12. 고객관리 프로그램 II

6장에서 만들었던 고객의 정보를 관리하는 프로그램을 객체지향 프로그램의 개념을 적용하고, 컬렉션 프레임워크 이용하여 고객의 정보를 저장하도록 수정할 수 있습니다.

12.1. 요구사항 정의

고객의 정보는 이름, 성별, 이메일, 출생년도가 있습니다. 고객의 정보를 입력받아 리스트(List)에 저장해야 합니다. 이름은 문자열로 저장하며, 성별은 남자는 M, 여자는 F로 저장합니다. 이메일은 문자열로 저장하며 태어난 연도는 정수로 저장합니다.

고객 관리 프로그램은 고객의 정보를 저장, 조회, 수정, 삭제할 수 있는 기능이 있어야 합니다. 고객 정보를 파일에 저장하는 기능을 구현하지 않아도 됩니다.

I를 눌러 고객의 정보를 입력받도록 하며, 저장된 고객 정보는 P 또는 N을 눌러 이전 고객 정보 또는 다음 고객정보를 조회할 수 있어야 합니다. 조회한 고객 정보는 U를 눌러 새로운 정보로 수정할 수 있어야 합니다. D를 누르면 조회한 고객 정보를 배열에서 삭제해야 합니다. 프로그램의 종료는 Q를 누릅니다.

객체지향 개념을 적용하여 확장성을 고려한 애플리케이션이 되도록 해야 합니다.

12.2. Customer 클래스

요구사항 정의 단계에서 객체지향 개념을 적용하여 확장성을 고려한 애플리케이션을 만들기 위해서 관리해야 할 데이터를 하나의 클래스에 모아야 합니다. 그래서 고객의 정보를 저장하기 위해 고객정보를 추상화한 Customer 클래스를 정의합니다. 데이터를 저장하기 위한 용도로 만든 클래스를 VO(Value Object)[37]라고 부릅니다. VO 클래스에는 시스템의 기능을 구현하지는 않습니다. 데이터를 저장하는 용도로만 사용합니다.

고객의 정보는 이름, 성별, 이메일, 출생년도가 있습니다. 이를 객체 추상화 하여 Customer 클래스를 정의하면 다음과 같습니다.

Customer.java

```
1:  public class Customer {
2:     private String name
3:     private char gender
4:     private String email
5:     private int birthYear
6:     //생성자, setter/getter, toString 구현
7:  }
```

앞의 코드에 생성자, setter/getter, toString() 메서드를 추가하여 완성한 코드는 다음과

37) DTO(Data Transfer Object) 또는 TO(Transfer Object)라고 부르기도 합니다.

같습니다. 이클립스에서 [Source] 메뉴를 이용하면 쉽게 추가할 수 있습니다.

Customer.java

```
1:  public class Customer {
2:     private String name;
3:     private char gender;
4:     private String email;
5:     private int birthYear;
6:
7:     public Customer(String name, char gender, String email, int birthYear) {
8:        super();
9:        this.name = name;
10:       this.gender = gender;
11:       this.email = email;
12:       this.birthYear = birthYear;
13:    }
14:
15:    public String getName() {
16:       return name;
17:    }
18:    public void setName(String name) {
19:       this.name = name;
20:    }
21:    public char getGender() {
22:       return gender;
23:    }
24:    public void setGender(char gender) {
25:       this.gender = gender;
26:    }
27:    public String getEmail() {
28:       return email;
29:    }
30:    public void setEmail(String email) {
31:       this.email = email;
32:    }
33:    public int getBirthYear() {
34:       return birthYear;
35:    }
36:    public void setBirthYear(int birthYear) {
37:       this.birthYear = birthYear;
38:    }
39:
40:    @Override
41:    public String toString() {
42:       return "Customer [name=" + name + ", gender=" + gender + ", email=" + email
    + ", birthYear=" + birthYear + "]";
43:    }
44: }
```

12.3. CustomerManager 클래스

다음 코드는 고객의 정보를 저장하는 배열 자료구조를 ArrayList을 이용하도록 변경했습니다. 진하게 작성된 코드가 6장의 CustomerManager 클래스에서 수정된 부분입니다.

