

PHỤ LỤC: HƯỚNG DẪN BÀI TẬP

TUẦN 1: LÀM QUEN NGÔN NGỮ JAVA

Nội dung bài tập cơ bản:

Bài 1: Viết chương trình xuất ra màn hình các thông tin sau.

“Hello! I’m <your name>.”

“I’m <your age> years old”.

“This is my first java program.”

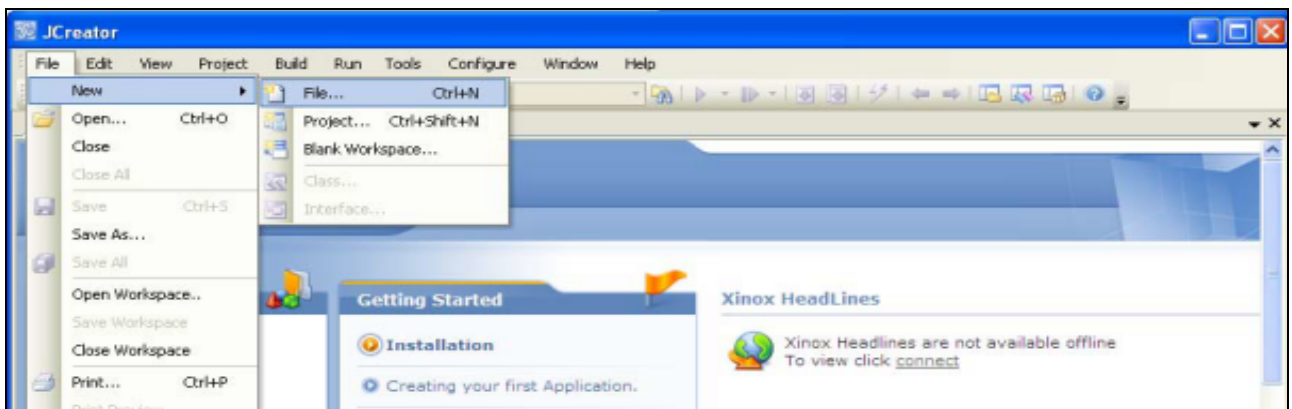
Bài 2: Nhập các thông tin của 1 sinh viên gồm mã sinh viên, họ tên, tuổi, năm sinh, điểm trung bình. Xuất các thông tin ra màn hình.

Bài 3: Nhập 1 mảng các số nguyên từ bàn phím. Tìm phần tử lớn nhất trong mảng.

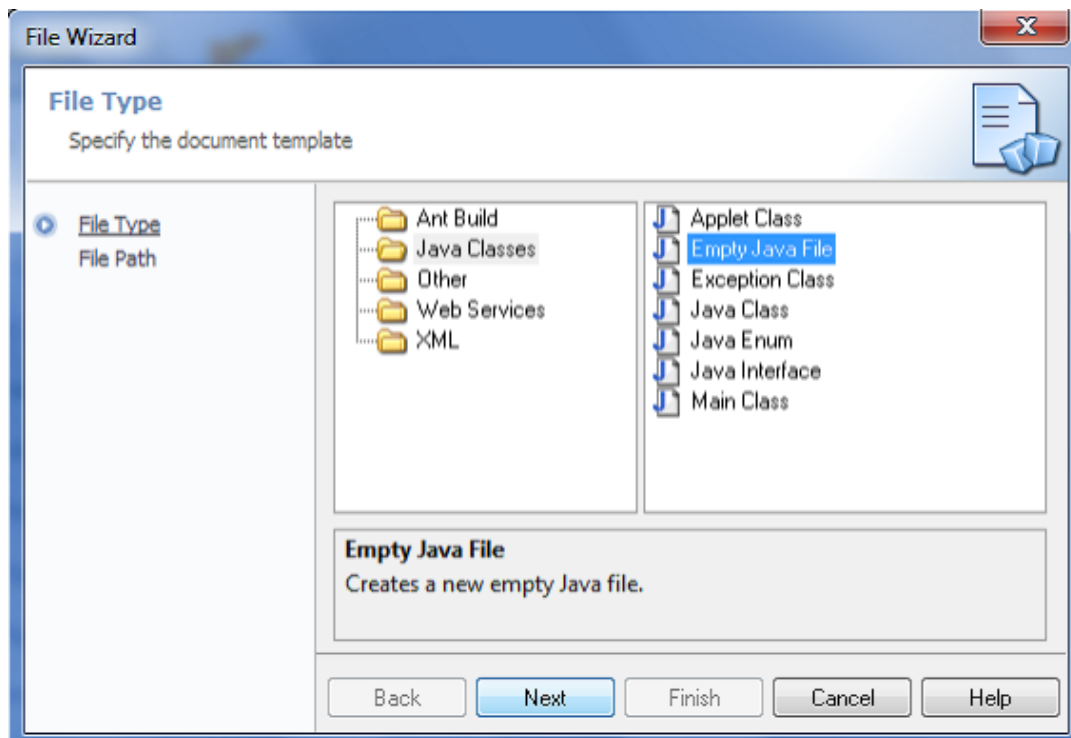
Bài 4: Tạo một ma trận gồm m dòng và n cột, trong đó mỗi phần tử của ma trận là một giá trị nguyên được sinh ngẫu nhiên trong phạm vi [0, 50]. Xuất ma trận ra màn hình. Tính tổng giá trị các phần tử của ma trận.

Hướng dẫn làm quen với môi trường lập trình Jcreator :

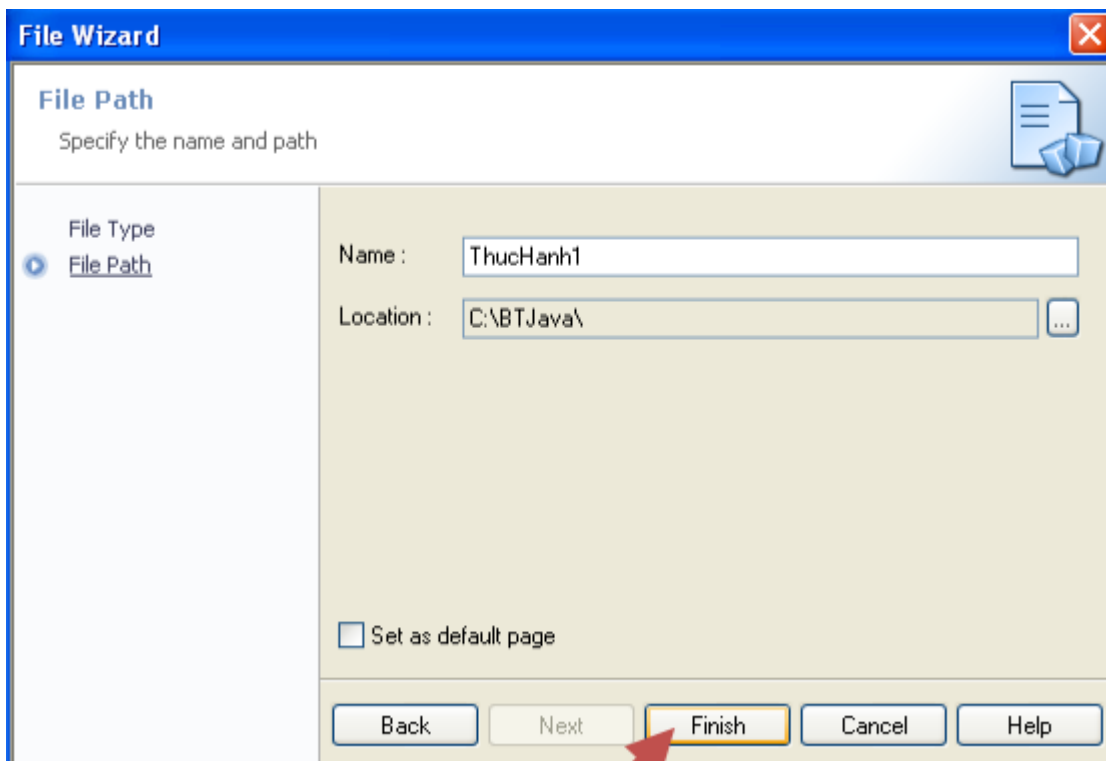
- Khởi động chương trình Jcreator.
- Để tạo mới một file chứa chương trình nguồn java, chọn menu File → New → New File



