PHIẾU BÀI TẬP MINH HỌA COLLECTION - EXCEPTION

Bài tập áp dụng 2 slide (58,59) bài 7

Yêu cầu:

Thông tin Thí sinh dự thi được mô tả thông qua: SBD, tên thí sinh, điểm toán, điểm lý, điểm hóa. Yêu cầu tên, mã không được nhập trống, điểm phải nằm trong khoảng 0-10. SBD không được trùng nhau

Mỗi thí sinh dự thi được xếp vào phòng thi. Thông tin về phòng thi bao gồm: mã phòng thi, địa điểm thi, số lượng thí sinh trong phòng, danh sách thí sinh

Cài đặt lớp chứa hàm main thực hiện 8 yêu cầu thực hiện tại 1 điểm thi.

Phần mở rộng sinh viên tự hoàn thiện: Đưa ra thông tin thí sinh điểm cao nhất, thấp nhất

Chương trình có menu cho chọn:

1. Nhập danh sách phòng thi. Nhập danh sách thi sinh trong phòng. (chú ý số thí sinh không vượt quá số chỗ trong phòng, bắt lỗi dữ liệu)
2. In danh sách sau khi nhập theo dạng bảng
3. Nhập mã thí sinh. Kiểm tra ts có trong danh sách không. Xóa thí sinh có mã vừa nhập
4. Nhập mã thí sinh. Kiểm tra thí sinh có trong danh sách không. Sửa thông tin thí sinh có mã vừa nhập
5. Tìm kiếm 1 thí sinh theo mã hay theo vị trí. Hiển thị thí sinh tìm được
6. Sắp xếp thí sinh trong phòng tăng dần theo tên. Tên trùng nhau sắp theo mã
7. Lưu thông tin vào file
8. Đọc thông tin phòng thi từ file

