TÀI LIỆU THỰC TẬP LẬP TRÌNH MẠNG: LAB 02

DANH MỤC THUẬT NGỮ TIẾNG ANH

Từ	Nghĩa của từ
abstract	Trừu tượng
break	Dừng vòng lặp
catch	Từ khóa đầu của một khối bắt ngoại lệ
continue	Bỏ qua phần cuối vòng lặp, tiếp tục sang bước tiếp theo
default	Giá trị mặc định của phương thức switch()
extends	Kế thừa
final	Một hằng số, phương thức hay một lớp không được ghi đè
finally	Một phần của khối xử lý ngoại lệ try luôn được thực hiện
implements	Thực hiện giao diện
import	Khai báo một gói thư viện
instanceof	Kiểm tra một đối tượng là một thể hiện của lớp
interface	Giao diện
new	Tạo một đối tượng mới của lớp
null	Tham chiếu rỗng
package	Gói
private	Tiền tố chỉ được truy cập bởi phương thức của lớp
protected	Tiền tố được truy cập bởi phương thức của lớp, lớp con của và các lớp khác trong cùng một gói
public	Tiền tố có thể được truy cập bởi phương thức của tất cả các lớp
return	Trả về của một phương thức
super	Gọi phương thức của lớp cha
synchronized	Đồng bộ
this	Tham chiếu đến đối tượng hiện tại

DANH MỤC CHỮ VIẾT TẮT

Chữ viết tắt	Ý nghĩa
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
IP	Internet Protocol
URL	*Uniform Resource Locator*
CSDL	Cơ Sở Dữ Liệu
JDBC	Java Database Connectivity
CNTT	Công Nghệ Thông Tin
HĐH	Hệ Điều Hành
MVC	Model-View-Control
DNS	Domain Name System
API	Application Programming Interface
FTP	File Transfer Protocol
JDK	Java Development Kit
GB	GigaByte
UCLN	Ước Chung Lớn Nhất
BCNN	Bội Chung Nhỏ Nhất
RAM	Random Access Memory
RMI	Remote Method Invocation
JVM	Java Virtual Machine
NIC	Network Interface Card
ĐH KTKT CN	Đại học Kinh tế Kỹ thuật Công nghiệp

LỜI NÓI ĐẦU

Ngày nay do nhu cầu thực tế và do sự phát triển mạnh mẽ của nhiều công nghệ tích hợp, dẫn đến các chương trình ứng dụng hầu hết đều có khả năng thực hiện trên môi trường mạng. Ngôn ngữ JAVA là ngôn ngữ phù hợp để viết các ứng dụng mạng. So với lập trình thông thường, lập trình mạng đòi hỏi người lập trình hiểu biết và có kỹ năng tốt để viết các chương trình giao tiếp và trao đổi dữ liệu giữa các máy tính với nhau.
Để hỗ trợ sinh viên chuyên ngành CNTT trong nhà trường tiếp cận với kỹ thuật lập trình mới này, tiếp theo cuốn tài liệu học tập lý thuyết "**Công nghệ JAVA**", chúng tôi xây dựng cuốn "**Bài tập lập trình mạng**", nhằm cung cấp cho sinh viên những kiến thức và kỹ thuật cơ bản nhất để phát triển các chương trình ứng dụng mạng, thông qua các dạng bài tập từ cơ bản đến nâng cao qua các chủ đề: lập trình cơ bản, lập trình hướng đối tượng, lập trình CSDL JDBC, lập trình mạng dùng socket, lập trình phân tán với RMI. Sinh viên sẽ thực hiện các bài thực hành này trên phòng máy nhà trường.

Nội dung cuốn tài liệu bao gồm 12 bài lab chia thành các chủ đề khác nhau. Trong mỗi chủ đề chúng tôi đưa ra tóm tắt lý thuyết, bài tập mẫu, sau đó là bài tập tương tự, và bài tập tổng hợp. Kết quả qua những bài lab, sinh viên được rèn và thành thạo các kỹ năng lập trình hướng đối tượng, lập trình CSDL, lập trình với giao thức truyền thông có sẵn và khả năng tích hợp trong các ứng dụng khác nhau, nhất là các giao thức truyền thông thời gian thực, từ đó sinh viên có thể viết được các phần mềm quản lý theo mô hình MVC, xây dựng được các ứng dụng mạng, các ứng dụng tích hợp và triệu gọi lẫn nhau trên mạng Intranet (mạng cục bộ), mạng Internet (mạng toàn cầu), các hệ thống xử lý truy xuất dữ liệu phân tán hoàn chỉnh. Nội dung biên soạn phù hợp với chuẩn đầu ra của ngành CNTT và ngành mạng máy tính và truyền thông dữ liệu về kỹ năng và kiến thức. Sau khi học xong học phần này sinh viên có thể viết phần mềm quản lý, truyền thông.

Chúng tôi xin chân thành cảm ơn Thầy Nguyễn Hoàng Chiến, phó chủ nhiệm khoa, phụ trách khoa CNTT trường ĐH KTKT CN cùng với các đồng nghiệp đã đóng góp ý kiến cho cuốn tài liệu này. Vì tài liệu được biên soạn lần đầu, chúng tôi đã cố gắng hoàn chỉnh, song không tránh khỏi thiếu sót. Rất mong nhận được sự góp ý của bạn đọc để tài liệu học tập được hoàn thiện hơn.

Xin trân trọng cảm ơn!

Nhóm tác giả

LAB 2. LỚP - ĐỐI TƯỢNG - KẾ THỪA [1, 2, 7]

A. MỤC TIÊU

Cung cấp cho sinh viên các kỹ thuật về lập trình hướng đối tượng trong JAVA:

- Biết cách đặc tả truy xuất các thành phần bên trong lớp.
- Khai thác các tính chất của lập trình hướng đối tượng (kế thừa, nạp chồng, ghi đè phương thức)
- Biết sử dụng mô hình lớp với mô tả kế thừa.
- Biết cách viết code kế thừa và đa hình trên Java.

B. NỘI DUNG

- Khai báo sử dụng lớp, đối tượng.
- Tạo mảng các đối tượng, giải quyết các bài toán quản lý cơ bản.
- Viết các chương trình vận dụng tính kế thừa.