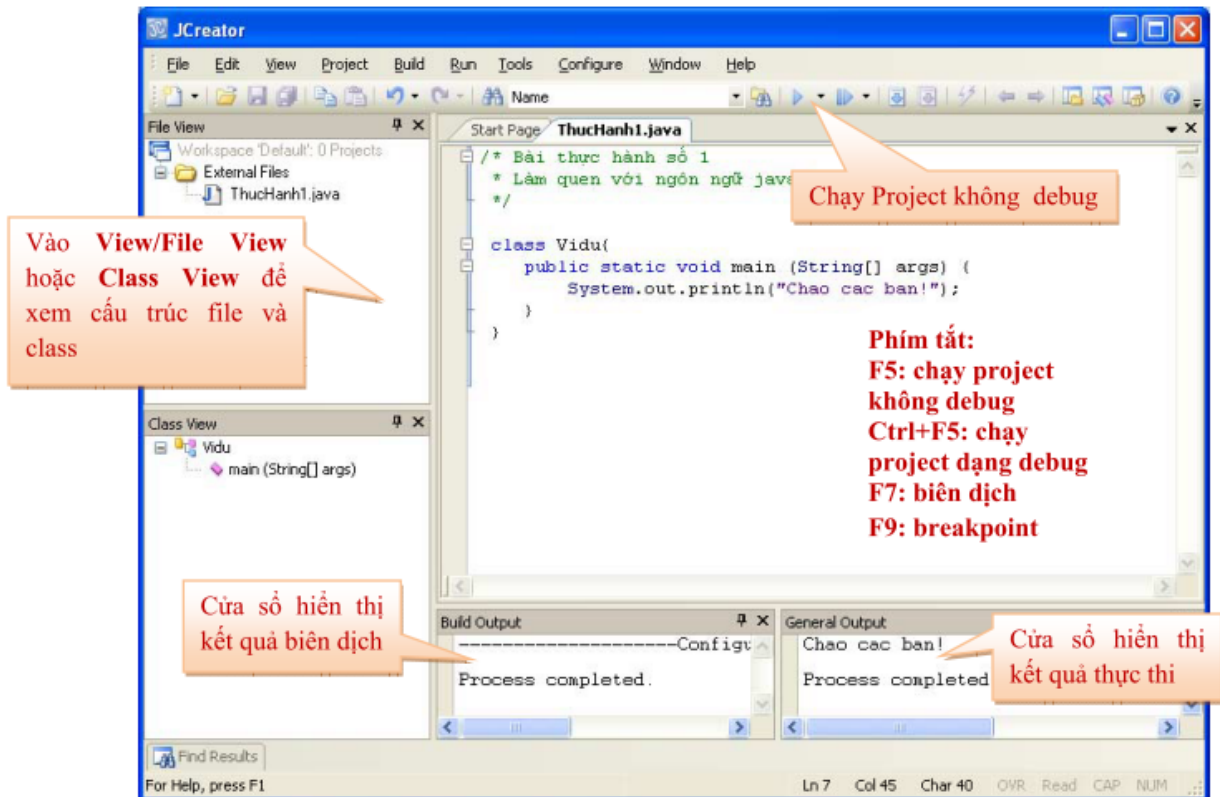
- Xuất hiện cửa sổ sau:



- Gõ tên file lưu trữ và chọn đường dẫn lưu trữ. Lưu ý file sẽ có phần mở rộng mặc định là .java.



- Các thành phần trên màn hình Jcreator:



Hướng dẫn bài tập:

Bài 1: Viết chương trình xuất ra màn hình các thông tin sau.

"Hello! I'm <your name>."

"I'm <your age> years old".

"This is my first java program."

Hướng dẫn:

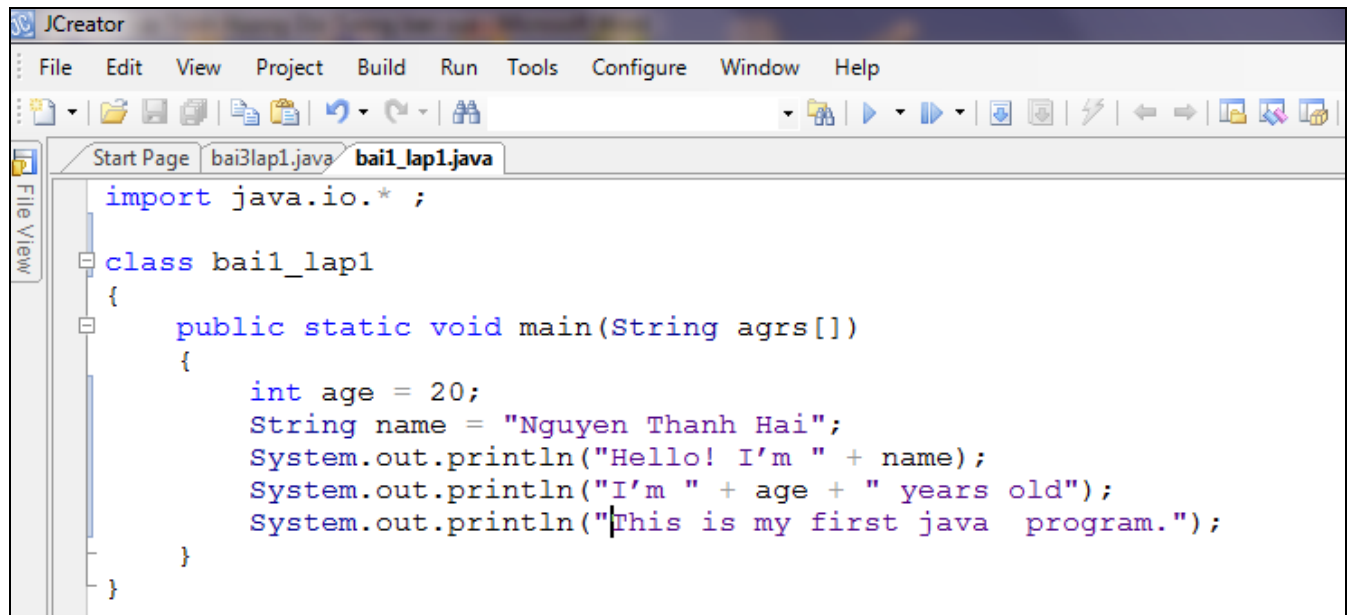
- Làm quen với các kiểu dữ liệu và cách khai báo biến:

```
int age = 20;
String name = "Nguyen Thanh Hai";
```

- Làm quen với câu lệnh xuất dữ liệu:

```
System.out.println("Hello! I'm " + name);
System.out.println("I'm " + age + " years old");
System.out.println("This is my first java program.");
```

- Tham khảo chương trình sau:



```

import java.io.* ;

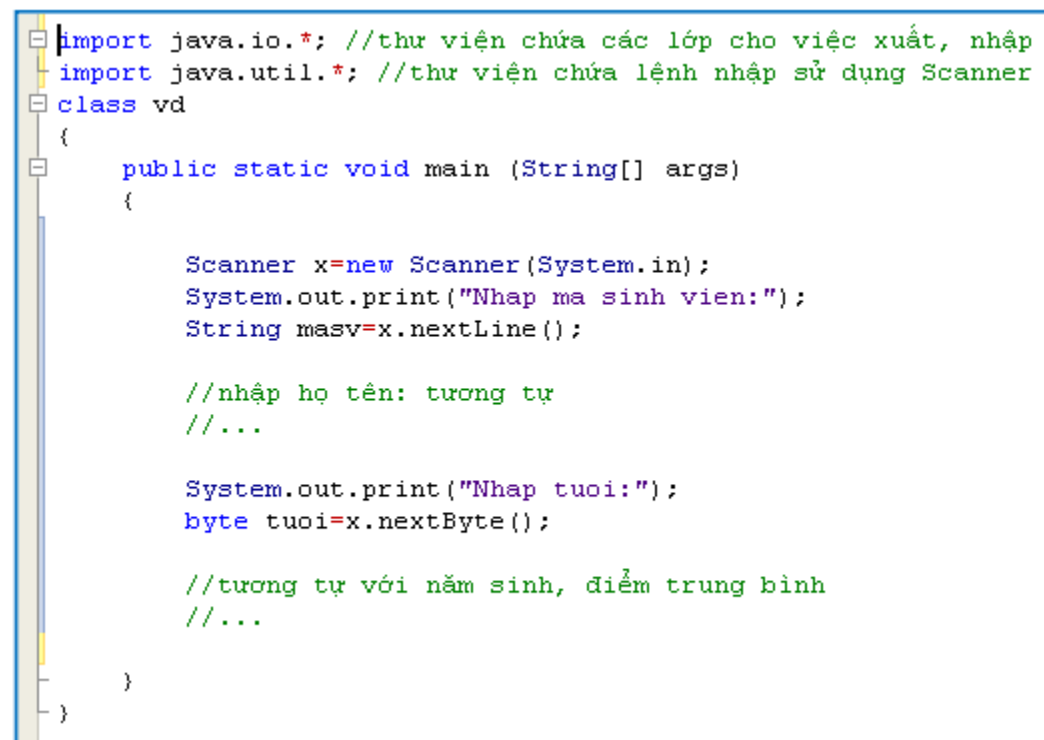
class bai1_lap1
{
    public static void main(String args[])
    {
        int age = 20;
        String name = "Nguyen Thanh Hai";
        System.out.println("Hello! I'm " + name);
        System.out.println("I'm " + age + " years old");
        System.out.println("This is my first java program.");
    }
}

```

Bài 2: Nhập các thông tin của 1 sinh viên gồm mã sinh viên, họ tên, tuổi, năm sinh, điểm trung bình. Xuất các thông tin ra màn hình.