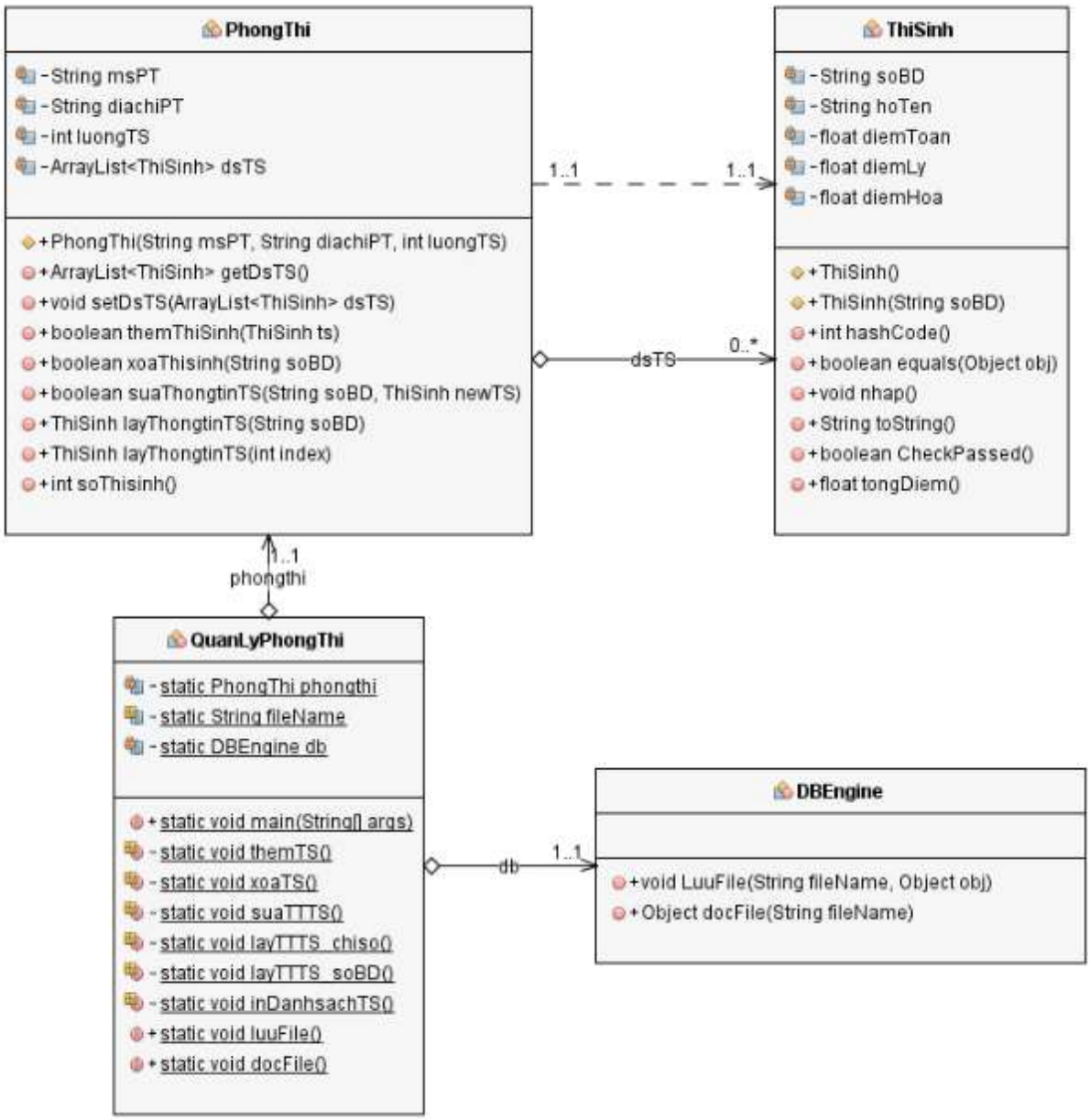
Gợi ý thực hiện

Sơ đồ lớp và mối quan hệ lớp:

- Thí sinh lưu thông tin về các thí sinh dự thi: số báo danh, họ tên và các điểm toán, lý hóa

- Phòng thi: mã phòng, địa điểm, lượng thí sinh dự thi và danh sách thí sinh trong phòng
- DBEngine: lớp quản lý đọc ghi file
- Quản lý phòng thi chứa hàm main thực hiện 8 nhiệm vụ chương trình

Sơ đồ lớp:



Gợi ý cài đặt:

ThiSinh.java

```
//Bước1: khai báo lớp vàd thuộc tính và lớp
public class ThiSinh implements Serializable{
    private String soBD;
    private String hoTen;
    private float diemToan;
```

```
    private float diemLy;
    private float diemHoa;
```

```
//Bước 2: xây dựng hàm tạo
public ThiSinh() {
    soBD="";hoTen="no-name";diemToan=diemLy=diemHoa=0f;
}
public ThiSinh(String soBD) {
    this.soBD = soBD;
  }
```

```
//bước 3 ghi đè phương thức nhận diện khóa
 @Override
 public int hashCode() {
     int hash = 5;
     hash = 59 * hash + Objects.hashCode(this.soBD);
     return hash;
 }
 @Override
 public boolean equals(Object obj) {
     if (obj == null) {
         return false;
     }
     if (getClass() != obj.getClass()) {
         return false;
     }
     final ThiSinh other = (ThiSinh) obj;
     if (!Objects.equals(this.soBD, other.soBD)) {
         return false;
     }
     return true;
 }
```

```
/** bước 4: Ứng với mỗi data instance ở Bước 2, tạo các phương
    *  thức setters/getters.
    *  Cũng theo tính chất Encapsulation kiểm soát thay đổi
    *  giá trị của  thuộc tính đảm đảo tính toàn vẹn dữ liệu.
    */
```

```
//xử lý bắt lỗi hợp lệ dữ liệu số báo danh không trống
public void setSoBD(String soBD) throws Exception{
    if(soBD.trim().equals(""))
        throw new Exception("Số báo danh không được trống!");
    this.soBD = soBD;
}
```

```
//xử lý bắt lỗi hợp lệ dữ liệu tên không trống
public void setHoTen(String hoTen)throws Exception {
    if(hoTen.trim().equals(""))
        throw new Exception("Họ tên không được trống!");
    this.hoTen = hoTen;
}
//xử lý bắt lỗi hợp lệ dữ liệu điểm trong khoảng 0 đến 10
public void setDiemToan(float diemToan) throws Exception {
    if(diemToan<0||diemToan>10)
           throw new Exception("Điểm toan không hợp lệ");
    this.diemToan = diemToan;
}
public void setDiemLy(float diemLy) throws Exception{
    if(diemLy<0||diemLy>10)
        throw new Exception("Điểm ly không hợp lệ");
    this.diemLy = diemLy;
}
public void setDiemHoa(float diemHoa) throws Exception{
    if(diemHoa<0||diemHoa>10)
        throw new Exception("Điểm hoa không hợp lệ");
    this.diemHoa = diemHoa;
}
```

```
//bước 5 nhập thông tin ts
public void nhap(){
    try {
        Scanner sc=new Scanner(System.in);
        System.out.print("\tNhập số báo danh:");
        setSoBD(sc.nextLine());
        System.out.print("\tNhập họ tên thí sinh:");
        setHoTen(sc.nextLine());
        System.out.print("\tNhập điểm toán:");
        setDiemToan(sc.nextFloat());
        System.out.print("\tNhập điểm lý:");
        setDiemLy(sc.nextFloat());
        System.out.print("\tNhập điểm hóa:");
        setDiemHoa(sc.nextFloat());
    } catch (Exception ex) {
        System.out.print("có lỗi: "+ex.toString());
    }
}
```

```
//bước 6: xuất thông tin đối tượng
```

```
@Override
public String toString() {
    String tsValue;
    tsValue=      "SBD:"+soBD+"\t      Ho      ten:"+hoTen+"\tdiem
toan:"+diemToan+"\t điểm lý";
    tsValue=tsValue +diemLy+"\tđiem hóa"+diemHoa;
    return tsValue;
}
```

```
/*
 * Bước 7: Viết các business methods của đối tượng.
Đây là bước quan trọng nhất bởi nó định nghĩa tập các dịch vụ
của đối tượng cung cấp cho bên ngoài.
 */
```

```
/**
* Kiểm tra 1 thí sinh có đậu hay không
* @return true nếu sinh viên có tổng số điểm trên 15 và không
* có điểm nào dưới 3.
*/
public boolean CheckPassed() {
    return tongDiem()>15 && diemToan>=3 && diemHoa>=3 && diemLy>=3;
}
```

```
/**
* Tổng điểm của thí sinh
* @return: tổng điểm
*/
public float tongDiem() {
    return diemHoa+diemLy+diemToan;
}
```

```
}//end of class ThiSinh
```