C. YÊU CẦU PHẦN CỨNG, PHẦN MỀM

- Máy tính cài HĐH windows, RAM tối thiểu 1GB.
- Phần mềm NETBEAN IDE 8.0, JDK 1.8

D. KẾT QUẢ SAU KHI HOÀN THÀNH

- Xây dựng được ứng dụng với nhiều lớp được tổ chức theo sự phân cấp kế thừa
- Sử dụng lại những gì đã có ở một lớp cha
- Ghi đè hiệu chỉnh lại nội dung của một phương thức ở lớp con.

E. HƯỚNG DẪN CHI TIẾT

1. Lớp

- Tên lớp: Viết hoa
- Danh sách các thuộc tính (khai báo là **private**)
- Phương thức khởi tạo: Để khởi tạo đối tượng (phương thức tạo 0, 1, 2… tham số)
- Danh sách các phương thức được cài đặt thêm

Mô tả lớp:

```
package  pack.name;
Modifier class ClassName{
      class's body
}
```

Khai báo mảng đối tượng:

```
ClassName [ ]  ob = new ClassName [size]
```

Bài 1.

Tạo **class** Product gồm các thuộc tính:

- Tên hàng hóa
- Nhà sản xuất

- Giá bán

+ Tạo 2 constructor cho lớp này.

+ Cài đặt phương thức nhập và hiển thị.

Tạo **class** ProductMenu, khai báo phương thức main và tạo menu sau:

1. Nhập thông tin cho n sản phẩm
2. Hiển thị thông tin vừa nhập
3. Sắp xếp thông tin giảm dần theo giá và hiển thị
4. Thoát

Hướng dẫn:

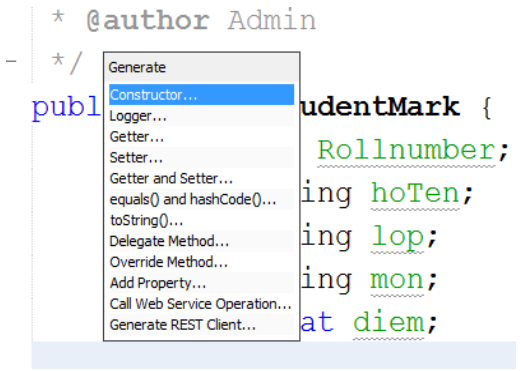
Bước 1: Tạo lớp Product gồm các thuộc tính.

private String tenHangHoa;

private String nhaSanXuat;

private float giaBan;

Chèn tự động phương thức tạo, phương thức set, get. Sinh viên có thể dùng tổ hợp phím Alt+Insert để chèn code tự động.



Hình 1. Chèn code tự động

Viết phương thức nhập và phương thức xuất thông tin là các thuộc tính của lớp.

Bổ sung phương thức sắp xếp:

```
public void sort(Product[] b){
         for (int i = 0; i < b.length - 1; i++) {
            for (int j = i + 1; j < b.length; j++) {
                if (b[i].giaBan > b[j].giaBan) {
                    Product tem = b[i];
                    b[i] = b[j];
                    b[j] = tem;
                }
            }
      }
}
```

Bước 2: Tạo lớp ProductMenu có nội dung như sau.

```
public class ProductMenu {
    static void menu() {
       Viết nội dung lựa chọn dùng switch tương ứng với đề bài
    }
    public static void main(String[] args) {
        int n = 0;
        Product a = new Product();
        Product[] product = null;
        do {
            menu();
            System.out.println("Nhap vao lua chon cua ban :");
            Scanner sc = new Scanner(System.in);
            n = Integer.parseInt(sc.nextLine());
            switch (n) {
                case 1: {
                    int m;
                    System.out.println("Nhap vao n :");
                    m = Integer.parseInt(sc.nextLine());
                    product = new Product[m];
                    for (int i = 0; i < m; i++) {
                        product[i] = new Product();
                        product[i].nhap();
                    }
                    break;
                }
                case 2: {
                    if (product == null) {
                        System.out.println("ban chua nhap du lieu");
                    } else {
                        System.out.println("du lieu ban vua nhap la :");
                        for (int i = 0; i < product.length; i++) {
                 System.out.println("thong tin hang hoa thu " + (i + 1));
                            product[i].hienthi();
                        }
                    }
                    break;
                }
                case 3: {
                    if (product == null) {
                        System.out.println("ban chua nhap du lieu");
```

```
                    } else {
                        a.sort(product);
                     System.out.println("du lieu sau khi sap xep  la :");
                        for (int i = 0; i < product.length; i++) {
                 System.out.println("thong tin hang hoa thu " + (i + 1));
                            product[i].hienthi();
                        }
                    }
                    break;
                }
                case 4:
                    break;
                default: {
                    System.out.println("khong co lua chon cua ban ");
                    break;
                }
            }
        } while (n != 4);
    }
}
```

Bài 2.

Cài đặt lớp Product gồm các thuộc tính (khai báo là **private**)

- **String** maHH;
- **String** tenHH;
- **float** soLuong;
- **float** gia1SP;

Cài đặt 2 construcors, các phương thức get/set.

Cài đặt phương thức input(), display().

Khai báo phương thức main và thực hiện như sau:

- Khai báo mảng có n phần tử kiểu Product.
- Gọi phương thức nhập thông tin cho các phần tử của mảng.
- Tìm ra sản phẩm nào có giá bán cao nhất.
- Sắp xếp theo thứ tự giảm dần của giá
- Tìm trong danh sách hàng hóa có mặt hàng "Sữa" hay không?

Hướng dẫn:

Bước 1: Tương tự , xây dựng lớp ProductBai2 gồm các thuộc tính, phương thức tạo, Phương thức set, get.