Hướng dẫn:

- Nhập dữ liệu sử dụng đối tượng thuộc lớp Scanner trong thư viện **java.util.***. **Lưu ý:** Chọn phương thức phù hợp với kiểu dữ liệu cần nhập.



```

import java.io.*; //thư viện chứa các lớp cho việc xuất, nhập
import java.util.*; //thư viện chứa lệnh nhập sử dụng Scanner

class vd
{
    public static void main (String[] args)
    {

        Scanner x=new Scanner(System.in);
        System.out.print("Nhap ma sinh vien:");
        String masv=x.nextLine();

        //nhập họ tên: tương tự
        //...

        System.out.print("Nhap tuoi:");
        byte tuoi=x.nextByte();

        //tương tự với năm sinh, điểm trung bình
        //...

    }
}

```

Bài 3: Nhập 1 mảng các số nguyên từ bàn phím. Tìm phần tử lớn nhất trong mảng.

Hướng dẫn:

- Khai báo mảng:

- **Kiểu_dữ_liệu Tên_biến_mảng[];**
- hoặc **Kiểu_dữ_liệu[] Tên_biến_mảng;**

– Cấp phát bộ nhớ cho mảng:

tênBiếnMảng = new KiểuDữLiệu[sốPT];

```
//Khai báo mảng, khai báo biến giữ số lượng phần tử cho mảng
int a[], n=0;

//nhập số lượng phần tử của mảng: nhập n
//...bạn tự code

//cấp phát số ô nhớ = số lượng phần tử sẽ dùng
a=new int[n];

//nhập giá trị cho từng phần tử trong mảng
for(int i=0; i<n; i++)
    a[i]= ...//tự code

//tìm phần tử lớn nhất trong mảng, bạn tự code :)
```

Bài 4: Tạo một ma trận gồm m dòng và n cột, trong đó mỗi phần tử của ma trận là một giá trị nguyên được sinh ngẫu nhiên trong phạm vi [0, 50]. Xuất ma trận ra màn hình. Tính tổng giá trị các phần tử của ma trận.

Hướng dẫn:

- Mảng 2 chiều tương tự như mảng một chiều.
- Khai báo biến mảng, khai báo biến số dòng, biến số cột.
- Nhập số dòng, số cột cho ma trận. Cấp phát bộ nhớ cho mảng với số dòng, số cột tương ứng. Ví dụ: ma trận m gồm 5 dòng, 7 cột:

```
{ int m[][];
  m= new int[5][]; // cap 5 dong
  int i,j;
  for (i=0;i<5;i++) m[i]=new int[7]; // 1 dong cap 7 cot
```

- Làm tương tự cho ma trận n dòng, m cột.
- Sinh giá trị ngẫu nhiên cho từng phần tử trong ma trận $m \times n$:

```
for(int i=0; i<m; i++)  
    for(int j=0; j<n; j++)  
        a[i][j]=(int) (Math.random()*100); //phải khai báo gói java.lang.*;
```

– Hàm **random()**: là phương thức thuộc lớp Math, sinh giá trị ngẫu nhiên là một số thực trong phạm vi 0..1. Phải nhân với 100, sau đó mới ép kiểu. Các bạn giải thích vì sao nhé!

- Xuất ma trận
- Tính tổng các phần tử của ma trận: tự làm.

Nội dung bài tập mở rộng: tự làm

Bài 5. Viết các chương trình xuất bảng cửu chương từ 2 đến 9.

Bài 6. Viết các chương trình xuất trị bình phương , lập phương từ 1 đến 10.

Bài 7. Viết các chương trình tạo 1 mảng số int dạng in-line 10 phần tử, xuất mảng này tăng dần.

Bài 8. Viết các chương trình nhập 1 mảng int các số mang giá trị là mã của các ký tự nhập từ bàn phím. Xuất mảng này dạng chữ rồi xuất mã của chúng.

Bài 9. Viết các chương trình: Xuất 100 số Fibonacci đầu tiên. Dãy Fibonacci : 1,1,2,3,5,8,... 2 số đầu là 1, các số sau bằng tổng 2 số trước nó.

Bài 10. Nhập một ma trận bất kỳ từ bàn phím. Tính tổng hàng thứ k của ma trận.

TUẦN 2: XÂY DỰNG LỚP – TẠO ĐỐI TƯỢNG

Nội dung bài tập cơ bản:

Bài 1: Xây dựng lớp học sinh, biết rằng mỗi học sinh có:

- Thành phần dữ liệu: mã số, họ tên, điểm trung bình
- Phương thức: `set()`, `get()`, `input()`, `output()`, `rank()` - xếp loại cho học sinh theo dtb.

Viết lớp `Demo1` chứa phương thức `main()`:

- Tạo một đối tượng học sinh
- Nhập thông tin cho học sinh
- Xuất thông tin cùng xếp loại của học sinh.
- Đổi tên của học sinh thành một tên mới được nhập từ bàn phím
- Nhập thêm một học sinh. Cho biết tên của học sinh có điểm trung bình lớn hơn

Bài 2: Xây dựng lớp `Mang` gồm có các Thành phần dữ liệu:

- Thành phần dữ liệu: Số phần tử của mảng, mảng chứa các số nguyên, kích thước của mảng
- Phương thức: `set()`, `get()`, `input()`, `output()` và
 - Tính trị trung bình của các số lẻ
 - Tìm phần tử lớn nhất của mảng
 - Sắp xếp mảng theo thứ tự tăng dần

Viết lớp `Demo2` chứa phương thức `main()`:

- Tạo một đối tượng mảng
- Thực hiện các phương thức cho đối tượng vừa tạo.

Bài 3: Xây dựng lớp `Danh sách học sinh` gồm các phương thức:

- Nhập danh sách
- In danh sách

- Sắp xếp danh sách giảm dần theo điểm trung bình của học sinh

Viết lớp Demo3 chứa phương thức main():

- Tạo một đối tượng danh sách học sinh
- Nhập thông tin cho danh sách học sinh
- In danh sách học sinh đã được sắp thứ tự.

Bài 4: Viết lớp Demo4 chứa phương thức main():

- Tạo một danh sách đối tượng học sinh
- Nhập thông tin cho danh sách học sinh
- In danh sách học sinh

Hướng dẫn bài tập:

Bài 1: Xây dựng lớp học sinh, biết rằng mỗi học sinh có:

- Thành phần dữ liệu: mã số, họ tên, điểm trung bình
- Phương thức: set(), get(), input(), output(), rank() - xếp loại cho học sinh theo dtb.

Viết lớp Demo1 chứa phương thức main():

- Tạo một đối tượng học sinh
- Nhập thông tin cho học sinh
- Xuất thông tin cùng xếp loại của học sinh.
- Đổi tên của học sinh thành một tên mới được nhập từ bàn phím
- Nhập thêm một học sinh. Cho biết tên của học sinh có điểm trung bình lớn hơn.