CustomerManager.java

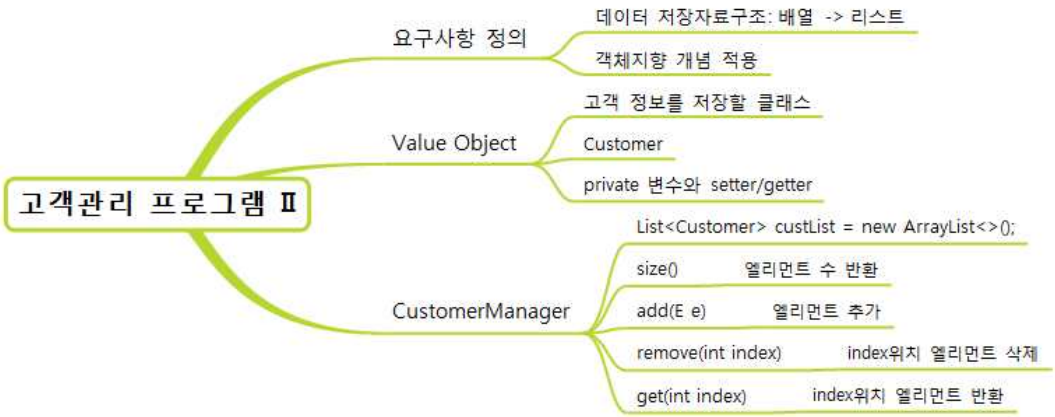
```
1:  import java.util.ArrayList;
2:  import java.util.Scanner;
3:
4:  public class CustomerManager {
5:
6:     //고객 정보를 저장할 자료구조 선언
7:     static ArrayList<Customer> custList = new ArrayList<>();
8:
9:     //리스트 정보를 조회하기 위해 인덱스를 필요로 함
10:    static int index = -1;
11:
12:    static int count = 0;   //custList.size()로 대체 가능
13:
14:    //기본 입력장치로부터 데이터를 입력받기 위해 Scanner객체 생성
15:    static Scanner scan = new Scanner(System.in);
16:
17:    public static void main(String[] args) {
18:
19:       while(true) {
20:          count = custList.size();
21:          System.out.printf("\n[INFO] 고객 수 : %d, 인덱스 : %d\n", count, index);
22:          System.out.println("메뉴를 입력하세요.");
23:          System.out.println("(I)nsert, (P)revious, (N)ext, " +
24:                "(C)urrent, (U)pdate, (D)elete, (Q)uit");
25:          System.out.print("메뉴 입력: ");
26:          String menu = scan.next();
27:          menu = menu.toLowerCase();        //입력한 문자열을 모두소문자로 변환
28:          switch(menu.charAt(0)) {
29:          case 'i':
30:             System.out.println("고객정보 입력을 시작합니다.");
31:             insertCustomerData();
32:             System.out.println("고객정보를 입력했습니다.");
33:             break;
34:          case 'p' :
35:             System.out.println("이전 데이터를 출력합니다.");
36:             if(index <= 0) {
37:                System.out.println("이전 데이터가 존재하지 않습니다.");
38:             }else {
39:                index--;
40:                printCustomerInfo(index);
41:             }
```

```
42:             break;
43:          case 'n' :
44:             System.out.println("다음 데이터를 출력합니다.");
45:             if(index >= count-1) {
46:                System.out.println("다음 데이터가 존재하지 않습니다.");
47:             }else {
48:                index++;
49:                printCustomerInfo(index);
50:             }
51:             break;
52:          case 'c' :
53:             System.out.println("현재 데이터를 출력합니다.");
54:             if( (index >= 0) && (index < count)) {
55:                printCustomerInfo(index);
56:             }else {
57:                System.out.println("출력할 데이터가 선택되지 않았습니다.");
58:             }
59:             break;
60:          case 'u' :
61:             System.out.println("데이터를 수정합니다.");
62:             if( (index >= 0) && (index < count)) {
63:                System.out.println(index + "번째 데이터를 수정합니다.");
64:                updateCustomerData(index);
65:             }else {
66:                System.out.println("수정할 데이터가 선택되지 않았습니다.");
67:             }
68:             break;
69:          case 'd' :
70:             System.out.println("데이터를 삭제합니다.");
71:             if( (index >= 0) && (index < count)) {
72:                System.out.println(index + "번째 데이터를 삭제합니다.");
73:                deleteCustomerData(index);
74:             }else {
75:                System.out.println("삭제할 데이터가 선택되지 않았습니다.");
76:             }
77:             break;
78:          case 'q' :
79:             System.out.println("프로그램을 종료합니다.");
80:             scan.close();          //Scanner 객체를 닫아준다.
81:             System.exit(0);        //프로그램을 종료시킨다.
82:             break;
83:          default :
84:             System.out.println("메뉴를 잘 못 입력했습니다.");
85:          }//end switch
86:       }//end while
87:    }//end main
88:
89:    public static void insertCustomerData() {
90:       System.out.print("이름 : ");
91:       String name = scan.next();
```

```
92:         System.out.print("성별(M/F) : ");
93:         char gender = scan.next().charAt(0);