Bước 2: Xây dựng lớp TestProduct có phương thức main và một số code như sau:

Tìm mặt hàng có giá cao nhất.

```
ArrayList<ProductBai2> arrlist = new ArrayList<ProductBai2>();
float max = 0;
for (ProducBai2 pr : arrlist) {
      if (max < pr.getGia1SP()) {
            max = pr.getGia1SP();
      }
}
System.out.println("thong tin mat hang co gia cao nhat la :");
for (ProductBai2 pr : arrlist) {
      if (pr.getGia1SP() == max) {
            pr.hienthi();
      }
}
```

Sắp xếp theo giá: Sử dụng Collections.sort

```
Collections.sort(arrlist, new Comparator<ProductBai2>() {
      @Override
      public int compare(ProductB2 pr1, ProductBai2 pr2) {
            if (pr1.getGia1SP() < pr2.getGia1SP()) {
                  return 1;
            } else {
                  if (pr1.getGia1SP() == pr2.getGia1SP()) {
                  return 0;
            } else {
                  return -1;
            }
        }
   }
);
System.out.println("danh sach duoc sap xep giam dan theo gia la:");
int i=1;
for (ProductBai2 pr : arrlist) {
      System.out.println("san pham thu :"+(i));
      pr.hienthi();
      i++;
}
```

Tìm kiếm mặt hàng tên là "Sữa"

```
for (ProductBai2 pr : arrlist)
      {
      if (pr.getTenHH().equals("sữa") || pr.getTenHH().equals("SỮA"))
```

```
      pr.hienthi();
     }
```

2. Kế thừa

- Lớp mới kế thừa những thành viên đã có trong lớp cũ
- Một lớp chỉ có thể kế thừa 1 lớp khác nhưng có thể thực thi nhiều giao diện.
- **extends** với lớp và **implements** với giao diện

Cú pháp

```
class Subclass extends Superclass{
//Subclass body
}
```

Lớp con có thể kế thừa được gì?

- Kế thừa được các thành viên (bao gồm dữ liệu và phương thức) được khai báo là **public** và **protected** của lớp cha.
- Không kế thừa được các thành viên **private**.

Khởi tạo đối tượng trong kế thừa

- Lớp con không kế thừa phương thức khởi tạo của lớp cha
- Lớp cha phải được khởi tạo trước lớp con
- Các phương thức khởi tạo của lớp con luôn gọi phương thức khởi tạo của lớp cha:

  + Tự động gọi (không tường minh – không cần thể hiện bằng câu lệnh gọi): nếu lớp cha có phương thức khởi tạo **mặc định**

  + Gọi trực tiếp (tường minh): nếu lớp cha có phương thức khởi tạo **khác mặc định**. Cú pháp: **super(**parameter List)

Bài 3. Cài đặt lớp Book gồm các thuộc tính

```
private String bookName;
private String bookAuthor;
private String producer;
private int yearPublishing;
private float price;
```

Cài đặt 2 constructors, các phương thức set/get cho các thuộc tính của lớp.

Cài đặt 2 phương thức input() và display để nhập và hiển thị các thuộc tính của lớp.

Cài đặt lớp UnetiBook kế thừa lớp Book và bổ sung thêm vào thuộc tính:

```
private String language;
private int semester;
```

Cài đặt 2 constructor trong đó sử dụng **super** để gọi đến constructor của lớp cha.

Cài đặt các phương thức get/set cho các thuộc tính bổ sung

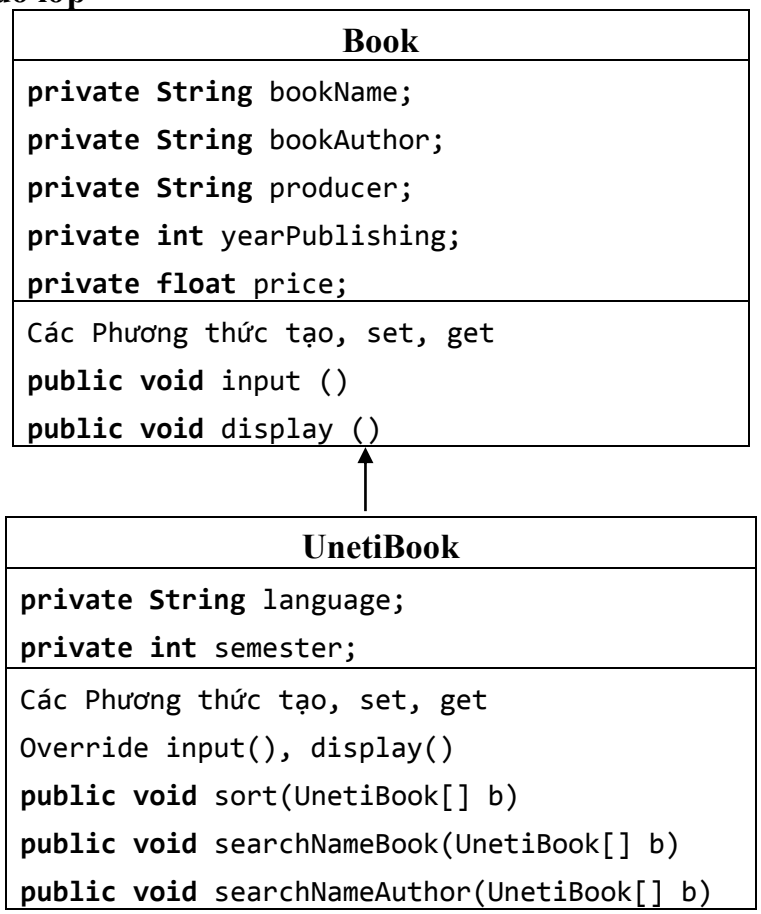
Override các phương thức input() và display().

Cài đặt lớp Test trong đó tạo menu và thực hiện theo các chức năng của menu:

1. Nhập thông tin n cuốn sách của Uneti
2. Hiển thị thông tin vừa nhập

3. Sắp xếp thông tin giảm dần theo năm xuất bản và hiển thị
4. Tìm kiếm theo tên sách
5. Tìm kiếm theo tên tác giả
6. Thoát.

Hướng dẫn: Sơ đồ lớp



Bước 1: Tạo lớp Book gồm các thuộc tính

```
public class Book {
      private String bookName;
      private String bookAuthor;
      private String producer;
      private int yearPublishing;
      private float price;
Sau đó chèn các phương thức tạo, set, get, phương thức input(), display()
}
```

Bước 2: Tạo lớp UnetiBook kế thừa từ lớp Book

```
public class UnetiBook extends Book{
      private String language; //Hai thuộc tính riêng của lớp UnetiBook
      private int semester;
Bổ sung phương thức tạo, phương thức set, get cho các thuộc tính tương ứng của class này
}
```

Sau đó Override hai phương thức input () và display () của lớp Book như sau:

```
@Override
     public void input(){
         super.input();
         Scanner sc=new Scanner(System.in);
         System.out.println("nhap ngon ngu :");
         language=sc.nextLine();
         System.out.println("Nhap hoc ki :");
         semester =Integer.parseInt(sc.nextLine());
     }
   @Override
     public void display(){
         super.display();
         System.out.println("ngon ngu :"+language);
         System.out.println("hoc ki :"+semester);
     }
```