Hướng dẫn:

1. Xây dựng lớp HOCSINH:

```
import java.io.*;

class HOCSINH{

}
```

Viết các Property và các Method bên trong lớp

- Khai báo các thành phần dữ liệu (Properties) có phạm vi truy xuất là **private** (để bảo mật dữ liệu, không cho truy xuất bên ngoài lớp).
- Cú pháp: **private kiểu_dữ_liệu tên_biến;**

```
private int maso;  
private String hoten;  
private float dtb;
```

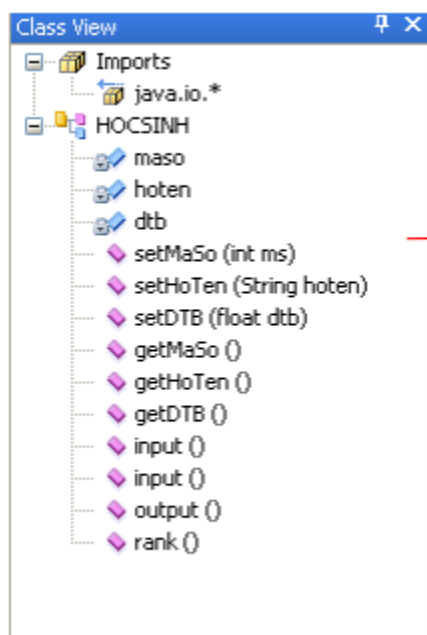
- Để truy xuất được các thuộc tính từ bên ngoài lớp, ta định nghĩa (viết code) cho các phương thức lấy giá trị (get) và đặt/ thay đổi giá trị (set) cho thành phần dữ liệu.

```
//get ma so  
public String getMaSo() {  
    return maso;  
}  
  
//set ma so  
public void setMaSo(String ms) {  
    maso = ms;  
}  
  
// tu lam cho hoten, dtb
```

- Viết phương thức input() để nhập thông tin của 1 HOCSINH
- Viết phương thức output() để xuất thông tin của lớp HOCSINH
- Viết phương thức rank() để xếp loại cho học sinh theo điểm trung bình

```
//xếp hạng học sinh theo dtb  
public void rank() {  
    if (dtb < 5)  
        System.out.println("Xep loai yeu");  
    else  
        if (dtb >= 5 && dtb < 7)  
            System.out.println("Xep loai trung binh");  
        else  
            //...tự viết tiếp  
}
```

- Sau khi viết các thuộc tính và phương thức, vào menu View → Class View, ta thấy cửa sổ Class View hiện ra bên trái như sau:



*Phát sinh theo source code
của chương trình*

2. Trong cùng file Demo1.java đang làm, tiếp tục xây dựng lớp Demo1 chứa phương thức main như sau:

```
//=====
class Demo1{
    public static void main (String[] args) {
        HOCSINH hs=new HOCSINH(); //tạo một đối tượng HOCSINH hs
        hs.input(); //gọi phương thức nhập cho đối tượng hs
        hs.output(); //gọi phương thức xuất cho đối tượng hs
        hs.rank(); //gọi phương thức xếp loại cho đối tượng hs
    }
}
```

- Xem và đổi tên của đối tượng học sinh vừa tạo, sử dụng phương thức getHoten() và setHoten() như sau (viết tiếp trong hàm main):

```
System.out.println("Ho ten cua hs vua nhap:" + hs.getHoten());
//nhập chuỗi họ tên mới
String htmoi="";
//bạn tự nhập...:D
//thay họ tên đã có của học sinh bằng chuỗi họ tên mới nhập vào
hs.setHoTen(htmoi);
//xem lại họ tên của học sinh sau khi sửa
System.out.println("Ho ten cua hs sau khi sua:" + hs.getHoten());
```

- Vào menu View → Class View, cửa sổ Class View lúc này có thêm lớp Demo1

Bài 2: Xây dựng lớp Mảng gồm có các Thành phần dữ liệu:

- Thành phần dữ liệu: Số phần tử của mảng, mảng chứa các số nguyên, kích thước của mảng
- Phương thức: `set()`, `get()`, `input()`, `output()` và
 - Tính trị trung bình của các số lẻ
 - Tìm phần tử lớn nhất của mảng
 - Sắp xếp mảng theo thứ tự tăng dần

Viết lớp Demo2 chứa phương thức `main()`:

- Tạo một đối tượng mảng
- Thực hiện các phương thức cho đối tượng vừa tạo.

Hướng dẫn:

1. Xây dựng lớp Mảng gồm các thành phần dữ liệu và các phương thức theo yêu cầu của đề bài, làm tương tự bài 1.
2. Xây dựng lớp Demo2 chứa phương thức `main()`:
 - Tạo một đối tượng mảng thuộc lớp Mảng
 - Thực hiện các phương thức đã định nghĩa ở lớp Mảng cho đối tượng vừa tạo.

Bài 3: Xây dựng lớp Danh sách học sinh gồm các phương thức:

- Nhập danh sách
- In danh sách
- Sắp xếp danh sách giảm dần theo điểm trung bình của học sinh

Viết lớp Demo3 chứa phương thức `main()`:

- Tạo một đối tượng danh sách học sinh
- Nhập thông tin cho danh sách học sinh
- In danh sách học sinh đã được sắp thứ tự.

Hướng dẫn:

Sử dụng bài 1, tiếp tục xây dựng lớp DSHOCSINH:

```
class DSHOCSINH(
    //các thuộc tính
    private HOCSINH ds[]; //ds là mảng 1 chiều, mỗi phần tử lưu 1 hs
    private int sl; //số lượng phần tử của mảng

    //các phương thức
    public void nhapds(){
        //nhập số lượng học sinh: nhập sl
        //...tự code

        ds=new HOCSINH[sl]; //cấp phát bộ nhớ cho mảng ds
        //nhập thông tin cho từng người trong ds
        for(int i=0; i<sl; i++)
        {
            //phần tử thứ i trong mảng là 1 đối tượng học sinh nên phải khởi tạo đối tượng (cấp phát ô nhớ)
            ds[i]=new HOCSINH();
            ds[i].Input(); //nhập dữ liệu cho học sinh thứ i
        }
    }
    public void xuatds(){
        System.out.println("Danh sach hoc sinh la:\n");
        for(int i=0; i<sl; i++)
            ds[i].Output();
    }
    public void sapxep(){
        //bạn tự code - good luck !^!
    }
}
```

Viết lớp DEMO3 chứa phương thức main():

```
class Demo3{
    public static void main (String[] args) {
        DSHOCSINH danh sach = new DSHOCSINH(); //tao danh sach
        danh sach.nhapds();
        danh sach.xuatds();
        danh sach.sapxep();
    }
}
```

Bài 4: Viết lớp Demo chứa phương thức main():

- Tạo một danh sách đối tượng học sinh
- Nhập thông tin cho danh sách học sinh
- In danh sách học sinh

Hướng dẫn:

Xây dựng lớp Demo chứa phương thức main() như sau:

```
//=====
class Demo {
    public static void main (String[] args) {

        //khai báo mảng, khởi tạo bộ nhớ cho mảng chứa tối đa 20 hs
        HOCSINH[] dshs=new HOCSINH[20];
        int n=0;
        //nhập số lượng học sinh, nhập n
        //...tự viết

        //nhập thông tin cho từng học sinh dshs[i]
        for(int i=0; i<n; i++){
            dshs[i]=new HOCSINH(); //khởi tạo đối tượng HOCSINH thứ i
            dshs[i].input();
        }

        //xuất thông tin từng học sinh
        //...tương tự, tự viết

    }
}
```

TUẦN 3: LỚP - PHƯƠNG THỨC KHỞI TẠO

Nội dung bài tập cơ bản:

Bài 1: Làm lại các bài 1,2,3: viết thêm phương thức khởi tạo(constructor) cho các lớp.

Bài 2: Tạo lớp phân số bao gồm:

Thành phần dữ liệu:

- Tử số
- Mẫu số

Phương thức:

- Phương thức khởi tạo chuẩn
- Phương thức khởi tạo 2 tham số.
- Nhập phân số
- In phân số
- Tính ước số chung lớn nhất
- Rút gọn phân số
- Cộng 2 phân số
- Trừ 2 phân số
- Nhân 2 phân số
- Chia 2 phân số

Bài 3: Xây dựng lớp COODINATE: tọa độ của điểm trong không gian hai chiều.

- Thành phần dữ liệu: hoành độ x và tung độ y
- Các phương thức gồm:
 - Phương thức khởi tạo chuẩn: $x = 0, y = 0$
 - Phương thức khởi tạo 2 tham số.
 - Phương thức tính tổng các thành phần x và y của 2 điểm.

- Phương thức tìm điểm đối xứng của một điểm.
- Phương thức in tọa độ của một điểm.

