|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **[유헬스2\_12주\_실습과제]** | **학번:** | **20175105** | **이름:** | **곽영주** |

* 컬렉션 클래스 : ArrayList

import java.util.\*; //컬렉션 클래스 사용을 위해 반드시 작성

public class ArrayListTest {

public static void main(String args[]) {

ArrayList<String> list = new ArrayList<String>(); //ArrayList 객체 생성

list.add("MILK"); //ArrayList에 원소 추가

list.add("BREAD");

list.add("BUTTER");

list.add(1, "APPLE"); // 인덱스 1에 “APPLE"을 삽입

list.set(2, "GRAPE"); // 인덱스 2의 원소를 “GRAPE"로 대체

list.remove(3); // 인덱스 3의 원소 제거

for (int i = 0; i < list.size(); i++) // size()는 원소 개수 반환

System.out.println(list.get(i)); // get()는 i번째 위치의 원소 반환

//for(String s:list) // list에 저장된 원소 출력 시 foreach 사용

// System.out.println(s)

}

}

|  |
| --- |
| **[실행결과]** |

* LinkedList

**import** java.util.\*;

**public** **class** LinkedListTest{

**public** **static** **void** main(String[] args){

LinkedList<String> list = **new** LinkedList<String>();

list.add("MILK");

list.add("BREAD");

list.add("BUTTER");

list.add("BANANA");

list.add(1, "APPLE"); // 인덱스 1에 “APPLE"을 삽입

list.set(2, "GRAPE"); // 인덱스 2의 원소를 “GRAPE"로 대체

list.remove(1); // 인덱스 1의 원소를 삭제한다.

**for** (**int** i = 0; i < list.size(); i++)

System.***out***.println(list.get(i)); //LinkedList 에 저장된 원소 출력

System.***out***.println("\n >>>>> Iterator 을 이용한 출력 ");

Iterator<String> e=list.iterator();

**while**(e.hasNext())

{

String str = e.next();

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

}

}}

|  |
| --- |
| **[실행결과]** |

* Set 인터페이스
* HashSet : 성능 가장 우수, 하지만 원소들의 순서가 일정하지 않은 단점이 있다.
* TreeSet : 값에 따라서 순서 결정, 하지만 HashSet보다는 느리다.
* LinkedHashSet : 해쉬 테이블과 연결 리스트를 결합한 것으로 원소들의 순서는 삽입되었던 순서와 같다**.**

============= 사용 예1 ================

import java.util.\*;

public class SetTest {

public static void main(String args[]) {

**// TreeSet, LinkedHashSet 으로 변경한 후 실행결과 반드시 확인할 것**

HashSet<String> set = new HashSet<>();

set.add("Milk");

set.add("Bread");

set.add("Butter");

set.add("Cheese");

set.add("Ham");

set.add("Ham"); //중복된 원소는 추가되지 않음

System.out.println(set); //원소들의 순서가 일청하지 않음

}

}

|  |
| --- |
| **[실행결과]** |

============= 사용 예2 ================

import java.util.\*;

public class SetTest1 {

public static void main(String[] args) {

Set<String> s1 = new HashSet<String>();

Set<String> s2 = new HashSet<String>();

s1.add("A"); s1.add("B");s1.add("C");

s2.add("A"); s2.add("D");

//s1을 가지고 새로운 집합 union 생성 - 원본 보관

Set<String> union = new HashSet<String>(s1);

System.out.println("부분집합 : " +union.containsAll(s1)); //부분집합

union.addAll(s2); //합집합

Set<String> intersection = new HashSet<String>(s1);

intersection.retainAll(s2); //교집합

System.out.println("합집합 " + union);

System.out.println("교집합 " + intersection);

}

}

|  |
| --- |
| **[실행결과]** |

* Map 인터페이스
* 사전과 같은 자료 구조
* 키(key)에 값(value)이 매핑 된다.
* 중복된 키를 가질 수 없고, 하나의 값에만 매핑
* Map 인터페이스 구현 클래스
* HashMap : 해쉬 테이블에 데이터 저장
* TreeMap : 탐색 트리에 저장
* LinkedHashMap