PhongThi.java

```
//Khai báo lớp và thuộc tính
```

```
public class PhongThi implements Serializable {
    //các thuộc tính của đối tượng phòng thi
    private String msPT;
    private String diachiPT;
    private int luongTS;
    //đối tượng dùng chứa tập các thí sinh
    private ArrayList<ThiSinh> dsTS;
```

```
// đinh nghĩa các phương thức khởi tạo
```

```
public PhongThi(String msPT, String diachiPT, int luongTS) {
    this.msPT = msPT;
    this.diachiPT = diachiPT;
    this.luongTS = luongTS;
    dsTS=new ArrayList<ThiSinh>();
}
```

```
//get/set danh sách thí sinh
public ArrayList<ThiSinh> getDsTS() {
        return dsTS;
}
public void setDsTS(ArrayList<ThiSinh> dsTS) {
        this.dsTS = dsTS;
}
```

```
/**
* Thêm 1 thí sinh vào phòng thi có kiểm tra trùng mã
* @param ts: thí sinh thêm vào
* @return true nếu việc thêm thành công
*/
public boolean themThiSinh(ThiSinh ts) {
//Nếu thí sinh đã tồn tại thì không cho thêm
    if(dsTS.contains(ts)) return false;
    if(dsTS.size()+1>luongTS)//nếu đã đủ lượng thí sinh
    {
        System.out.println("đã đủ số lượng thí sinh trong phòng.");
        return false;
    }
    else
    {   dsTS.add(ts);
        return true;
    }
}
```

```
/* Xóa 1 thí sinh khỏi phòng thi
* @param soBD là số báo danh của thí sinh cần xóa
* @return trả về true nếu xóa thành công
*/
public boolean xoaThisinh(String soBD) {
    ThiSinh ts=new ThiSinh(soBD);
    if(!dsTS.contains(ts))    return false;
    else
    {   dsTS.remove(ts);
        return true;
```

```
    }
}
```

```
/**
* Sửa thông tin thí sinh
* @param soBD: số DB của thí sinh cần sử thông tin
* @param newTS: thông tin mới cần cập nhật
* @return true nếu sửa chữa thành công
*/
 public boolean suaThongtinTS(String soBD, ThiSinh newTS) {
    ThiSinh ts=new ThiSinh(soBD);
    if(!dsTS.contains(ts))
          return false;
    dsTS.set(dsTS.indexOf(ts),newTS);
    return true;
}
```

```
/**
* Lấy thông tin của 1 thí sinh khi biết số báo danh
* @param soBD số báo danh của ts cần lấy thông tin
* @return null nếu không lấy được
*/
public ThiSinh layThongtinTS(String soBD) {
    ThiSinh ts=new ThiSinh(soBD);
    if(!dsTS.contains(ts))
          return null;
    ts=dsTS.get(dsTS.indexOf(ts));
    return ts;
}
```

```
/**
* Lấy thông tin của 1 thí sinh khi biết
*  số thứ tự của ts đó trong danh sách
* @param index :số thứ tự của ts
* @return null nếu không thành công
*/
public ThiSinh layThongtinTS(int index){
    if(index<0||index>dsTS.size())
          return null;
    return dsTS.get(index);
}
```

```
/**
* Lấy số thí sinh thực sự đang có trong phòng thi
* @return
```

```
*/
public int soThisinh() {
    return dsTS.size();
}
```

```
}//end of class phòng thi
```

DBEngine.java

```
//khai báo lớp
public class DBEngine {
```

```
/**
* Serial 1 đối tượng xuống file
* @param fileName file chỉ định
* @param obj: đối tượng cần serial
* @throws Exception
*/
public void LuuFile(String fileName, Object obj)
        throws Exception{
    //Tạo luồng ghi file
    FileOutputStream fs=new FileOutputStream(fileName);
    //Tạo luồng để serial đối tượng
    ObjectOutputStream os=new ObjectOutputStream(fs);
    //chuyển tải đối tượng tới đích (tập tin)
    os.writeObject(obj);
    //đóng luồng
    fs.close();
    os.close();
}
```

```
/**
* Khôi phục(deserial) 1 đối tượng đã được serial trước đó
* lên bộ nhớ.
* @param fileName: file chỉ định
* @return đối tượng đã được phục hồi
* @throws Exception
*/
public Object docFile(String fileName) throws Exception{
    Object kq=null;
    //Tạo luồng đọc file đã được serial
    FileInputStream fi=new FileInputStream(fileName);
    //Tạo luồng để Deserialize đối tượng
    ObjectInputStream ois=new ObjectInputStream(fi);
```

```
    //Tiến hành khôi phục đối tượng
    kq=ois.readObject();
    //đóng luồng
    fi.close();ois.close();
    return kq;
}
```

```
}//end of class
```

