|  |  |  |
| --- | --- | --- |
| **2019\_2\_java2\_11** | **학번 : 20165164** | **이름 : 지현한** |

* **내용 점검**

1. ArrayList

|  |
| --- |
| **import** java.util.\*;  **class** Person{  **private** String name; //이름  **private** **int** height; //키    **public** Person(String name, **int** height) {  **this**.name = name;  **this**.height = height;  }  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }  **public** **int** getHeight() {  **return** height;  }  **public** **void** setHeight(**int** height) {  **this**.height = height;  }  @Override  **public** String toString() {  **return** "Person [name=" + name + ", height=" + height + "]";  }  }  **public** **class** ArrayTest {  **public** **static** **void** write(ArrayList<Person> alist) {  **for**(Person person : alist)  System.***out***.println(person);  }  **public** **static** **void** main(String[] args) {  ArrayList<Person> alist=**new** ArrayList<>();  alist.add(**new** Person("기태", 170));  alist.add(**new** Person("동규", 173));  *write*(alist);  alist.add(1, **new** Person("재승", 171));  *write*(alist);  alist.set(1, **new** Person("환수", 176));  *write*(alist);  }  } |
| **[실행 결과]** |

1. Set

|  |
| --- |
| ============= 사용 예1 ================  import java.util.\*;  public class SetTest {  public static void main(String args[]) {  **// TreeSet, LinkedHashSet 으로 변경한 후 실행결과 반드시 확인할 것**  Set<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]**  HashSet    TreeSet    LinkedHashSet |
| =========== 사용 예2 ===============  import java.util.\*;  public class HashSetTest {  public static void main(String args[]) {  Set<String> s = new HashSet<String>();  String[] sample = { "단어", "중복", "구절", "중복" };  for (String a : sample)  //중복되는 단어는 저장되지 않으며 false 반환  if (!s.add(a))  System.out.println("중복된 단어==> " + a);  System.out.println("HashSet에 저장된 단어 개수 : " + s.size());  System.out.println("중복되지 않은 단어: " + s);  }  } |
| **[실행결과]** |

* **실습 과제**

1. HashSet 을 사용 하여 2자리수 난수 10개를 중복없이 생성하는 프로그램을 작성하시오.



|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** Q1main {  **public** **static** **void** main(String[] args) {  Set<Integer> set = **new** HashSet<>();  Random rnd = **new** Random();  **while** (set.size() < 10) {  set.add(rnd.nextInt(89) + 10);  }  System.***out***.println(set);  }  } |
| **[실행 결과]** |

1. 다음과 같이 리스트가 생성되었을 때 제시된 문제를 해결하는 프로그램을 작성하시오.

String[] s = { "사과", "배", "바나나" };

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

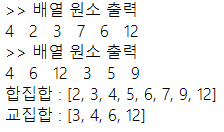
|  |  |
| --- | --- |
|  | 1. “포도”를 리스트 마지막에 삽입 2. “자몽”을 인덱스 2에 삽입 3. 리스트에 저장된 데이터 출력 4. 인덱스 3에 저장된 데이터 출력 5. 인덱스 2에 저장된 데이터를 “메론” 변환 6. “배” 를 리스트 마지막에 삽입 7. “배” 가 저장된 위치를 출력 8. 리스트에서 “바나나” 데이터 삭제 – remove(Object obj), remove(int index) 9. 리스트에 저장된 데이터 출력 |

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** Q2main {  **public** **static** **void** main(String[] args) {  String[] s = { "사과", "배", "바나나" };  ArrayList list = **new** ArrayList(Arrays.*asList*(s));  list.add("포도");  list.add(2, "자몽");  System.***out***.println("------- 리스트 출력 ------------");  **for**(Object obj:list)  System.***out***.println(obj);  System.***out***.println("----------------------------");  System.***out***.println("인덱스 3: " + list.get(3));  list.set(2, "메론");  list.add("배");  System.***out***.println("배가 저장된 위치: " + list.indexOf("배"));  list.remove("바나나");  System.***out***.println("------- 리스트 출력 ------------");  **for**(Object obj:list)  System.***out***.println(obj);  System.***out***.println("----------------------------");  }  } |
| **[실행결과]** |