Viết phương thức sắp xếp sách theo năm xuất bản

```
public void sort(UnetiBook[] b){
         for (int i = 0; i < b.length - 1; i++) {
            for (int j = i + 1; j < b.length; j++) {
                if (b[i].getYearPublishing() < b[j].getYearPublishing())
                 {
                    UnetiBook tem = b[i];
                    b[i] = b[j];
                    b[j] = tem;
                 }
            }
        }
    }
```

Phương thức tìm kiếm sách theo tên sách

```
public void searchNameBook(UnetiBook[] b){
      Scanner sc=new Scanner(System.in);
      System.out.println("nhap ten sach:");
      String nameBook=sc.nextLine();
      int dem=0;
      System.out.println("thong tin sach ban muon tim la :");
      for (int i = 0; i < b.length; i++) {
            if(b[i].getBookName().equals(nameBook)){
                  b[i].hienthi();
```

```
                dem++;
            }
      }
      if(dem==0){
            System.out.println("khong co sach ban muon tim");
      }
   }
```

Phương thức tìm kiếm theo tên tác giả

```
public void searchNameAuthor(UnetiBook[] b){
  Viết tương tự Phương thức tìm kiếm theo tên sách.
   Sử dụng for để duyệt mảng.
 Sử dụng if(b[i].getBookName().equals(nameAuthor)) kiểm tra tên tác giả
            }
```

Phương thức tạo menu() để sử dụng

```
void menu(){
         System.out.println("1 nhap thong tin n cuon sach ");
         System.out.println("2 hien thi thong tin vua nhap");
         System.out.println("3 sap xep giam dan theo nam xuat ban");
         System.out.println("4 tim kiem theo ten sach");
         System.out.println("5 tim kiem theo ten tac gia ");
         System.out.println("6 thoat");
     }
```

Phương thức main() để chạy chương trình

```
public static void main(String[] args) {
        int n = 0;
        UnetiBook ab=new UnetiBook();
        UnetiBook[] ab1=null;
        do {
           ab.menu();
            System.out.println("Nhap vao lua chon cua ban :");
            Scanner sc = new Scanner(System.in);
            n = Integer.parseInt(sc.nextLine());
            switch (n) {
                case 1: {
                    int m;
                    System.out.println("Nhap vao n :");
                    m = Integer.parseInt(sc.nextLine());
                    ab1= new UnetiBook[m];
                    for (int i = 0; i < m; i++) {
```

```
                    ab1[i] = new UnetiBook();
                    ab1[i].input();
                }
                break;
            }
            case 2: {
                if (ab1 == null) {
                    System.out.println("ban chua nhap du lieu");
                } else {
                    System.out.println("du lieu ban vua nhap la :");
                    for (int i = 0; i < ab1.length; i++) {
                System.out.println("thong tin sach thu " + (i + 1));
                        ab1[i].hienthi();
                    }
                }
                break;
            }
            case 3: {
                if (ab1 == null) {
                    System.out.println("ban chua nhap du lieu");
                } else {
                    ab.sort(ab1);
          System.out.println("du lieu sau khi sap xep  la :");
                    for (int i = 0; i < ab1.length; i++) {
            System.out.println("thong tin hang hoa thu " + (i + 1));
                        ab1[i].hienthi();
                    }
                }
                break;
            }
            case 4:
            {
                ab.searchNameBook(ab1);
                break;
             }
            case 5:
            {
                ab.searchNameBook(ab1);
                break;
            }
```

```
                case 6:
                    break;
                default:{
                    System.out.println("khong co lua chon cua ban ");
                    break;
                }
            }
        } while (n != 6);
    }
}
```

Bài 4.

Tạo **class** Engine gồm các thuộc tính:

- engineId (Mã máy)
- engineName (Tên máy)
- manufacturer (Tên nhà sản xuất)
- yearMaking (Năm sản xuất)
- price (Giá bán)

+ Tạo 2 constructors

+ Tạo các phương thức get/set

+ Cài đặt phương thức input(), display()

Tạo **class** Mobile kế thừa lớp Engine ở trên và bổ sung thêm các thuộc tính:

- country (Nước sản xuất)

+Tạo 2 constructors, trong đó constructor có tham số phải sử dụng từ khóa **super** để gọi đến constructor có tham số của lớp cha

+ Cài đặt các phương thức get/set cho thuộc tính bổ sung

+ Override phương thức input() và display() của lớp cha.

Tạo **class** Car kế thừa **class** Engine và bổ sung thêm thuộc tính:

- totalSeat (Số chỗ ngồi)
- speed (Tốc độ)

+ Tạo 2 constructors, trong đó constructor có tham số phải sử dụng từ khóa **super** để gọi đến constructor có tham số của lớp cha

