 없습니다
1. 전체 학생의 학점 평균 조회
2. 학생 정보 등록
3. 학생 정보 삭제
4. 학생 학점 평균 검색
5. 종료
선택: 3
등록된 학생이 없습니다.
```



결과 분석

학생 정보 등록 후 전체 학생의 학점 평균이 알맞게 나왔고 특정 학생 학점 평균 검색도 되었다. 학생 정보를 삭제할 때 학번을 기준으로 삭제하였고 등록된 학생 정보가 없을 때 알맞은 메세지가 출력되었다.

□ 소감

Node 를 이용하여 LinkedList 를 구성하는 게 생각보다 더 어려웠다. 자료구조 때 배운 걸 활용하는 거 같았다. 그리고 학점 평균 조회가 학번 검색해서 조회하는 건 줄 알았는데 아니어서 따로 기능을 추가했다.

□ 실습문제 2

□ 문제분석

Shape 를 관리하는 ShapeManager 클래스를 만들어야한다.

먼저 멤버 변수 size 와 이를 출력하는 toString()함수를 작성한다. ArrayList 를 상속받은 ShapeManager 클래스는 새로운 Shape 입력 받고 내림차순으로 보관하는 add 메서드와 사이즈가 가장 큰 Shape 를 반환하는 get(), 해당 인덱스 반환하는 get(i)이 있다.

□ 프로그램 설계 및 알고리즘

먼저 내림차순으로 정렬하는 메서드를 만들고 이를 add 에서 호출하여 내림차순으로 보관한다. 또한 add 에서 현재 모든 Shape 를 출력한다. get()은 내림차순 보관이 되어있으므로 0 번째 Shape 를 반환하며 get(i)는 유효한 인덱스인지 확인 후 ArrayList 의 get 메서드를 사용하여 반환한다. Shape 클래스는 인덱스를 가져야 하므로 인덱스를 나타내는 i 객체마다가 아닌 클래스마다 관리해야 한다. 그래서 생성할 때마다 1 증가하여 인덱스에 저장한다. 그리고 Shape 를 출력하는 draw 메서드, size 반환 메서드, 멤버 변수를 스트링으로 반환하는 toString 메서드가 있다. 메인에서 Shape 개수를 입력받아 그 만큼 사이즈를 입력받아 내림차순으로 저장한다. 그 후 while 에서 조회를 하는데 q 입력시 종료한다. -1 입력시 가장 큰 값, 다른 수 입력 시 해당 인덱스의 Shape 정보가 출력되도록 했다.

□ 소스코드 및 주석

(Shape.java)

```
package week13_2;

public class Shape {//Shape 클래스
        private int size;//사이즈를 멤버 변수로 가짐
        static private int i = 0;//인덱스를 저장하기 위해 선언
        int index;
        public Shape(int size) {
         this.size = size;
         i++;//생성할 때 마다 증가시켜 인덱스 카운트를 해준다
         index = i;
    }

        public void draw() {
                System.out.println("Shape");
        }
        public int getSize() {//size 반환
         return size;
        }
        public String toString() {//멤버 변수 정보 스트링으로 반환
                return "Shape"+index+": "+size;
        }
}
```

(ShapeEx.java)

```java
package week13_2;
import java.util.Scanner;

public class ShapeEx {
	public static void main(String[] args) {
		int iCnt;
		int iSize;
		String sInput;
		int iInput;

		Scanner oInDev = new Scanner(System.in);
	 ShapeSequence shapeManager = new ShapeSequence();

	 System.out.print("Shape 개수 입력: ");
	 iCnt = oInDev.nextInt();

	 for(int i = 0 ; i < iCnt ; i++) {//입력받은 Shape 개수만큼 사이즈 입력받음
		System.out.print("Shape"+(i+1)+"의"+" size 입력: ");
		iSize = oInDev.nextInt();
		System.out.println("===========현재 상태===========");
		shapeManager.add(new Shape(iSize));
		System.out.println("============================");


	 }


	 while(true) {
		System.out.println("몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)");
		sInput = oInDev.next();
		if(sInput.equals("q")) {//q 입력시 종료
			oInDev.close();
			break;
		}

		iInput = Integer.parseInt(sInput);
		if(iInput == -1) {//-1 입력시 가장 큰 값 반환
			System.out.println(shapeManager.get());
		}
		else {//인덱스 입력시 해당 Shape 정보 반환
			System.out.println(shapeManager.get(iInput-1));
		}


	 }


    }
}
```

(ShapeSequence.java)

```java
package week13_2;
import java.util.ArrayList;


public class ShapeSequence extends ArrayList<Shape> {

	private void sortShapes() {
	 // 내림차순 정렬
```

```
                for (int i = 0; i < this.size(); i++) {
            for (int j = i + 1; j < this.size(); j++) {
                if (this.get(i).getSize() < this.get(j).getSize()) {
                    // swap
                    Shape temp = this.get(i);
                    this.set(i, this.get(j));
                    this.set(j, temp);
                }
            }
        }
    }
        @Override
        public boolean add(Shape shape) {
            super.add(shape);//ArraList 의 add 메서드 사용
            sortShapes();//내림차순 정렬

         // 추가됐을 때 현재 있는 모든 Shape 출력
            System.out.println("현재 있는 모든 Shape:");
            for (Shape s : this) {
                System.out.println(s.toString());
            }

            return true;
        }

    public Shape get() {
         // 리스트가 비어있을 경우 null 반환
        if(this.isEmpty()) {
                return null;
        }
        else {
                return this.get(0);
        }

    }

    public Shape get(int i) {
         // 유효한 인덱스인지 확인 후 반환
        if(i >= 0 && i < this.size()) {
                return super.get(i);//ArrayList 의 get 메서드 사용
        }
        else {
                return null;
        }

    }
}
```

□ 결과 및 결과 분석

1. 결과 화면을 캡처하여 첨부, 해당 결과가 도출된 이유와 타당성 분석
2. 핵심 내용을 중심으로 결과 분석, 필요시 다음 표를 활용하여 명확히 작성

결과 화면

```
Shape개수 입력: 3
Shape1의 size 입력: 8
===========현재 상태===========
현재 있는 모든 Shape:
Shape1: 8
============================
Shape2의 size 입력: 12
===========현재 상태===========
현재 있는 모든 Shape:
Shape2: 12
Shape1: 8
============================
Shape3의 size 입력: 10
===========현재 상태===========
현재 있는 모든 Shape:
Shape2: 12
Shape3: 10
Shape1: 8
============================
몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)
-1
Shape2: 12
몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)
1
Shape2: 12
몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)
2
Shape3: 10
몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)
3
Shape1: 8
몇 번째 값을 가져올까요?(가장 큰 값: -1, 종료: q)
q
```

결과 분석

먼저 3 개를 입력받도록 개수 입력 후 사이즈 입력시 내림차순으로 보관된 것을 볼 수 있다. -1 입력 시 가장 큰 값인 12, 그 외의 숫자 입력 시 index 해당하는 Shape 의 정보가 출력되었다.

□ 소감

ArrayList 를 선택한 이유는 검색에 적합하고 삭제를 하지 않기 때문이다. 인덱스를 지녀 검색에 유리하고 배열이기에 삽입 삭제에 부적합하다. LinkedList 는 검색에 용이 하지 않아 밑에 조회하는 기능에서 부적합하기에 ArrayList 를 선택했다. Vector 는 하나의 스레드가 하나의 자원을 이용하기 때문에 제외했다.