Bài 4: Xây dựng lớp Nhân Viên gồm: mã số, họ tên, lương cơ bản, hệ số lương, số lượng nhân viên; các constructor chuẩn, có tham số, sao chép; các phương thức: set/get cho mã số, họ tên, hệ số lương; các phương thức: nhập dữ liệu cho nhân viên từ bàn phím, toString() để mô tả đối tượng nhân viên, tính lương cho nhân viên, in số lượng nhân viên.

Xây dựng lớp Demo chứa phương thức main() thực hiện các việc sau:

- Tạo 3 đối tượng Nhân viên bằng 3 cách khác nhau
- Nhập dữ liệu cho một nhân viên từ bàn phím, xuất ra màn hình
- Thay đổi họ tên cho nhân viên và xuất ra màn hình
- In ra thông tin của nhân viên có hệ số lương cao nhất trong 3 nhân viên
- Nhập danh sách nhân viên và xuất ra màn hình cùng với lương của mỗi nhân viên
- In số lượng nhân viên trong danh sách.

Hướng dẫn bài tập:

Bài 1: Làm lại các bài 1,2,3 : viết thêm phương thức khởi tạo(constructor) cho các lớp.

Hướng dẫn:

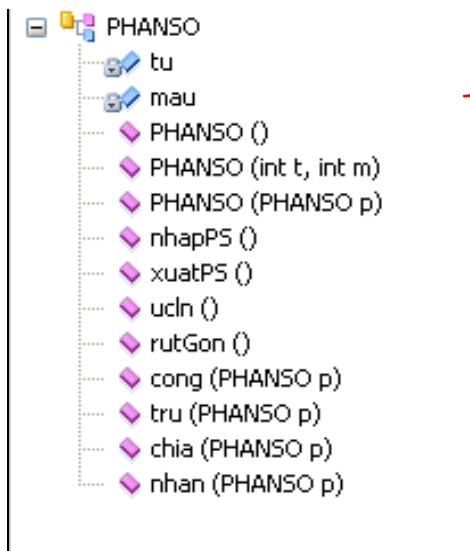
Sử dụng các lớp đã xây dựng trong các bài 1, 2, 3 ở tuần 2, mỗi lớp thêm vào ít nhất 3 loại phương thức khởi tạo: chuẩn, có tham số và sao chép.

Trong lớp Demo, bên trong phương thức main(), tạo các đối tượng bằng các loại phương thức khởi tạo khác nhau. So sánh sự khác biệt giữa các đối tượng được tạo.

Bài 2: Tạo lớp phân số

Hướng dẫn:

1. Xây dựng lớp PHANSO gồm các thuộc tính và phương thức như sau:



*Cửa sổ Class View
(menu View → Class View) sau khi cài đặt tất cả các thành phần của lớp*

- Cách viết các phương thức khởi tạo (Constructor):

- Phương thức khởi tạo mặc định, không tham số: Khi gọi phương thức này, chương trình sẽ tạo một phân số có tử số mặc định là 0, mẫu số mặc định là 1

```
//phương thức khởi tạo chuẩn, không tham số
//mặc định tử=0 và mẫu=1
public PHANSO() {
    tu=0; mau=1;
}
```

- Phương thức khởi tạo phân số khi biết tử và mẫu, tham số truyền vào là hai giá trị tương ứng cho tử và mẫu. Khi gọi phương thức này, chương trình sẽ tạo ra một phân số có tử số bằng giá trị của tham số thứ nhất truyền vào và mẫu số bằng giá trị của tham số thứ hai truyền vào.

```
//phương thức khởi tạo 2 tham số
public PHANSO(int t, int m){
    tu=t; mau=m;
}
```

- Phương thức khởi tạo phân số từ một phân số đã có, tham số truyền vào là một phân số. Khi gọi PTKT này, chương trình sẽ tạo ra một phân số có tử bằng tử của phân số truyền vào, có mẫu bằng mẫu của phân số truyền vào.


```
//pt khởi tạo phân số từ một phân số đã có
//còn gọi là pt khởi tạo sao chép
public PHANSO(PHANSO p){
    this.tu=p.tu;
    this.mau=p.mau;
}
```

- Viết phương thức nhập và xuất phân số

```
//nhập phân số
public void nhapPS(){
    //nhập tử số -- bạn tự viết :>

    //nhập mẫu số
}
//xuất phân số
public void xuatPS(){
    //xuất ra màn hình dưới dạng tu/mau
    //ví dụ: 3/4 -- bạn tự viết ;)|
}
```

- Viết phương thức tìm ước chung lớn nhất của hai số nguyên để dùng cho việc rút gọn phân số sau này

Thuật toán tìm ucln của 2 số nguyên a và b

Trong khi $a \neq b$ thì thực hiện:

Nếu $a > b$ thì $a = a - b$

Ngược lại thì $b = b - a$

$UCLN = a$;

```
//tìm ước chung lớn nhất của hai số nguyên
//trả về: giá trị ucln tìm được
public int ucln(int a, int b){
}
}
```

- Viết phương thức rút gọn hai phân số

```
//rút gọn phân số
public void rutGon(){
    int tam=ucln(tu, mau);
    //tự viết tiếp.. :>
}
```

- Viết phương thức cộng hai phân số: Cộng phân số là đối tượng của lớp đang xét với phân số p là tham số truyền vào. Kết quả trả về là phân số tổng.
 - Bước 1: tìm mẫu số chung, mẫu số của phân số kết quả bằng mẫu số chung
 - Bước 2: tìm tử số của phân số kết quả bằng cách cộng tử của 2 phân số sau khi quy đồng
 - Bước 3: rút gọn phân số kết quả và trả về (*Bạn hãy tự làm, khi nào bị lỗi hoặc cho kết quả sai thì mới tham khảo hướng dẫn sau nhé!*)

```
//cộng 2 phân số
//cộng phân số của lớp hiện tại với phân số truyền vào
public PHANSO cong(PHANSO p){

    //Khởi tạo kq là phân số kết quả tính được
    PHANSO kq=new PHANSO();

    kq.mau=this.mau*p.mau;
    kq.tu=this.tu*p.mau + p.tu*this.mau;

    //sau khi cộng xong, rút gọn phân số kết quả
    kq.rutGon();
    return kq;
}
```

- Các phương thức trừ, nhân, chia làm tương tự
2. Xây dựng lớp DEMO chứa phương thức main() thực hiện:
- Tạo đối tượng phân số $p1$ dùng PTKT mặc định không tham số, xuất ra màn hình, quan sát kết quả.
 - Gọi hàm *nhapPS()* cho phân số $p1$ và xuất ra màn hình.
 - **Lưu ý:** Với cách khởi tạo này thì phân số tạo ra luôn có tử =0 và mẫu =1. Nếu muốn phân số có tử mẫu tùy muốn thì phải sử dụng hàm *nhapPS()*
 - Tạo đối tượng phân số $p2 = 4/16$ dùng PTKT hai tham số, xuất ra màn hình, quan sát kết quả.

```
PHANSO p2=new PHANSO(4, 16);
p2.xuatPS();
```

- Tạo đối tượng phân số $p3$ dùng PTKT hai tham số, với tham số là giá trị được nhập từ bàn phím

```
int t, m;  
  
//nhập giá trị cho 2 biến t, m  
  
//...bạn tự nhập  
  
//khởi tạo phân số p3 có tử mẫu theo tham số  
truyền vào là t, m  
  
PHANSO p3=new PHANSO(t, m);  
  
p3.xuatPS();
```

- **Lưu ý:** Với cách khởi tạo này thì hàm nhapPS() không được dùng đến => không phải viết hàm nhapPS() khi bạn sử dụng cách khởi tạo này.
- Cộng phân số $p1$ và phân số $p3$, xuất ra màn hình phân số kết quả
- Tạo đối tượng phân số $p4$ dùng PTKT sao chép từ phân số kết quả tính được ở trên
- Nhân $p4$ với $p2$, xuất ra màn hình phân số kết quả

Bài 3: Xây dựng lớp COORDINATE: tọa độ của điểm trong không gian hai chiều.

Hướng dẫn:

Làm tương tự bài 2

TUẦN 4: LỚP - PHƯƠNG THỨC KHỞI TẠO, STATIC

Nội dung bài tập cơ bản:

Bài 1: Viết chương trình khai báo 1 lớp có tên HCN (Hình Chữ Nhật). Lớp này có 2 thành phần dữ liệu là chiều dài và chiều rộng hình chữ nhật.