+ Cài đặt các phương thức get/set cho thuộc tính bổ sung

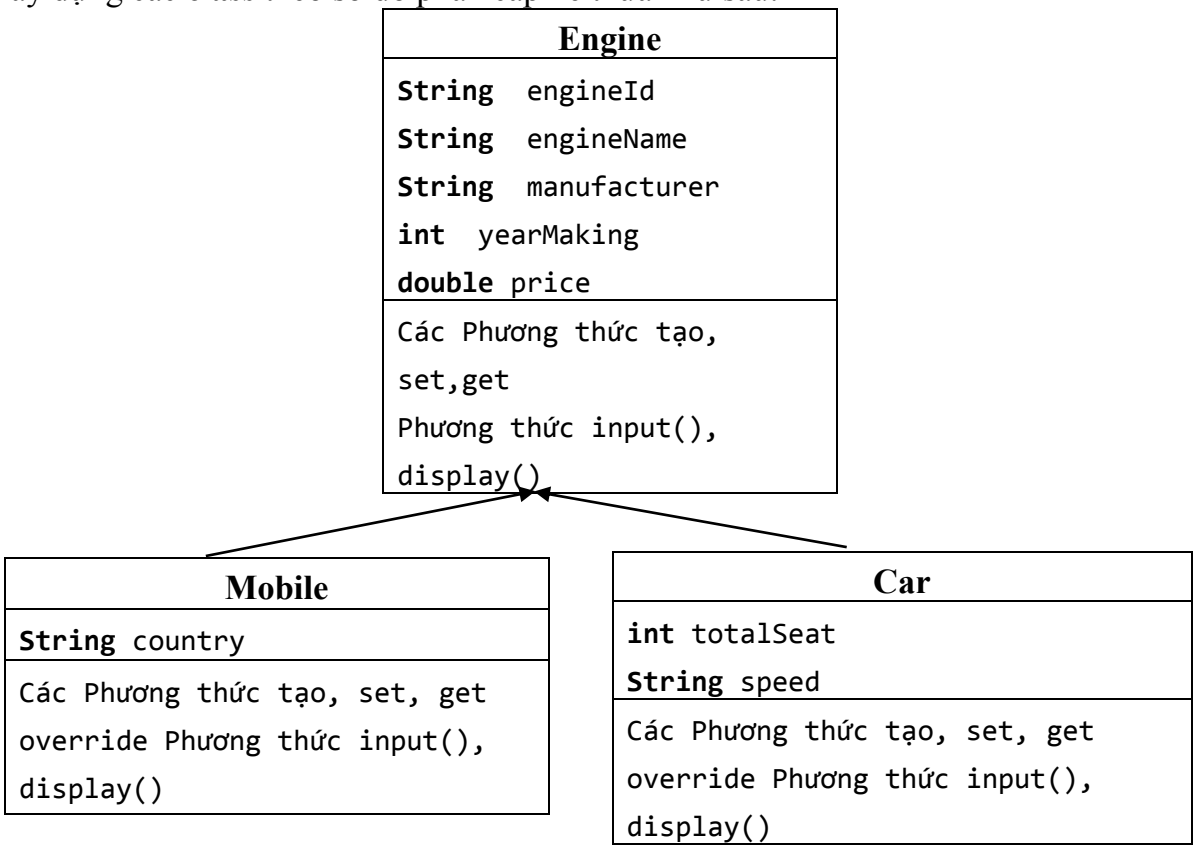
+ Override phương thức input() và display() của lớp cha.

Tạo **class** Manager trong đó có phương thức main.

1. Nhập vào thông tin cho n điện thoại
2. Nhập vào thông tin cho n ô tô
3. Hiển thị thông tin cả điện thoại và ô tô
4. Tìm kiếm thông tin theo tên nhà sản xuất.
5. Thoát

Hướng dẫn:

Xây dựng các **class** theo sơ đồ phân cấp kế thừa như sau:



Bài 5.

Một thư viện cần quản lý các tài liệu bao gồm Sách, Tạp chí, Báo

+ Mỗi tài liệu có các thuộc tính: Mã tài liệu, Tên nhà xuất bản, Số bản phát hành.
+ Các loại sách cần quản lý: Tên tác giả, Tên sách, số trang
+ Các tạp chí cần quản lý: Số phát hành, tháng phát hành
+ Các báo cần quản lý: ngày phát hành. (Date)

Xây dựng các lớp quản lý các loại tài liệu trên sao cho việc sử dụng lại được nhiều nhất.

Xây dựng lớp QuanLySach cài đặt các phương thức thực hiện các công việc sau:

a. Nhập thông tin về các tài liệu
b. Hiển thị thông tin về các tài liệu
c. Tìm kiếm tài liệu theo loại
d. Tìm kiếm tài liệu theo tên tác giả
e. Tìm kiếm theo tên tác giả "gần đúng" .
f. Hiển thị danh sách về loại tài liệu theo chỉ định của người sử dụng.

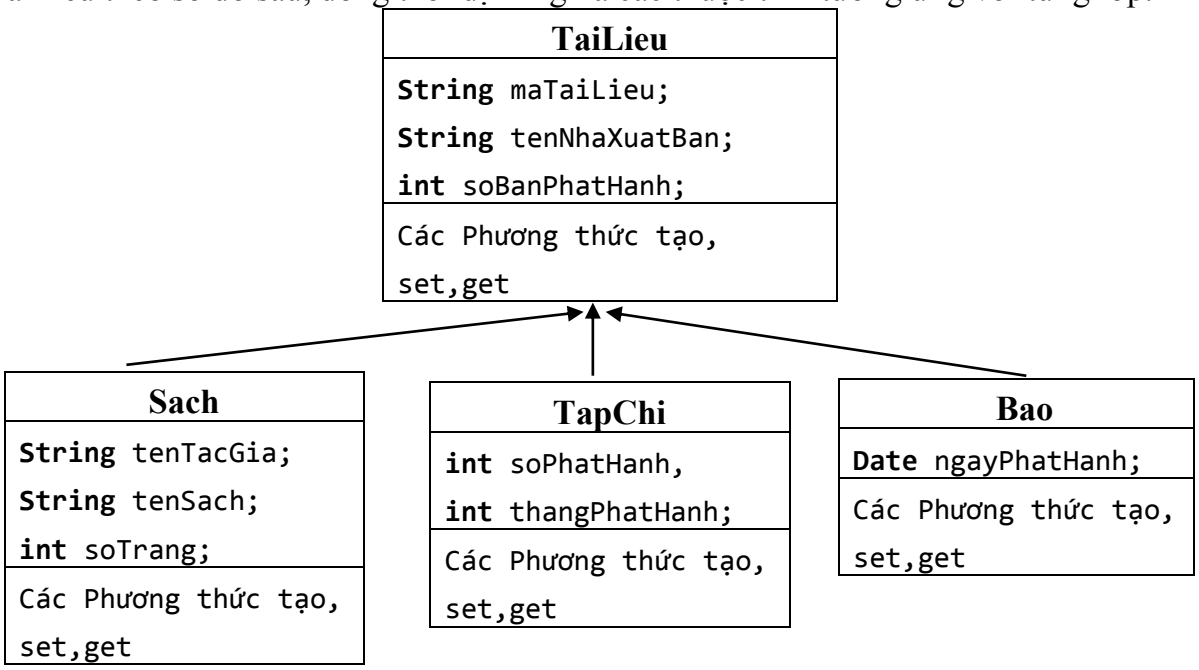
Hướng dẫn:

Bước 1: Xây dựng class TaiLieu

```
class TaiLieu
{
	String maTaiLieu;
	String tenNhaXuatBan;
```

```
      int soBanPhatHanh;
      Chèn tự động các Phương thức khởi tạo, set, get thuộc tính.
}
```