============= 사용 예1 : 사전================

public class EnglishDic {

public static void main(String[] args) {

Map<String, String> st = new HashMap<String, String>();

Scanner sc = new Scanner(System.in);

st.put("map", "지도");

st.put("java", "자바");

st.put("school", "학교");

do {

System.out.print("영어 단어를 입력하시오:");

String key = sc.next();

if( key.equals("quit") ) break;

System.out.println("단어의 의미는 " + st.get(key));

} while(true);

}

}

|  |
| --- |
| **[실행결과]** |

============= 사용 예2 : 빈도수 ================

**import** java.util.\*;

**public** **class** WordFreq{

**public** **static** **void** main(String[] args){

Map<String, Integer> m=**new** HashMap<String, Integer>();

String[] sample = {"to", "be", "or", "not", "to", "be", "is", "a", "pro"};

//문자열에 포함된 빈도수 계산

**for**(String a : sample){

Integer freq=m.get(a); //key에 해당하는 value 반환

m.put(a, (freq==**null**)?1:freq+1);

}

System.***out***.println(m.size() + " 단어가 있습니다");

System.***out***.println(m.containsKey("to")); //해당 키 존재하면 true

System.***out***.println(m.isEmpty()); //해쉬 맵이 비어 있으면 true

System.***out***.println(" >>>>>> 1: 람다식을 이용한 원소 출력 ");

m.forEach((key,value)->{

System.***out***.println("key = " + key + ", value = " + value);

});

System.***out***.println(" >>>>>> 2: Map.Entry 인터페이스를 이용한 원소 출력 ");

**for** (Map.Entry<String, Integer> s : m.entrySet()) { // 모든 항목을 방문한다

String key = s.getKey();

Integer value = s.getValue();

System.***out***.println("key= " + key + ", value= " + value);

}

System.***out***.println(" >>>>>> 3: toString()을 이용한 원소 출력 ");

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

}}

|  |
| --- |
| **[실행결과]** |

* 중간 점검

1. 다음과 같이 리스트가 생성되었다고 하자. 다음의 각 문장을 실행한 후의 결과를 쓰시오.

String[] s = { "사과", "배", "바나나" };

ArrayList list = **new** ArrayList(Arrays.*asList*(s));

**빨간색 : 각각 문장만 했을 경우**

**파란색 : 각각 문장을 이어서 했을 경우**

(1) list.add("포도"); System.*out*.println(list);

**사과, 배, 바나나, 포도 // 사과, 배, 바나나, 포도**

(2) list.add(2, "자몽"); System.*out*.println(list);

**사과, 배, 자몽, 바나나 // 사과, 배, 자몽, 바나나, 포도**

(3) System.*out*.println(list.get(3));

**오류 // 바나나**

(4) list.remove(1); System.*out*.println(list);

**사과, 바나나 // 사과, 자몽, 바나나, 포도**

(5) System.out.println(list.contains("사과"));

**true // true**

(6) System.out.println(list.indexOf("사과"));

**0 // 0**

1. list가 ArrayList<Double>의 객체를 참조하고 있다고 하자. list의 모든 원소를 출력하는 문장을 다음과 같이 작성하라.