- Viết 3 phương thức khởi tạo (constructor) tường minh cho lớp này:
 - Phương thức khởi tạo không tham số, mặc định chiều dài và chiều rộng của hình chữ nhật bằng 1.
 - Phương thức khởi tạo với 1 tham số kiểu int, khi đó chiều dài và chiều rộng được khởi tạo với giá trị tham số đưa vào (hình vuông).
 - Phương thức khởi tạo với 2 tham số kiểu int, tương ứng là chiều dài và chiều rộng của hình chữ nhật.
- Viết phương thức tính chu vi và diện tích của hình chữ nhật.
- Viết lớp thử nghiệm (DEMO class) cho lớp HCN vừa tạo (trong lớp này chứa phương thức main) để mô tả một vài đối tượng được tạo từ lớp HCN. Cho biết diện tích và chu vi của mỗi đối tượng.

Bài 2: Sử dụng bài 1, tiếp tục xây dựng lớp DSHCN (danh sách hình chữ nhật) gồm các thuộc tính:

- Mảng các đối tượng hình chữ nhật
- Số lượng hình chữ nhật

Và phương thức:

- Nhập danh sách
- In danh sách

Sửa đổi phương thức main() ở lớp Demo:

- Tạo một đối tượng danh sách hình chữ nhật
- Nhập thông tin cho danh sách hình chữ nhật
- In danh sách hình chữ nhật với diện tích và chu vi tương ứng.

Bài 3: Xây dựng lớp Nhân Viên gồm: mã số, họ tên, lương cơ bản, hệ số lương, số lượng nhân viên; các constructor chuẩn, có tham số, sao chép; các phương thức: set/get cho mã số, họ tên, hệ số lương; các phương thức: nhập dữ liệu cho nhân viên từ bàn phím, toString() để mô tả đối tượng nhân viên, tính lương cho nhân viên, in số lượng nhân viên.

Xây dựng lớp Demo chứa phương thức main() thực hiện các việc sau:

- Tạo 3 đối tượng Nhân viên bằng 3 cách khác nhau
- Nhập dữ liệu cho một nhân viên từ bàn phím, xuất ra màn hình
- Thay đổi họ tên cho nhân viên và xuất ra màn hình
- In ra thông tin của nhân viên có hệ số lương cao nhất trong 3 nhân viên
- Nhập danh sách nhân viên và xuất ra màn hình cùng với lương của mỗi nhân viên
- In số lượng nhân viên trong danh sách.

Hướng dẫn bài tập:

Bài 1: Xây dựng lớp HCN, xác định xem lớp có những thuộc tính (properties) và phương thức (method, behavior) nào?

```
//lớp hình chữ nhật
class HCN {

    //các thuộc tính???

    ...

    //các phương thức???

    ...
}
```

Xây dựng lớp thử nghiệm (DEMO class) cho lớp HCN vừa tạo

```
class DEMO {  
    public static void main (String[] args) {  
        //Tạo đối tượng  
        ...  
        //Gọi thực thi các phương thức  
        ...  
    }  
}
```

Bài 2: Xây dựng lớp DSHCN (danh sách hình chữ nhật): Tự làm

Bài 3: Xây dựng lớp Nhân Viên:

Hướng dẫn:

Xây dựng lớp Nhân Viên tương tự các bài trên. Điểm khác biệt là lớp này có các thành phần static: dữ liệu static và phương thức static. Tham khảo các ví dụ trong mục 3.8, đọc và phân tích yêu cầu để xác định chính xác dữ liệu static và phương thức static.

1. Xây dựng lớp Nhân Viên có các dữ liệu sau:

```
String maso, hoten;  
static float lcb;  
static int slnv = 0;  
float hsl;
```

- Tự xây dựng các phương thức: các constructor chuẩn, có tham số, sao chép; các phương thức: set/get cho mã số, họ tên, hệ số lương; các phương thức: nhập dữ liệu cho nhân viên từ bàn phím.
- toString() để mô tả đối tượng nhân viên:

```
public String toString(){  
    return (maso + " " + hoten + " " + hsl);  
}
```

- Phương thức tính lương cho nhân viên: tự viết

- Phương thức in số lượng nhân viên: là phương thức static

```
static void inSLNV() {  
    System.out.print("So luong nv: " + slnv);  
}
```

2. Xây dựng lớp Demo chứa phương thức main() thực hiện các việc sau:
 - a. Tạo 3 đối tượng Nhân viên bằng 3 cách khác nhau
 - b. Nhập dữ liệu cho một nhân viên từ bàn phím, xuất ra màn hình
 - c. Thay đổi họ tên cho nhân viên và xuất ra màn hình
 - d. In ra thông tin của nhân viên có hệ số lương cao nhất trong 3 nhân viên

```
float max = nv1.getHSL();  
NhanVien nvt = new NhanVien(nv1);  
if(nv2.getHSL() > max) {  
    max = nv2.getHSL();  
    nvt = nv2;  
}  
if(nv3.getHSL() > max) {  
    max = nv3.getHSL();  
    nvt = nv3;  
}  
System.out.println("Nhan vien co hsl max: ");  
System.out.println(nvt);
```

- a. Nhập danh sách nhân viên và xuất ra màn hình cùng với lương của mỗi nhân viên
- b. In số lượng nhân viên trong danh sách.

Mở rộng

Tương tự, làm các bài sau:

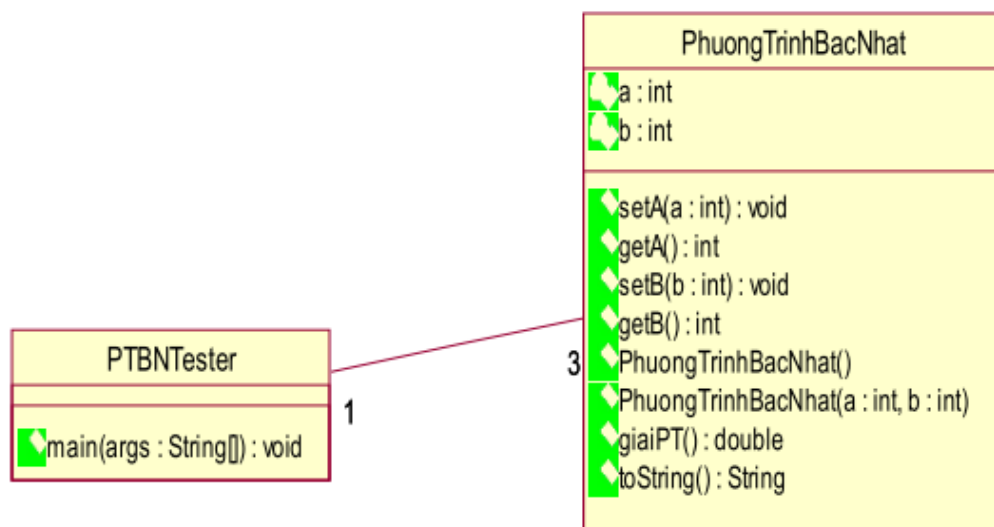
Bài 4: Xây dựng lớp Tam giác gồm các Thành phần dữ liệu: Độ dài cạnh thứ nhất, độ dài cạnh thứ hai, độ dài cạnh thứ ba của tam giác.

- Viết Phương thức khởi tạo (constructor) với 3 tham số kiểu int tương ứng là 3 cạnh của tam giác.

- Viết các phương thức của đối tượng tam giác: Tính chu vi tam giác, tính diện tích tam giác và xác định loại tam giác.
- Viết lớp thử nghiệm (driver class) cho lớp tam giác vừa tạo (trong lớp này chứa phương thức main) để mô tả một vài đối tượng được tạo từ lớp tam giác. Cho biết diện tích và chu vi của mỗi tam giác, đồng thời cho biết loại tam giác.

Bài 5: Tạo lớp `PhuongTrinhBacNhat` có 2 biến `a` và `b` là 2 số nguyên.

- Định nghĩa các phương thức setters/getters cho các biến.
- Tạo 2 Phương thức constructors cho đối tượng:
 - constructor default: là constructor không có tham số, dùng để khởi gán các giá trị bằng 0 cho các biến của đối tượng.
 - constructor copy: constructor có đầy đủ tham số (số tham số của constructor này bằng với số data instance ta đã khai báo). Constructor này thường dùng để khởi tạo 1 đối tượng đầy đủ.
- Định nghĩa phương thức đặc tả dạng chuỗi của đối tượng (phương thức `toString`).
- Viết một phương thức `giaiPT` dùng để giải phương trình bậc nhất $ax+b=0$
- Viết lớp cho phần thử nghiệm (Driver Class) của lớp `PhuongTrinhBacNhat` vừa tạo.
- UML class diagram:



TUẦN 5: KẾ THỪA và ĐA HÌNH

Nội dung bài tập cơ bản:

Bài 1: Phân tích phân cấp kế thừa cho các lớp:

HÀNG ĐIỆN MÁY (Mã hàng, tên hàng, giá, thời gian bảo hành, điện áp, công suất)

HÀNG THỰC PHẨM (Mã hàng, tên hàng, giá, ngày sản xuất, ngày hết hạn dùng)

Xây dựng các lớp theo sơ đồ phân cấp kế thừa và lớp thử nghiệm chứa phương thức main() cho phép tạo mỗi loại một mặt hàng cụ thể, sau đó xuất thông tin về các mặt hàng này.