QuanLyPhongThi.java

```
//khai báo lớp
public class QuanLyPhongThi {
```

```
//khai báo các thuộc tính tổng thể quan trọng
private static PhongThi phongthi=null;
static String fileName="d:thiSinh.txt";
private static DBEngine db=new DBEngine();
```

```
//Xây dựng menu
static void menu(){
 System.out.println("================================");
    System.out.println("1. Thêm Thí sinh mới");
    System.out.println("2. Hiệu chỉnh thông tin thí sinh");
    System.out.println("3. Xóa thí sinh khỏi phòng thi");
    System.out.println("4. Lấy thông tin Thí sinh khi biết số báo
danh.");
    System.out.println("5. Lấy thông tin Thí sinh khi biết số thứ
tự.");
    System.out.println("6. In danh sách thí sinh");
    System.out.println("7. Lưu phòng thi xuống file");
    System.out.println("8. Đọc thông tin phòng từ từ file");
    System.out.println("9. Thoát");
    System.out.println("******************");
    System.out.print("\t**Chọn lựa của bạn? <1->9>:");
}
```

```
//thêm thí sinh vào phòng thi
static void themTS() {
    ThiSinh ts=new ThiSinh();
    ts.nhap();
    if(phongthi.themThiSinh(ts))
        System.out.println("Thêm thành công");
    else
        System.out.println("Không thêm được.");
}
```

```
//xóa thí sinh khỏi phòng thi.
static void xoaTS() {
    Scanner input=new Scanner(System.in);
    System.out.print("\tNhập số báo danh cần xóa:");
    String soBD=input.nextLine();
    input.nextLine();// dua con tro xuong dong tiep, tranh lay ky
ty enter
    if(phongthi.xoaThisinh(soBD))
        System.out.println("Xóa thành công");
    else
        System.out.println("Không xóa được!");
}
```

```
//sửa thông tin thí sinh
static void suaTTTS()  {
    Scanner input=new Scanner(System.in);
    System.out.print("\tNhập số báo danh cần sửa:");
    String soBD=input.nextLine();
    ThiSinh newTS=new ThiSinh();
    newTS.nhap();
    if(phongthi.suaThongtinTS(soBD, newTS))
         System.out.println("Sửa thành công");
    else
         System.out.println("Không sửa được!");
}
```

```
//lấy thông tin thí sinh qua chỉ số
static void layTTTS_chiso() {
    Scanner input=new Scanner(System.in);
    System.out.println("Nhập số thứ tự cần lấy thông tin:");
    int stt=input.nextInt();
    ThiSinh ts= phongthi.layThongtinTS(stt);
    if(ts==null)
        System.out.println("không có");
    else
        System.out.println(ts);
}
```

```
//lấy thông tin thí sinh qua số báo danh
static void layTTTS_soBD() {
    Scanner input=new Scanner(System.in);
    System.out.print("\tNhập số báo danh cần lấy thông tin:");
    String soBD=input.nextLine();
    ThiSinh ts=phongthi.layThongtinTS(soBD);
```

```
    System.out.println("~~~~~~~~~~Kết quả~~~~~~~~~~~~~~");
    if(ts==null)
        System.out.println("Không có");
    else
        System.out.println(ts);
        System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
}
```

```
//in danh sách thí sinh
static void inDanhsachTS() {
    System.out.println("==========DANH SÁCH THÍ SINH=========");
    System.out.println("Số BD \tHọ tên \tĐiểm toán \tĐiểm Lý \tĐiểm
Hóa");
    for (int i = 0; i < phongthi.soThisinh(); i++)
    {
        ThiSinh ts=phongthi.layThongtinTS(i);
        System.out.println(ts);
    }
    System.out.println("====================================");
}
```

```
//luu phòng thi vào file
public static void luuFile(){
    try {
        db.LuuFile(fileName, phongthi);
        System.out.println("Lưu file thành công");
    } catch (Exception ex) {
        System.out.println(ex.toString());
    }
}
```

```
//đọc file lưu trở lại phòng thi
public static void docFile(){
    try {
        phongthi=(PhongThi) db.docFile(fileName);
    } catch (Exception ex) {
        System.out.println("không đọc được file");
   }
}
```

```
// hàm main
public static void main(String[] args)  {
    System.out.println("*****QUẢN LÝ PHÒNG THI*****");
    //=================================
    phongthi=new PhongThi("pt01SE","Phòng 502, A9",20);
    //=================================
```

```
    Scanner sc=new Scanner(System.in);
    do {
        menu();
        int tl=sc.nextInt();
        switch(tl) {
        case 1: themTS();break;
        case 2: suaTTTS();break;
        case 3: xoaTS();break;
        case 4: layTTTS_soBD();break;
        case 5: layTTTS_chiso();break;
        case 6: inDanhsachTS();   break;
        case 7: luuFile();   break;
        case 8: docFile();   break;
        case 9: System.out.println("BYE");  System.exit(1);
        }
    }while(true);
}
}//end of class
```