Sau khi xây dựng lớp TaiLieu, lần lượt tạo các lớp Sach, TapChi, Bao kế thừa lớp TaiLieu theo sơ đồ sau, đồng thời định nghĩa các thuộc tính tương ứng với từng lớp:



Bước 2: Định nghĩa lớp Sach kế thừa lớp TaiLieu

```
public class Sach extends TaiLieu{
    private String tenTacGia;
    private String tenSach;
    private int soTrang;
Chèn tự động Phương thức tạo, các Phương thức set, get
   }
```

Bước 3: Tượng tự tạo lớp TapChi kế thừa lớp TaiLieu

```
public class TapChi extends TaiLieu{
    private int soPhatHanh,thangPhatHanh;
   Chèn tự động các Phương thức tạo, set, get
 }
```

Bước 4: Lớp Bao kế thừa lớp TaiLieu

```
public class Bao extends TaiLieu{
    private Date ngayPhatHanh;
   Chèn tự động Phương thức tạo, các Phương thức set, get tự động
    }
```

Bước 5: Tạo lớp QuanLyTV để định nghĩa các phương thức nhằm cung cấp các chức năng cho hệ thống như sau:

```
public class QuanLyTV {
      private ArrayList<TaiLieu> taiLieus;
```

```
    private Scanner reader;
    public QuanLyTV() {
      taiLieus = new ArrayList<>();
      reader = new Scanner(System.in);
    }
    public Sach taoMoiSach(){
          Sach s = new Sach();
          System.out.println("Mã tài liệu: ");
          s.setMaTaiLieu(reader.nextLine());
          System.out.println("Tên nhà xuất bản: ");
          s.setTenNhaXuatBan(reader.nextLine());
          System.out.println("Số bản phát hành: ");
          s.setSoBanPhatHanh(Integer.parseInt(reader.nextLine()));
          System.out.println("Tên tác giả: ");
          s.setTenTacGia(reader.nextLine());
          System.out.println("Tên sách: ");
          s.setTenSach(reader.nextLine());
          System.out.println("Số trang: ");
          s.setSoTrang(Integer.parseInt(reader.nextLine()));
          return s;
    }
    public TapChi taoMoiTapChi(){
          TapChi tapChi = new TapChi();
          System.out.println("Mã tài liệu: ");
          tapChi.setMaTaiLieu(reader.nextLine());
          System.out.println("Tên nhà xuất bản: ");
          tapChi.setTenNhaXuatBan(reader.nextLine());
          System.out.println("Số bản phát hành: ");
          tapChi.setSoBanPhatHanh(Integer.parseInt(reader.nextLine()));
          System.out.println("Số phát hành: ");
          tapChi.setSoPhatHanh(Integer.parseInt(reader.nextLine()));
          System.out.println("Tháng phát hành: ");
          tapChi.setThangPhatHanh(Integer.parseInt(reader.nextLine()));
          return tapChi;
    }
    public Bao taoMoiBao(){
          Bao bao = new Bao();
          System.out.println("Mã tài liệu: ");
          bao.setMaTaiLieu(reader.nextLine());
          System.out.println("Tên nhà xuất bản: ");
```

```
            bao.setTenNhaXuatBan(reader.nextLine());
            System.out.println("Số bản phát hành: ");
            bao.setSoBanPhatHanh(Integer.parseInt(reader.nextLine()));
            System.out.println("Ngày phát hành: ");
            bao.setNgayPhatHanh(convertStringToDate(reader.nextLine()));
            return bao;
      }
      private Date convertStringToDate(String ddMMyyyy) {
            try {
            return new SimpleDateFormat("dd/MM/yyyy").parse(ddMMyyyy);
            } catch (ParseException ex) {
Logger.getLogger(QuanLyTV.class.getName()).log(Level.SEVERE, null, ex);
            }
            return null;
      }
      public void nhapDanhSachTaiLieu(){
            System.out.println("Nhập 1 - Tạo mới sách");
            System.out.println("Nhập 2 - Tạo mới tạp chí");
            System.out.println("Nhập 3 - Tạo mới báo");
            System.out.println("Nhập 4 - Kết thúc");
            int selectedValue;
            do {
                  System.out.println("Bạn chọn: ");
                  selectedValue = Integer.parseInt(reader.nextLine());
                  switch(selectedValue){
                        case 1:
                              taiLieus.add(taoMoiSach());
                              break;
                        case 2:
                              taiLieus.add(taoMoiTapChi());
                              break;
                        case 3:
                              taiLieus.add(taoMoiBao());
                              break;
                  }
            } while (selectedValue!=4);
      }
      private void xuatThongTin(TaiLieu taiLieu){
       System.out.println("Mã tài liệu: " + taiLieu.getMaTaiLieu());
 System.out.println("Tên nhà xuất bản: " taiLieu.getTenNhaXuatBan());
```

```
  System.out.println("Số bản phát hành:" + taiLieu.getSoBanPhatHanh());
            if (taiLieu instanceof Sach) {
                  Sach sach = (Sach) taiLieu;
               System.out.println("Tên tác giả: " + sach.getTenTacGia());
               System.out.println("Tên sách: " + sach.getTenSach());
               System.out.println("Số trang: " + sach.getSoTrang());
            }
            else{
                  if (taiLieu instanceof TapChi) {
                        TapChi tapChi = (TapChi) taiLieu;
 System.out.println("Số phát hành: " + tapChi.getSoPhatHanh());
System.out.println("Tháng phát hành:" + tapChi.getThangPhatHanh());
                  }else{
                 Bao bao = (Bao)taiLieu;
System.out.println("Ngày phát hành"+
convertDateToString(bao.getNgayPhatHanh()));
                  }
            }
      }
   private String convertDateToString(Date ngayPhatHanh) {
      return new SimpleDateFormat("dd/MM/yyyy").format(ngayPhatHanh);
      }
   public void xuatDanhSachTaiLieu(){
            for (TaiLieu taiLieu : taiLieus) {
                  xuatThongTin(taiLieu);
            }
      }
   public void timTheoLoai(String loai){
            if (loai.equalsIgnoreCase("Sach")) {
                  for (TaiLieu taiLieu : taiLieus) {
                        if (taiLieu instanceof Sach) {
                              xuatThongTin(taiLieu);
                        }
                  }
            } else {
                  if (loai.equalsIgnoreCase("Tap Chi")) {
                        for (TaiLieu taiLieu : taiLieus) {
                              if (taiLieu instanceof TapChi) {
                                    xuatThongTin(taiLieu);
                              }
```

```
                        }
                  } else {
                        for (TaiLieu taiLieu : taiLieus) {
                              if (taiLieu instanceof Bao) {
                                    xuatThongTin(taiLieu);
                              }
                        }
                  }
            }
      }
   public void timGanDungTheoTenSach(String str){
            for (TaiLieu taiLieu : taiLieus) {
                  if (taiLieu instanceof Sach) {
                        Sach sach = (Sach) taiLieu;
                        if (sach.getTenSach().indexOf(str)!=-1) {
                              xuatThongTin(taiLieu);
                        }
                  }
            }
      }
   public static void main(String[] args) {
            Scanner input = new Scanner(System.in);
            QuanLyTV QuanLyTV = new QuanLyTV();
            QuanLyTV.nhapDanhSachTaiLieu();
            QuanLyTV.xuatDanhSachTaiLieu();
            System.out.println("Nhập loại bạn muốn tìm: ");
            QuanLyTV.timTheoLoai(input.nextLine());
      }
}
```

