LAB 8: GENERIC

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Sử dụng Enum
- ✓ Sử dụng Boxing/Unboxing
- ✓ Sử dụng Static import
- ✓ Sử dụng Annotation

PHẦN I

BÀI 1 (2 ĐIỂM)

Tạo file **Lab8Bai1.java** sử dụng ArrayList và thực hiện các công việc sau:

- ✓ Thêm vào ArrayList 1 số nguyên
- ✓ Thêm vào ArrayList 1 số thực
- ✓ Thêm vào ArrayList 1 giá trị boolean
- ✓ Thêm vào ArrayList 1 xâu ký tự
- ✓ In ra màn hình 4 giá trị trên từ ArrayList

BÀI 2 (2 ĐIỂM)

Tạo file **Lab8Bai2.java** sử dụng ArrayList<> và thực hiện các công việc sau:

- ✓ Generic ArrayList là kiểu Integer (ArrayList<**Integer**> myarrr = new ArrayList<**Integer**>();)
- ✓ Sử dụng vòng lặp để nhập các số từ 1 đến 10 vào mảng **myarrr** trên
- ✓ Sử dụng vòng lặp để hiển thị các số từ 1 đến 10 từ mảng **myarrr.**

PHẦN II

BÀI 3 (2 ĐIỂM)

1. Tạo lớp mô tả thông tin sản phẩm gồm tên và giá như sau

```
public class Product implements Serializable {
        public String name;
        public Double price;
        public Product(String name, Double price) {
                this.name = name;
                this.price = price;
        }
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public Double getPrice() {
                return price;
        }
        public void setPrice(Double price) {
                this.price = price;
        }
}
```

2. Tạo lớp DAO và khai báo các phương thức thao tác CSDL như sau

```
abstract public class DAO<Entity> {
        protected List<Entity> list = new ArrayList<>();
        public void add(Entity entity){
                list.add(entity);
        }
        public void remove(Entity entity){
                list.remove(entity);
```

```
    }
    abstract public void update(Entity entity);
    abstract public Entity find(Serializable id);
    public List<Entity> getList(){
        return list;
    }
    public void store(String path){
        try {
            FileInputStream fis = new FileInputStream(path);
            ObjectInputStream ois = new ObjectInputStream(fis);
            list = (List<Entity>) ois.readObject();
            ois.close();
            fis.close();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    public void load(String path){
        try {
            FileOutputStream fos = new FileOutputStream(path);
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(list);
            oos.close();
            fos.close();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

BÀI 4 (2 ĐIỂM)

1. Tạo lớp ProductDAO kế thừa từ lớp DAO (ở bài 3) và viết mã thực hiện các phương thức abstract

```
public class ProductDAO extends DAO<Product>{
    @Override
    public void update(Product entity) {
        for(int i=0;i<list.size();i++){
            if(list.get(i).name.equalsIgnoreCase(entity.name)){
                list.set(i, entity);
            }
        }
    }

    @Override
    public Product find(Serializable id) {
        for(Product p : list){
            if(p.name.equals(id)){
                return p;
            }
        }
        return null;
    }
}
```

2. Tạo lớp ProductManager chứa main() thực hiện việc quản lý 2 sản phẩm như sau:

```
public static void main(String[] args) {
    Product p1 = new Product("iPhone9", 1000.0);
    Product p2 = new Product("Samsung Start", 3000.0);

    ProductDAO dao = new ProductDAO();
    dao.add(p1);
    dao.add(p2);
    dao.store("c:/temp/prod.dat");
```

```
        ProductDAO dao2 = new ProductDAO();
        dao2.load("c:/temp/dat");
        Product p = dao2.find("iPhone9");
        System.out.println(">Name: " + p.name);
        System.out.println(">Price: " + p.price);
    }
```

BÀI 5 (2 ĐIỂM)

Giảng viên cho thêm