Bài 2: Sử dụng bài 1, tiếp tục xây dựng lớp DSHANGHOA (danh sách chứa nhiều mặt hàng, vừa có hàng điện máy vừa có hàng thực phẩm). Xây dựng hàm main sao cho chương trình có các chức năng sau:

- Tạo một danh sách gồm nhiều mặt hàng, vừa có hàng điện máy vừa có hàng thực phẩm.
- Xuất danh sách các mặt hàng theo loại.

Hướng dẫn bài tập:

Bài 1:

1. Phân tích phân cấp kế thừa cho các lớp:

- Tìm đặc điểm chung giữa hai lớp đối tượng HÀNG ĐIỆN MÁY và HÀNG THỰC PHẨM:

HÀNG ĐIỆN MÁY (**Mã hàng, tên hàng, giá**, thời gian bảo hành, điện áp, công suất)

HÀNG THỰC PHẨM (**Mã hàng, tên hàng, giá**, ngày sản xuất, ngày hết hạn dùng)

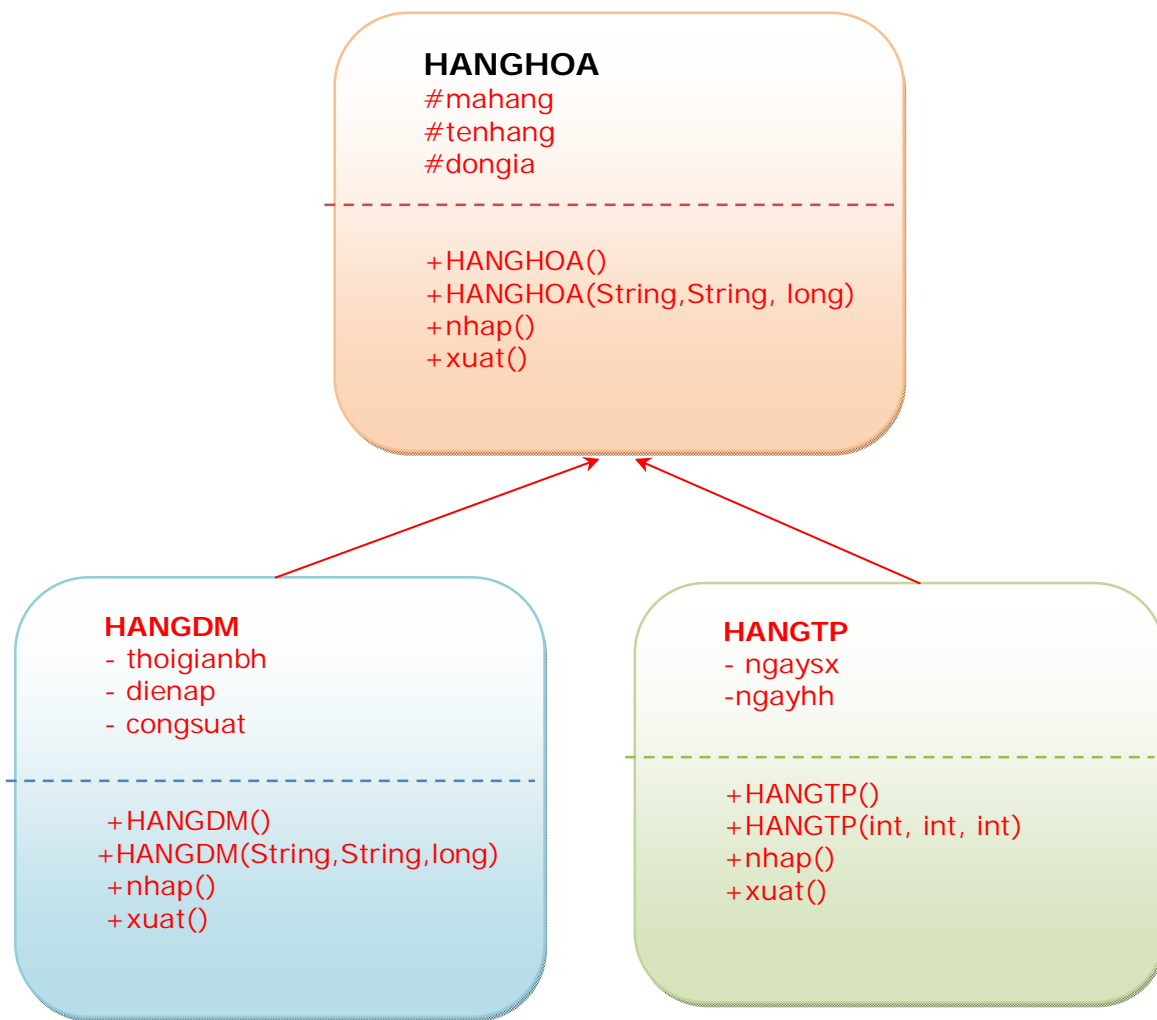
- Xây dựng lớp cha gồm những thành phần chung của hai lớp, giả sử đặt tên lớp cha là HÀNG HÓA, lớp cha có những thành phần dữ liệu là: Mã hàng, tên hàng, giá. Các phương thức chung là: nhập() và xuất().
 - Phần còn lại là các lớp con
- Sơ đồ lớp phân cấp kế thừa như sau

- Kí hiệu:

khai báo với phạm vi **protected** (chỉ được truy xuất trong phạm vi lớp cha và lớp con)

+ khai báo với phạm vi **public** (được truy xuất ở đâu??? – kiến thức cũ)

- khai báo với phạm vi **private** (được truy xuất ở đâu??? – kiến thức cũ)



2. Cài đặt các lớp dựa trên sơ đồ đã thiết kế:

- **Cài đặt lớp HANGHOA** như bình thường, trong đó các thành phần dữ liệu mã hàng, tên hàng, đơn giá được khai báo với phạm vi **protected** (cho phép các lớp con kế thừa nhưng không được truy xuất bên ngoài lớp cha và bên ngoài lớp con).

```

class HANGHOA {
    protected String mahang;
    protected String tenhang;
    protected long dongia;

    //PTKT mặc định không tham số
    public HANGHOA() {
        mahang="";
        tenhang="";
        dongia=0;
    }
    //PTKT có tham số truyền cho 3 thuộc tính tương ứng
    public HANGHOA(String mh, String th, long dg) {
        //tự code..
    }
    //hàm nhập
    public void nhap(){
        //tự code..
    }
    //hàm xuất
    //tự code....:)
}

```

- **Cài đặt lớp HANGDM** (hàng điện máy) kế thừa lớp HANGHOA, HANGDM sẽ có tất cả các thành phần của lớp HANGHOA, ngoài ra nó có thêm một số thành phần dữ liệu riêng (*thời gian bảo hành, điện áp, công suất*)

- Cú pháp khai báo lớp kế thừa: **class** <tên lớp con> **extends** <tên lớp cha> {...}

```

class HANGDM extends HANGHOA{
    private int thoigianbh;
    private int dienap;
    private int congsuat;

    //....
}

```

- Viết các phương thức khởi tạo cho HANGDM.
 - Một mặt hàng điện máy có bao nhiêu thuộc tính (properties)? → có 6 thuộc tính, 3 thuộc tính kế thừa từ lớp cha (*mahang, tenhang, dongia*) và 3 thuộc tính (*thoigianbh, dienap, congsuat*) của riêng nó.
 - PTKT mặc định không tham số:

```

public HANGDM(){
    super();
    thoigianbh=0;
    dienap=0;
    congsuat=0;
}

```

//gọi PTKT của lớp cha để khởi tạo mặc định các thuộc tính mã hàng, tên hàng, đơn giá. Phương thức super() không có tham số vì gọi theo PTKT không tham số của lớp cha.