94:         System.out.print("이메일 : ");
95:         String email = scan.next();
96:         System.out.print("출생년도 : ");
97:         int birthYear = scan.nextInt();
98:
99:         //입력받은 데이터로 고객 객체를 생성
100:        Customer cust = new Customer(name, gender, email, birthYear);
101:
102:        //고객 객체를 ArrayList에 저장
103:        custList.add(cust);
104:    }
105:
106:    //고객데이터 출력
107:    public static void printCustomerInfo(int index) {
108:        Customer cust = custList.get(index);
109:        System.out.println("==========CUSTOMER INFO===============");
110:        System.out.println("이름 : " + cust.getName());
111:        System.out.println("성별 : " + cust.getGender());
112:        System.out.println("이메일 : " + cust.getEmail());
113:        System.out.println("출생년도 : " + cust.getBirthYear());
114:        System.out.println("=====================================");
115:    }
116:
117:    //index 위치의 고객정보를 삭제합니다.
118:    public static void deleteCustomerData(int index) {
119:        custList.remove(index);
120:    }
121:
122:    //index 위치의 고객 정보를 수정합니다.
123:    public static void updateCustomerData(int index) {
124:        Customer cust = custList.get(index);
125:        System.out.println("---------UPDATE CUSTOMER INFO----------");
126:        System.out.print("이름(" + cust.getName() + ") :");
127:        cust.setName(scan.next());
128:
129:        System.out.print("성별(" + cust.getGender() + ") :");
130:        cust.setGender(scan.next().charAt(0));
131:
132:        System.out.print("이메일(" + cust.getEmail() + ") :");
133:        cust.setEmail(scan.next());
134:
135:        System.out.print("출생년도(" + cust.getBirthYear() + ") :");
136:        cust.setBirthYear(scan.nextInt());
137:    }
138:
139:}//end class
```


```
[INFO] 고객 수 : 0, 인덱스 : -1
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: I
고객정보 입력을 시작합니다.
이름 : 허현정
성별(M/F) : M
이메일 : heojk3@daum.net
출생년도 : 2006
고객정보를 입력했습니다.

[INFO] 고객 수 : 1, 인덱스 : -1
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: I
고객정보 입력을 시작합니다.
이름 : 허현준
성별(M/F) : M
이메일 : hhjun@daum.net
출생년도 : 2007
고객정보를 입력했습니다.

[INFO] 고객 수 : 2, 인덱스 : -1
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: I
고객정보 입력을 시작합니다.
이름 : 허현수
성별(M/F) : M
이메일 : -
출생년도 : 2010
고객정보를 입력했습니다.

메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: N
다음 데이터를 출력합니다.
==========CUSTOMER INFO================
이름 : 허현정
성별 : M
이메일 : heojk3@daum.net
출생년도 : 2006
========================================

[INFO] 고객 수 : 3, 인덱스 : 0
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: N
다음 데이터를 출력합니다.
==========CUSTOMER INFO================
이름 : 허현준
성별 : M
이메일 : hhjun@daum.net
출생년도 : 2007
========================================

[INFO] 고객 수 : 3, 인덱스 : 1
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: N
다음 데이터를 출력합니다.
==========CUSTOMER INFO================
이름 : 허현수
성별 : M
이메일 : -
출생년도 : 2010
========================================

[INFO] 고객 수 : 3, 인덱스 : 2
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력: U
데이터를 수정합니다.
2번째 데이터를 수정합니다.
---------UPDATE CUSTOMER INFO----------
이름(허현수) :허현수
성별(M) :M
이메일(-) :heojk2@daum.net
출생년도(2010) :2010

[INFO] 고객 수 : 3, 인덱스 : 2
메뉴를 입력하세요.
(I)nsert, (P)revious, (N)ext, (C)urrent, (U)pdate, (D)elete, (Q)uit
메뉴 입력:
```

12.4. 마인드맵 정리


고객관리 프로그램 II
요구사항 정의
데이터 저장자료구조: 배열 -> 리스트
객체지향 개념 적용
Value Object
고객 정보를 저장할 클래스
Customer
private 변수와 setter/getter
CustomerManager
List<Customer> custList = new ArrayList<>();
size() 엘리먼트 수 반환
add(E e) 엘리먼트 추가
remove(int index) index위치 엘리먼트 삭제
get(int index) index위치 엘리먼트 반환