1. Set을 사용하여 다음과 같은 집합에 대하여 합집합과 교집합을 구하여 출력하는 프로그램을 작성하시오.

**int**[] d1= {4,2,3,7,6,12};

**int**[] d2= {4,6,12,3,5,9};



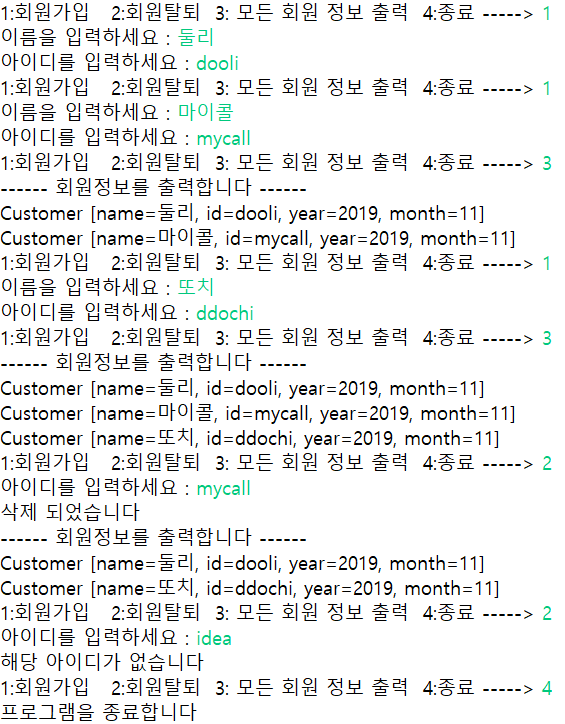
|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** Q3main {  **public** **static** **void** main(String[] args) {  **int**[] d1 = { 4, 2, 3, 7, 6, 12 };  **int**[] d2 = { 4, 6, 12, 3, 5, 9 };  Set<Integer> set1 = **new** HashSet<>();  Set<Integer> set2 = **new** HashSet<>();  **for** (**int** temp : d1)  set1.add(temp);  **for** (**int** temp : d2)  set2.add(temp);    *print*(set1);  *print*(set2);  Set<Integer> union = **new** HashSet<>(set1);  //union.addAll(set2);  **for**(**int** temp:set2)  union.add(temp);    Set<Integer> Intersection = **new** HashSet<>();  Set<Integer> Intersection2 = **new** HashSet<>(set1);  //Intersection2.retainAll(set2);  **for**(**int** temp:set2) {  **if**(!Intersection2.add(temp)) {  Intersection.add(temp);  }  }    System.***out***.println("합집합: " + union);  System.***out***.println("교집합: " + Intersection);  }  **public** **static** **void** print(Set<Integer> set) {  System.***out***.println(">> 배열 원소 출력");  **for** (**int** temp : set)  System.***out***.print(temp + " ");  System.***out***.println();  }  } |
| **[실행결과]** |

1. 내용 점검 1번 문제에서 ArrayList에 객체를 저장할 저장 할 때 오름차순으로 저장될 수 있도록 수정 하시오.

|  |  |
| --- | --- |
|  | **public** **class** ArrayTest {  **public** **static** **void** main(String[] args) {  ArrayList<Person> alist=**new** ArrayList<>();  ListExam.*add*(alist, **new** Person("성우", 179));  ListExam.*add*(alist, **new** Person("태동", 173));  ListExam.*add*(alist, **new** Person("인국", 183));  ListExam.*add*(alist, **new** Person("용권", 180));  ListExam.write(alist);  }  } |

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.\*;  **public** **class** ListExam {  **public** **static** **void** add(ArrayList<Person> alist, Person person) {  **int** asize = alist.size(); // alist.size가 계속 바뀌므로 변수에 저장해서 사용  **if** (asize == 0) { // list에 처음들어갈때  alist.add(person);  } **else** {  **for** (**int** i = 0; i < asize; i++) {  **if** (person.getHeight() < alist.get(i).getHeight()) { // 그 자리의 값보다 작으면 그 자리에 삽입  alist.add(i, person);  } **else** **if** (i == (asize - 1)) { // 저장되있는 값들보다 클때  alist.add(person);  }  }  }  }  **public** **static** **void** write(ArrayList<Person> alist) {  **for** (Person temp : alist)  System.***out***.println(temp);  }  } |
| **[실행결과]** |

1. 다음과 같은 객체를 LinkedList에 저장하고 메뉴에 따라 처리하는 프로그램을 작성하시오. 회원 탈퇴를 선택하면 LinkedList에서 해당 자료를 삭제한 후 저장된 모든 목록을 출력하여 삭제 되었음을 확인한다. 다만 해당 자료가 없으면 제시된 결과처럼 출력되도록 한다