Bài 6. Xây dựng chương trình quản lý danh sách hoá đơn tiền điện của khách hàng. Thông tin bao gồm các loại khách hàng :

- Khách hàng Việt Nam: mã khách hàng, họ tên, ngày ra hoá đơn (ngày, tháng, năm), đối tượng khách hàng (sinh hoạt, kinh doanh, sản xuất): số lượng (số KW tiêu thụ), đơn giá, định mức. Thành tiền được tính như sau:

- Nếu số lượng <= định mức thì: thành tiền = số lượng * đơn giá.

- Ngược lại thì: thành tiền = số lượng * đơn giá * định mức + số lượng KW vượt định mức * Đơn giá * 2.5.

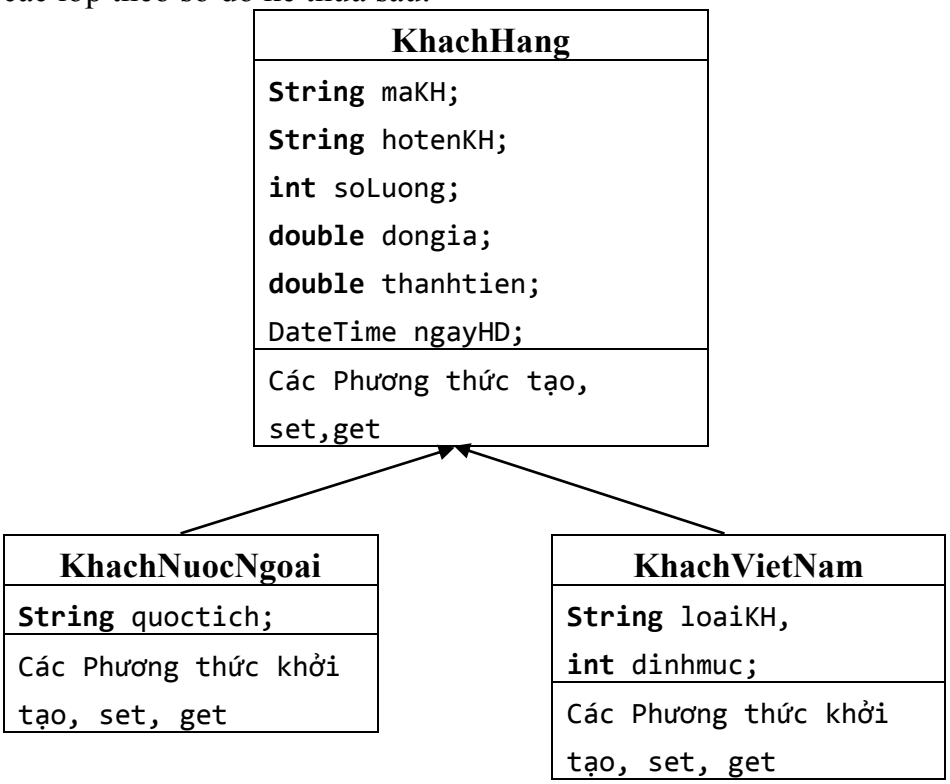
- Khách hàng nước ngoài: mã khách hàng, họ tên, ngày ra hoá đơn (ngày, tháng, năm), quốc tịch, số lượng, đơn giá. Thành tiền = số lượng * đơn giá.

Thực hiện các yêu cầu sau:

a. Xây dựng các lớp tương ứng với sơ đồ kế thừa.
b. Nhập xuất danh sách các hóa đơn khách hàng.
c. Tính tổng số lượng điện tiêu thụ cho từng loại khách hàng.
d. Tính trung bình thành tiền của khách hàng người nước ngoài.
e. Xuất ra các hoá đơn trong tháng 09 năm 2013 (của 2 loại khách hàng).

Hướng dẫn:

Xây dựng các lớp theo sơ đồ kế thừa sau:



Bước 1. Xây dựng lớp Khách hàng bao gồm các thuộc tính chung cho cả Khách hàng nước ngoài và Khách hàng Việt Nam. Gồm các thuộc tính: mã khách hàng, số lượng, đơn giá, thành tiền, ngày của hóa đơn và họ tên khách hàng.

Bước 2. Xây dựng lớp Khách hàng nước ngoài thừa kế lớp Khách hàng bao gồm thuộc tính: *quốc tịch.*

Bước 3. Xây dựng lớp Khách hàng Việt Nam thừa kế lớp Khách hàng bao gồm thuộc tính: loại khách hàng, định mức.

Bước 4. Xây dựng lớp quản lý danh sách các khách hàng (dùng cấu trúc mảng)

Bước 5. Xây dựng lớp quản lý thông tin cho phép nhập xuất, tính trung bình thành tiền.

Bài 7. Tổng hợp

Giả sử cần xây dựng chương trình quản lý dùng cho một học viện nghiên cứu giảng dạy và ứng dụng. Đối tượng quản lý bao gồm các sinh viên đang theo học, các nhân viên đang làm việc tại học viện, các khách hàng đến giao dịch mua bán sản phẩm ứng dụng.