(1) 인덱스 변수를 사용하는 보통의 for 루프

**for(int =0; i<list.size(); i++) {**

**System.out.println(list.get(i));**

**}**

(2) for-each 구문을 사용

**for(Double temp : list) {**

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

**}**

(3) Iterator를 사용

**Iterator<Double> e = list.iterator();**

**while(e.hasNext()) {**

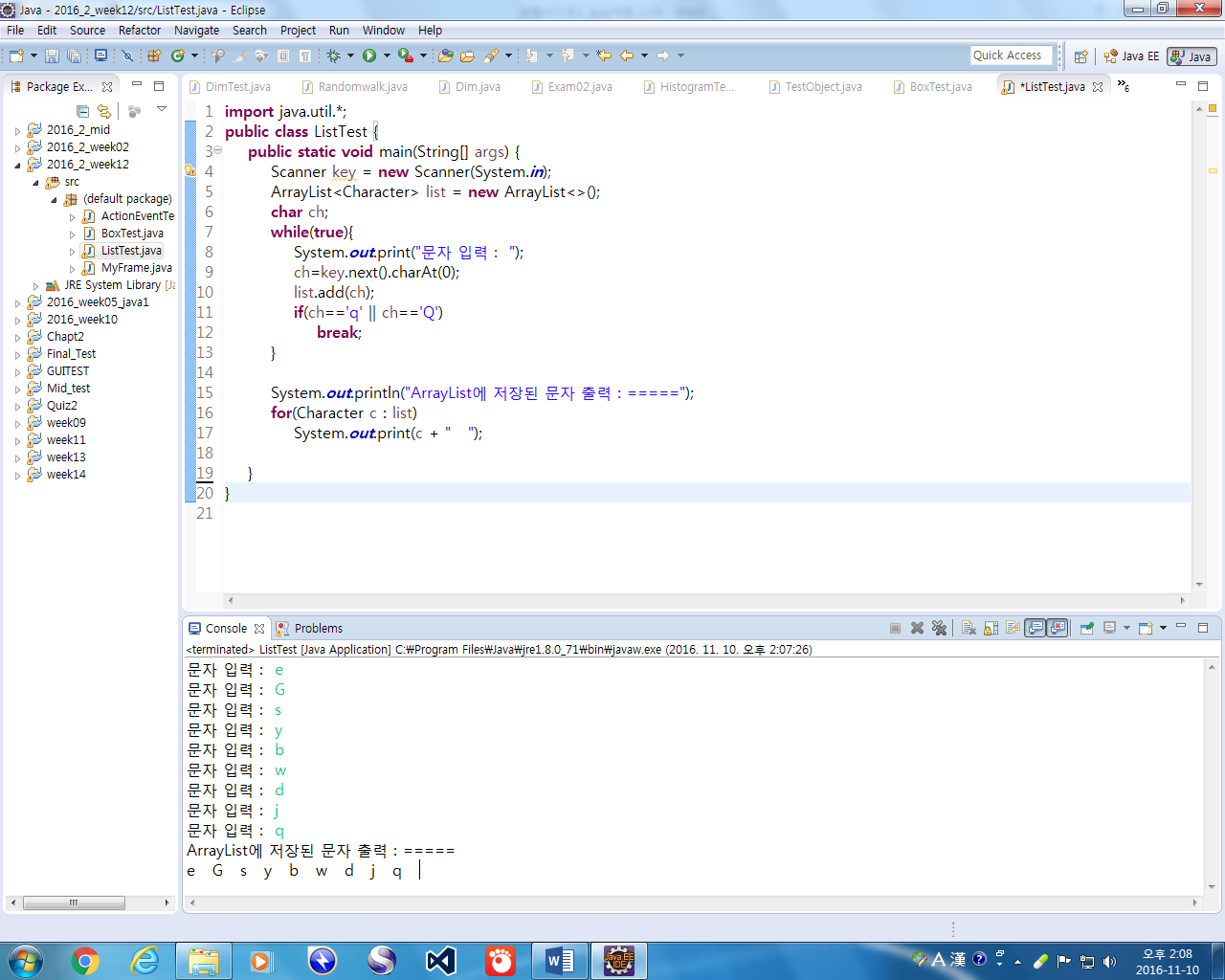
**Double dou = e.nextDouble();**

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

**}**

* 프로그램 과제

1. 키보드로 입력된 문자를 ArrayList에 저장한 후 출력하는 프로그램을 작성하시오. ‘q’ 또는 ‘Q’가 입력되면 반복을 종료한다. 단, ArrayList에 저장된 문자 출력 시 get()메소드를 사용할 것.



|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** asd {  **public** **static** **void** main(String[] args) {  Scanner sc = **new** Scanner(System.***in***);  ArrayList<Character> list = **new** ArrayList<>();  **char** ch;    **while**(**true**) {  System.***out***.print("문자 입력 : ");  ch = sc.next().charAt(0);  **if**(ch == 'Q' || ch == 'q')  **break**;  list.add(ch);  }    System.***out***.println("ArrayList에 저장된 문자 출력 : ===>");  **for**(**char** temp : list)  System.***out***.print(temp + "\t");  }  } |
| **[실행결과]** |

1. 로또 번호를 생성하는 프로그램을 작성하시오. 로또는 1부터 45까지의 숫자 중에서 6개를 선택한다. 로또 번호는 중복되면 안 된다.

