**GỢI Ý THIẾT KẾ VÀ HIỆN THỰC CHƯƠNG TRÌNH QUẢN LÝ NHÓM CƠ BẢN**

**Một số khái niệm:**

* Dữ liệu nhập bị ràng buộc: Dữ liệu phải thỏa mãn điều kiện về trị.

Thí dụ 1: mã sinh viên phải có dạng CCxxxx trong đó C là ký tự chữ hoa, x là ký số hệ 10

Thí dụ 2: giá bán phải trong khoảng 0..1000000,

….

* Dữ liệu xuất phải có dạng định trước:

Thí dụ 1: giá bán phải xuất dạng dùng dấu phẩy phân cách ngàn và có ký hiệu $ - 123,000$

Thí dụ 2: thông tìn ngày thàng được xuất dạng dd/mm/yyyy

**Các lớp cần có của một chương trình quản lý nhóm**

Các tác vụ trên nhóm: Thêm, tìm, xóa, sửa, duyệt, sắp xếp, nạp nhóm từ file, ghi nhóm lên file.

Một chương trình quản lý nhóm thường đòi hỏi các ràng buộc về dữ liệu nhập xuất

Như vậy, khi thiết kế một chương trình quản lý nhóm, chúng ta nên thiết kế như sau:

1. Thiết kế các class cho việc nhập dữ liệu (MyInput), xuất dữ liệu (MyOutput) trong đó có các hành vi cho việc nhập dữ liệu có kiểm tra, xuất dữ liệu theo định dạng được quy định của bài toán.
2. Thiết kế lớp Menu để người dùng chọn chức năng của chương trỉnh hoặc chọn 1 dữ liệu trong một nhóm khi chạy chương trình.
3. Thiết kế lớp mô tả một phần tử của nhóm.
4. Thiết kế lớp mô tả cho một nhóm trong đó các tác vụ trên nhóm theo đề bài được hiện thực.
5. Thiết kế lớp chứa hàm main() của chương trình.

**Gợi ý về thiết kế lớp mô tả cho một phần tử**

***public class Item implements Comparable*<Item> {**

String ID; …… //các field

**public Item()** { // constructor chuẩn

… code

}

**public Item(String ID)** { // constructor 1 tham số khi có nhu cầu tìm kiếm theo ID

this.ID = ID;

}

**public Item(**đủ các tham số**)** { // constructor để khởi tạo một Item

<gán tham số vào các field >

}

// Hành vi phục vụ cơ chế so sánh chuẩn

**public int compareTo** (Item obj){ // so sánh Item này với Item trong tham số

Dựa vào đề bài để xác định field cơ bản được dùng để so sánh và thứ tự trước sau

}

**public String toString** (){ // chuyển các field thành dạng String cho việc xuất data ra màn hình

Dựa vào đề bài để xác định thứ tự các field sẽ được xuất và dạng xuất phù hợp

}

Nếu có ghi đối tượng lên file:

Tình huống 1: Ghi data lên file giống như data được xuất ra màn hình: Dùng toString()

Tình huống 2: Ghi data lên file KHÁC với data được xuất ra màn hình: Hiện thực hành vi sau:

**public String toFile** (){ // chuyển các field thành dạng String cho việc xuất ghi data lên file

Dựa vào đề bài để xác định thứ tự các field sẽ được ghi lên file để viết code phù hợp

}

**Không nên hiện thực hành vi xuất nhập trong lớp Item vì khi quản lý nhóm thường đòi hỏi ID không trùng nhau. Do vậy việc nhập từng field nên để trong lớp mô tả cho nhóm.**

**}**

**Gợi ý về thiết kế lớp mô tả cho một NHÓM**

***Import java.util.Collections;***

***public class ItemList extends ArrayListe*<Item> {**

**public ItemList()** { // constructor chuẩn

super(); // gọi constructor chuẩn của lớp cha

}

**public int searchID (String ID) {** // Tìm Item dựa trên ID

**}**

**public void addItem()** {

<Khai báo các biến cho các field của một Item>

Do {

Nhập ID

}

While (this.searchItem(ID) >= 0);

Nhập các fields còn lại thỏa điều kiện của bài toán

Item newIt = new Item ( các đối số đã nhập);

this.add (it);

Xuất “Đã thêm “ + newIt;

}

**public void searchItem()** {

String ID

Nhập ID

Int pos = search(ID);

If (pos<0) Xuất (“Not existed!”);

Else Xuất this.get(pos);

}

**public void removeItem()** {

String ID

Nhập ID

Int pos = search(ID);

If (pos<0) Xuất (“Not existed!”);

Else {

This.remove(pos);

}

}

**public void updateItem()** {

<Khai báo các biến cho các field của một Item>

Nhập ID

Int pos = searchID(ID);

If (pos <0) Xuất “Nbot Existed!”;

Else {

Item it = this.get (pos); // lấy đối tượng sẽ sửa

Nhập các fields còn lại thỏa điều kiện của bài toán

Gán lại các field của itl

Xuất “Updated.”

}

}

**public void printConditions()** { // xuất các Item thỏa điều kiện

<Khai báo các biến cho các dữ liệu điều kiện>

Nhập các dữ liệu điều kiện;

Boolean existed = false;

For (Item it: this) {

If (it thỏa điều kiện) {

Existed = true;

Xuất it;

}

}

If (existed == false) Xuất (“Not found!”);

}

**public void printConditions()** { // xuất các Item thỏa điều kiện

<Khai báo các biến cho các dữ liệu điều kiện>

Nhập các dữ liệu điều kiện;

Boolean existed = false;

For (Item it: this) {

If (it thỏa điều kiện) {

Existed = true;

Xuất it;

}

}

If (existed == false) Xuất (“Not found!”);

}

**public void printAll()** { // xuất các Item có sắp xếp

if (this.size()==0) Xuất “Empty list”;

else {

Collections.sort(this);

For (Item it: this) {

Xuất it;

}

}

**public void loadFromFile(**String fname**)** { // nạp danh sách từ file

Tùy dạng file văn bản hay file nhị phân mà viết code phù hợp

}

**public void saveFromFile(**String fname**)** { // ghi danh sách lên file

Tùy dạng file văn bản hay file nhị phân mà viết code phù hợp

}

**}**

**Gợi ý về thiết kế lớp có hàm main**

***public class ItemMng* {**

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

String fname = “tên File Trong Đề Bài”;

Khai báo menu;

Int userChoice;

ItemList list = new ItemList(); // khởi tạo nhóm trống

List.addFromFile(fname); // nạp ds từ file

// xuất menu cho user chạy chương trình

Do {

userChoice = menu.getUserChoice();

switch (userChoice) {

case 1: ……

}

}

}