Dựa vào một số đặc tính của từng đối tượng, người quản lý cần đưa ra cách thức đánh giá khác nhau. Xây dựng các lớp sau:

- *Lớp **Person**: bao gồm các thuộc tính họ tên, địa chỉ, phương thức* toString.

- Các *lớp **Student, Employee, Customer*** (mô tả dưới đây) thừa kế lớp ***Person***.

o Lớp ***Student:*** bao gồm các thuộc tính điểm môn học 1, điểm môn học 2, và các phương thức: tính điểm TB, đánh giá, overriding phương thức toString trả về bảng điểm sinh viên (gồm thông tin thuộc tính và điểm TB).

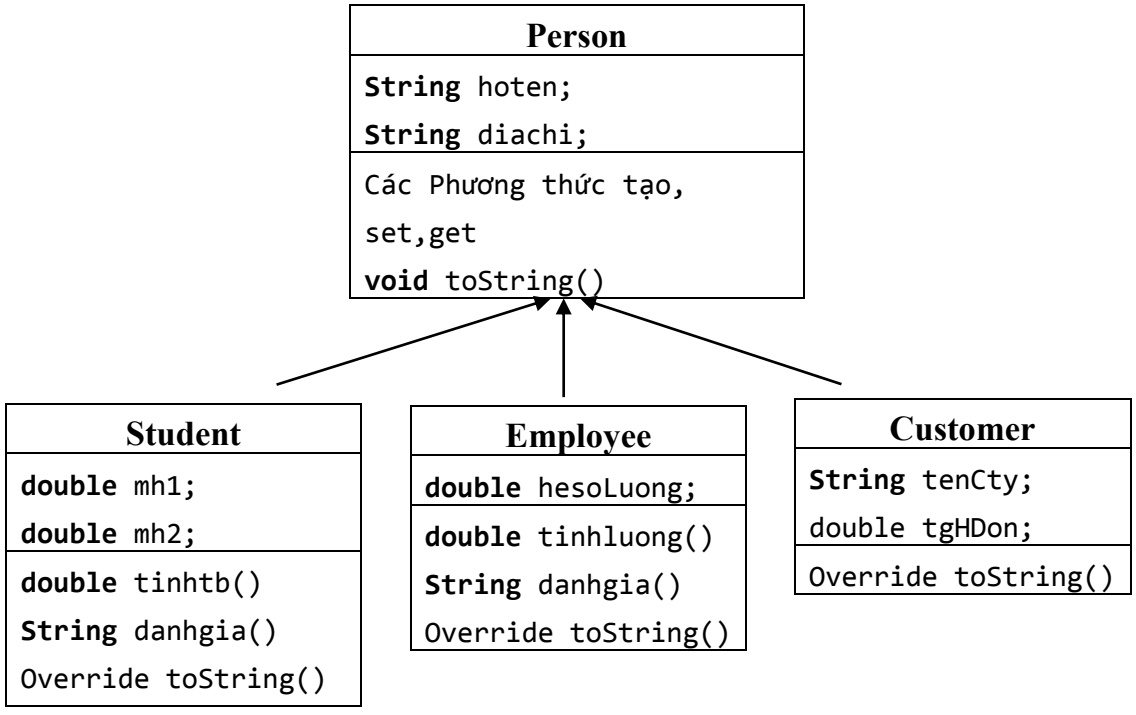
o Lớp ***Employee:*** bao gồm thuộc tính *heSoLương,* và các phương thức: tính lương, đánh giá, overriding phương thức toString trả về bảng lương cho nhân viên (gồm thông tin thuộc tính đối tượng và tiền lương).

o Lớp ***Customer:*** bao gồm thuộc tính tên công ty, trị giá hóa đơn, phương thức toString trả về thông tin hóa đơn cho khách hàng (gồm các thuộc tính của đối tượng).

- Lớp có 1 *biến* danh sách để lưu các sinh viên, nhân viên, khách hàng (dùng 1 biến array Person), biến lưu tổng số người có trong danh sách, constructor mặc định khởi tạo array với dung lượng cho trước, phương thức thêm một người vào danh sách (thông số Person), xóa 1 người khỏi danh sách (nhận thông số là họ tên của người cần xóa), sắp xếp danh sách theo thứ tự họ tên, phương thức xuất danh sách. Khi danh sách đầy thì tự động tăng dung lượng dãy lên 50%.

- Viết lớp với phương thức main cho phần kiểm nghiệm. Giao tiếp với người dùng bằng menu *(thể hiện tính đa hình – polymorphism bằng cách cho phép lựa chọn nhập thông tin là sinh viên, nhân viên hay khách hàng).*

Hướng dẫn: Xây dựng các lớp theo sơ đồ kế thừa sau:



TÀI LIỆU THAM KHẢO

[1]. Cay S. Horstmann, *Core Java Volum I - Fundamentals, Tenth Edition*, NewYork : Prentice Hall, 2016.

[2]. Cay S. Horstmann. *Core Java Volum II - Advanced Features, Tenth Edition*, New York : Prentice Hall, 2017.

[3].Eng.haneen Ei-masry, *Java database connection*, Islamic University of Gaza Faculty of Engineering Department of Computer Engineering ECOM 4113: DataBase Lab, 2014.

[4]. Angelos Stavrou, *Advanced Network Programming Lab using Java*, Network Security, ISA 656, Angelos Stavrou.

[5]. Marenglen Biba, Ph.D, *Manual for Lab practices, Remote Method Invocation Three Tier Application with a Database Server*, Department of Comsputer Science, University of New York.

[6].Elliotte Rusty Harold, *Java Network Programming, Fourth Edition*, O'Reilly Media, 2013.

[7]. Đoàn Văn Ban, *Lập trình hướng đối tượng với JAVA*, Nhà xuất bản Khoa học và Kỹ thuật, 2005.

[8]. ThS. Dương Thành Phết, *Bài tập thực hành Chuyên đề 1 CNPM- Java*, Khoa CNTT- Trường ĐH Công nghệ TP.HCM.

[9]. https://www.oracle.com/technetwork/java/socket-140484.html#

[10]. https://personales.unican.es/corcuerp/java/Labs/LAB_22.htm

[11]. http://www.nrcmec.org/pdf/Manuals/CSE/student/2-2%20java16-17.pdf

[12]. http://cse.mait.ac.in/pdf/LAB%20MANUAL/JAVA.pdf

[13]. https://www.academia.edu/35283541/Bài tập môn lập trình hướng đối tượng