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

**public** **class** Customer {

**private** String name; //회원명

**private** String cnum; //id

**private** **int** year, month; //가입년월

**public** Customer(String name, String cnum) {

**this**.name = name;

**this**.cnum = cnum;

dateCreat();

}

**void** dateCreat() {

Calendar now = Calendar.*getInstance*(); //Calendar 객체 생성

year = now.get(Calendar.***YEAR***); //년

month = now.get(Calendar.***MONTH***) + 1; //월 시작이 0

}

@Override

**public** String toString() {

**return** "Customer [name=" + name + ", id=" + cnum + ", year=" + year + ", month=" + month + "]";

}

**public** String getCnum() {

**return** cnum;

}

}

**class** CustomerManage {

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

//링크드 리스트에 Customer 객체를 저장하는 인스턴스 메소드, 리스트에 저장할 객체를 매개변수로 받으며 반환값 없음

//링크드 리스트에 저장된 객체를 출력하는 인스턴스 메소드, 반환값 없음

//매개변수로 받은 아이디를 검색하여 찾으면 true, 찾지 못하면 false를 반환하는 인스턴스 메소드

}

**public** **class** CustomerTest {

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

//메뉴 처리

}

}

|  |
| --- |
| **[프로그램 소스]**  **import** java.util.Scanner;  **public** **class** CustomerTest {  **public** **static** **void** main(String[] args) {  // 메뉴처리  Scanner scan = **new** Scanner(System.***in***);  CustomerManage cm = **new** CustomerManage();  **while** (**true**) {  System.***out***.print("1:회원가입 2:회원탈퇴 3:모든 회원 정보 출력 4:종료 ----> ");  **int** menu = scan.nextInt();  **if** (menu == 1) {  System.***out***.print("이름을 입력하세요: ");  String name = scan.next();  System.***out***.print("아이디를 입력하세요: ");  String cnum = scan.next();  cm.add(**new** Customer(name, cnum));  } **else** **if** (menu == 2) {  System.***out***.print("아이디를 입력하세요: ");  cm.delete(scan.next());  } **else** **if** (menu == 3) {  cm.print();  } **else** **if** (menu == 4) {  System.***out***.println("프로그램을 종료합니다.");  **break**;  } **else** {  System.***out***.println("입력 오류");  }  }  }  }  **import** java.util.\*;  **public** **class** CustomerManage {  LinkedList<Customer> customers = **new** LinkedList<>();  // 링크드 리스트에 Customer 객체를 저장하는 인스턴스 메소드, 리스트에 저장할 객체를 매개변수로 받으며 반환값 없음  **public** **void** add(Customer c) {  customers.add(c);  }  // 링크드 리스트에 저장된 객체를 출력하는 인스턴스 메소드, 반환값 없음  **public** **void** print() {  System.***out***.println("----- 회원정보를 출력합니다. -----");  **for** (Customer temp : customers)  System.***out***.println(temp);  }  // 매개변수로 받은 아이디를 검색하여 찾으면 true, 찾지 못하면 false를 반환하는 인스턴스 메소드  **public** **boolean** search(String id) {  **for** (Customer temp : customers)  **if** (temp.getCnum().equals(id))  **return** **true**;  **return** **false**;  }  //회원 삭제 메소드  **public** **void** delete(String id) {  **if** (!search(id)) {  System.***out***.println("해당 아이디가 없습니다.");  **return**;  }  **for** (Customer temp : customers)  **if** (temp.getCnum().equals(id))  customers.remove(temp);  System.***out***.println("삭제 되었습니다.");  print();  }  }  **import** java.util.\*;  **public** **class** Customer {  **private** String name; // 회원명  **private** String cnum; // id  **private** **int** year, month; // 가입년월  **public** Customer(String name, String cnum) {  **this**.name = name;  **this**.cnum = cnum;  dateCreat();  }  **void** dateCreat() {  Calendar now = Calendar.*getInstance*(); // Calendar 객체 생성  year = now.get(Calendar.***YEAR***); // 년  month = now.get(Calendar.***MONTH***) + 1; // 월 시작이 0  }  @Override  **public** String toString() {  **return** "Customer [name=" + name + ", id=" + cnum + ", year=" + year + ", month=" + month + "]";  }  **public** String getCnum() {  **return** cnum;  }  } |
| **[실행결과]** |