힌트 : HashSet 또는 LinkedHashSet 을 사용할 것

String a=”단어”;

if (!s.add(a))

System.out.println("중복된 단어==> " + a);

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** Lotto {  **public** **static** **void** main(String[] args) {  HashSet<Integer> hs = **new** HashSet<>();  Random lotto\_num = **new** Random();    **while**(**true**) {  hs.add(lotto\_num.nextInt(45)+1);  **if**(hs.size() == 6)  **break**;  }  System.***out***.println("=== 로또 번호 ===");  System.***out***.print(hs + "\t");  }  } |
| **[실행결과]** |

1. Book 클래스 이용하여 Book 정보를 ArrayList에 저장하고, 저장된 모든 데이터를 출력하는 프로그램을 작성하시오. 다음과 같은 메뉴를 제공하도록 한다.

1: 저장 2:대출 3:검색 4:리스트 출력 5. 삭제 6. 종료

public class Book {

private String number; //관리번호

private String title; //제목

private String author; //저자

private boolean status; // 대출 상태(대출 중-> true, 대출가능 -> false)

//생성자 – 본인작성

- 형식매개변수로 필드 초기화, 필드 status는 false로 할 것

// toString() 메소드 – 본인작성

- 객체 상태를 문자열로 반환(관리번호, 제목, 저자, 대출상태

//void rentBook() 메소드 – 본인작성

- status 필드값이 false이면 true로 변경한 후 “대출완료” 출력

- ture 이면 “대출 불가(대출 중)” 출력

//모든 필드에 대한 접근자 메소드 – 본인작성

}

import java.util.\*;

public BookTest{

public static void main(String[] args){

//메뉴를 선택

//1번 : ArrayList에 저장 – 필요한 자료는 입력을 받는다

//2번 : 대출하고자 하는 책 제목을 입력 받은 후 rentBook()메소드로 처리

//3번 : 검색하고자 하는 책 제목을 입력 받은 후 객체 상태 출력

//4번 : ArrayList에 저장 된 모든 원소 출력

//5번 : 입력된 관리번호와 일치하는 책을 ArrayList 에서 삭제

//6번 : 반복문 종료

}

}

|  |
| --- |
| **[프로그램 소스]**  **public** **class** Book {  **private** String number;  **private** String title;  **private** String author;  **private** **boolean** status;    **public** Book(String number, String title, String author) {  **this**.number = number;  **this**.title = title;  **this**.author = author;  **this**.status = **false**;  }    **public** String toString() {  **return** "관리번호 : " + number + "\n제목 : " + title + "\n저자 : " + author + "\n대출 상태 : " + status + "\n";  }    **public** **void** rentBook() {  **if**(status == **false**) {  status = **true**;  System.***out***.println("대출 완료");  }  **else** {  System.***out***.println("대출 불가(대출 중)");  }  }    **public** String getNumber() {  **return** number;  }    **public** String getTitle() {  **return** title;  }    **public** String getAuthor() {  **return** author;  }    **public** **boolean** getStatus() {  **return** status;  }  }  **import** java.util.\*;  **public** **class** BookTest {  **public** **static** **void** main(String[] args) {  Scanner sc = **new** Scanner(System.***in***);  ArrayList<Book> list = **new** ArrayList<>();  **int** menu;    **while**(**true**) {  System.***out***.println("1: 저장 2:대출 3:검색 4:리스트 출력 5: 삭제 6: 종료");  System.***out***.print("메뉴를 선택해 주세요 : ");  menu = sc.nextInt();    **if**(menu == 6) {  System.***out***.println("[종료]");  **break**;  }    **switch**(menu) {  **case** 1 :  System.***out***.println("[저장]");  System.***out***.print("관리번호 : ");  String number = sc.next();  System.***out***.print("제목 :");  String title = sc.next();  System.***out***.print("저자 : ");  String author = sc.next();  list.add(**new** Book(number,title,author));  **break**;    **case** 2 :  System.***out***.print("[대출] 책 제목 입력 : ");  String rentTitle = sc.next();  **for**(Book data : list) {  **if**(rentTitle.equals(data.getTitle()))  data.rentBook();  }  **break**;    **case** 3 :  System.***out***.print("[검색] 책 제목 입력 : ");  String searchTitle = sc.next();  **for**(Book data : list) {  **if**(searchTitle.equals(data.getTitle()))  System.***out***.println(data);  }  **break**;    **case** 4 :  System.***out***.println("=== 리스트 출력 ===");  **for**(Book data : list)  System.***out***.println(data);  **break**;    **case** 5 :  System.***out***.print("[삭제] 관리번호 입력 : ");  String delNumber = sc.next();  **for**(**int** i=0; i<list.size(); i++) {  **if**(delNumber.equals(list.get(i).getNumber()))  list.remove(i);  }  System.***out***.println("삭제 완료");  **break**;  }  }  }  } |
| **[실행결과]** |

1. 표준 입력장치로 입력되는 단어의 입력 횟수를 출력하는 프로그램을 HashMap을 이용하여 작성하시오

- HashMap을 이용하여 <단어, 입력횟수>의 key와 value을 저장

- 입력 받은 모든 단어 출력

- 단어와 입력횟수 출력 – 람다식 사용

|  |
| --- |
| [출력 예시]  입력된 단어 : java, c, java, cpp, cpp, cobol, java  4개 단어 검출  cobol=1, cpp=2, c=1, java=3 |

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** HasMapTest {  **public** **static** **void** main(String[] args) {  Scanner scan = **new** Scanner(System.***in***);  HashMap<String,Integer> m = **new** HashMap<>();  String[] dan = **new** String[7];    **for**(**int** i=0; i<dan.length; i++) {  System.***out***.print("단어 입력 :");  dan[i] = scan.next();  }    System.***out***.print("입력된 단어 : ");  **for**(String data : dan)  System.***out***.print(data + " ");    **for**(String a : dan) {  Integer freq = m.get(a);  m.put(a,(freq == **null**)? 1: freq+1);  }    System.***out***.println("\n" + m.size() + "개 단어 검출");    m.forEach((key,value) -> {  System.***out***.print(key + "=" + value + " ");  });  }  } |
| **[실행결과]** |

1. 간단한 주소록을 처리하는 프로그램을 HashMap을 이용하여 작성하시오

- <id, Person>의 key와 value 저장, id는 String

- 임의의 데이터를 저장

- id를 입력하면 해당 Person 데이터 출력.

- HashMap에 저장된 모든 데이터 출력. 단, Map.Entry 인터페이스를 이용한 원소 출력

**class** Person{

**private String** name; //이름

**private String** num; //전화번호

**public** Person(String name, **String** num){

**this**.name=name;

**this**.num=num;

}

**public** String toString(){

**return** " [이름 : " + name + ", 전화번호 : " + num +"]" ;

}

}

|  |
| --- |
| **[프로그램 소스]**  **public** **class** Person {  **private** String name;  **private** String num;    **public** Person(String name, String num) {  **this**.name = name;  **this**.num = num;  }    **public** String toString() {  **return** "[이름 : " + name + ", 전화번호 : " + num + "]";  }  }  **import** java.util.\*;  **public** **class** PersonTest {  **public** **static** **void** main(String[] args) {  Scanner scan = **new** Scanner(System.***in***);  HashMap<String,Person> m = **new** HashMap<>();    m.put("Kwak",**new** Person("곽영주","010-1234-5678"));  m.put("Kim",**new** Person("김재석","010-9876-5432"));  m.put("Jo",**new** Person("조성윤","010-7418-9632"));    System.***out***.print("id 입력하면 해당 주소록 출력 : ");  String id = scan.next();  System.***out***.println(m.get(id));    System.***out***.println("=== 모든 데이터 출력 ===");  **for**(Map.Entry<String,Person> data : m.entrySet()) {  String key = data.getKey();  Person value = data.getValue();  System.***out***.println("id : " + key + " " + value);  }  }  } |
| **[실행결과]** |