- PTKT có tham số: Khởi tạo một mặt hàng điện máy có các thuộc tính nhận giá trị tương ứng theo các tham số truyền vào

```

public HANGDM(String mh, String th, long dg, int tg, int da, int cs){
    super(mh, th, dg);
    thoigianbh=tg;
    dienap=da;
    congsuat=cs;
}

```

//các tham số truyền cho thuộc tính kế thừa từ lớp cha

//các tham số truyền cho thuộc tính của riêng lớp HANGDM

*//gọi PTKT của lớp cha để khởi tạo các thuộc tính mã hàng, tên hàng, đơn giá theo các tham số truyền vào. Phương thức **super(mh, th, dg)** có tham số vì gọi theo PTKT có tham số của lớp cha.*

- Viết phương thức nhập cho HANGDM (bạn tự code)
 - Nhập giá trị cho các thuộc tính kế thừa, gọi phương thức nhập của lớp cha. Có pháp gọi 1 phương thức của lớp cha: **super.tenPhuongThuc(các tham số nếu có);**
 - Nhập giá trị cho các thuộc tính của riêng nó (nhập như bình thường).
- Viết phương thức xuất cho HANGDM (bạn tự code)
 - Xuất giá trị của các thuộc tính kế thừa, gọi phương thức xuất của lớp cha

- Xuất giá trị của các thuộc tính của riêng nó.
- **Cài đặt lớp HANGTP** (hàng thực phẩm) kế thừa lớp HANGHOA tương tự như lớp HANGDM.
- **Cài đặt lớp DEMO** chứa phương thức main() cho phép tạo mỗi loại một mặt hàng cụ thể (sử dụng các cách khởi tạo khác nhau), sau đó xuất thông tin về các mặt hàng này.
 - Biến đối tượng là tham khảo nên hoàn toàn có thể khai báo biến là lớp cha nhưng khởi tạo biến là đối tượng thuộc lớp con.
 - VD: HANGHOA h=new HANGDM();

Bài 2: Sử dụng bài 1, tiếp tục xây dựng lớp DSHANGHOA

1. Để lưu trữ danh sách hàng hóa, ta sử dụng mảng một chiều. Vì mảng chứa nhiều mặt hàng, vừa có hàng điện máy vừa có hàng thực phẩm nên kiểu dữ liệu của mảng không thể là kiểu HANGDM hay HANGTP được.

Áp dụng cơ chế liên kết muện ta khai báo kiểu của mảng là kiểu HANGHOA. Khi tạo mặt hàng cụ thể cho từng phần tử của mảng ta mới khởi tạo theo loại mặt hàng mong muốn.

2. Xây dựng lớp DSHANGHOA có thuộc tính: số lượng mặt hàng, danh sách các mặt hàng và phương thức:
 - Khởi tạo danh sách ban đầu chưa có mặt hàng nào (số lượng =0)
 - Thêm 1 mặt hàng vào danh sách: `public void themMH(HANGHOA h){...}`
 - Xuất mặt hàng theo loại: `public void xuatDSTheoLoai(String loai){...}`
3. Xây dựng hàm main để chương trình thực hiện theo chức năng như sau

```
byte chon;
Scanner doc=new Scanner(System.in);
DSHANGHOA dshh=new DSHH();
do{
    System.out.println("1: Them mot mat hang vao dang sach");
    System.out.println("2: Xuat mat hang theo loai");
```

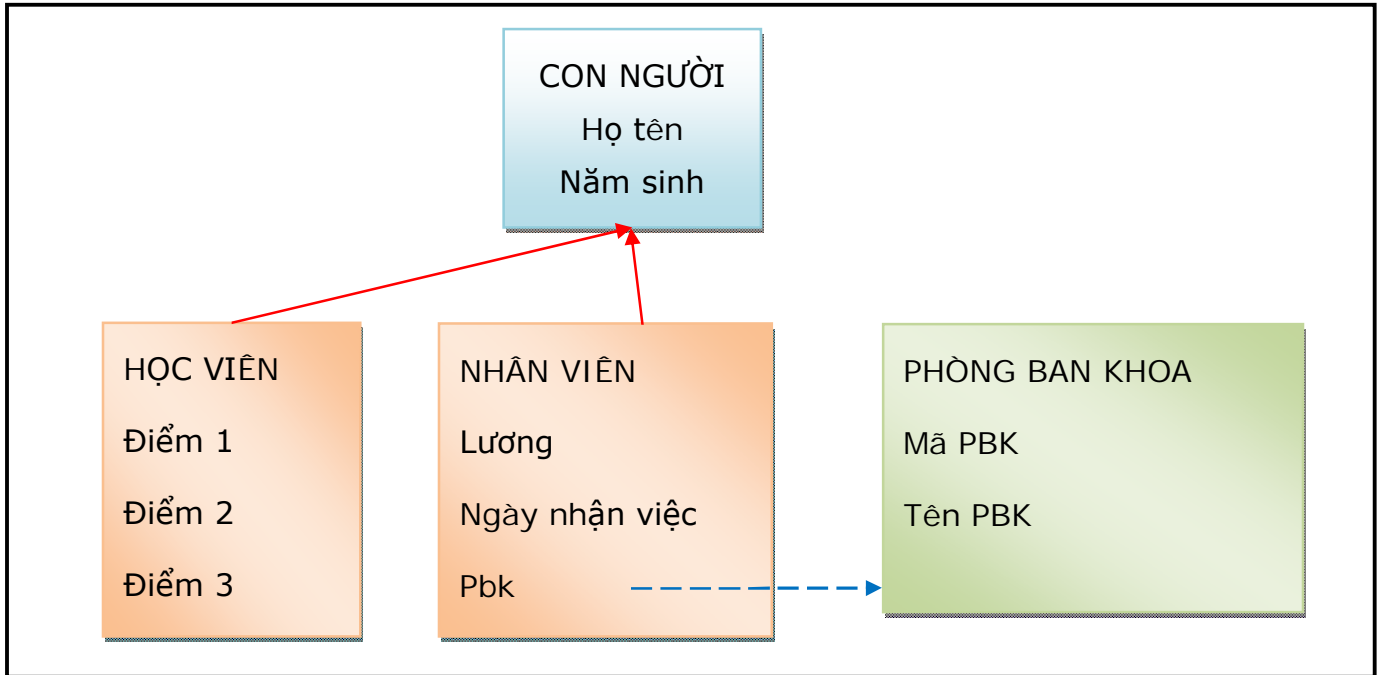
```
System.out.println("0: Thoat");
System.out.println("Chon chuc nang:"); chon=doc.nextByte();
switch(chon){
    case 1:
        System.out.println("1: hang dien may, 2: hang thuc pham. Hay chon
loai mat hang:");
        byte loai=doc.nextByte();
        if (loai==1) HANGHOA h=new HANGDM();
        else HANGHOA h=new HANGTP();
        h.nhap();
        dshh.themMH(h);
        break;
    case 2:
        System.out.println("Xuat danh sach mat hang nao (DM/TP):");
        String loaimh=doc.nextLine();
        xuatDSTheoLoai(loaimh);
        break;
    default: chon=0; break;
}
}while (chon!=0);
```

Nâng cao

Hãy xây dựng lớp DSHANGHOA trong đó thuộc tính danh sách không sử dụng kiểu dữ liệu mảng sơ cấp (ví dụ: HANGHOA ds[]) mà sử dụng kiểu dữ liệu lớp đối tượng [ArrayList](#).

TUẦN 6: QUAN HỆ GIỮA CÁC LỚP: THỪA KẾ VÀ BAO GỘP**Nội dung bài tập cơ bản:**

Bài 1: Viết chương trình minh họa thiết kế sau:



Viết chương trình khởi tạo một số đối tượng gồm: 1 học viên, 2 nhân viên. Xuất thông tin của các đối tượng.

Hướng dẫn

1. Xây dựng lớp CON NGƯỜI
 2. Xây dựng lớp HỌC VIÊN kế thừa lớp con người
 3. Xây dựng lớp PHÒNG BAN KHOA
 4. Xây dựng lớp NHÂN VIÊN kế thừa lớp CON NGƯỜI, và có dữ liệu PBK là một đối tượng của lớp PHÒNG BAN KHOA.
- Trong phương thức nhập dữ liệu cho một đối tượng NHANVIEN
 - Gọi hàm nhập của lớp cha để nhập *Họ tên* và *Năm sinh* của nhân viên
 - Nhập thông tin *Lương* và *Ngày nhận việc* của nhân viên như bình thường
 - Để nhập *phòng ban khoa* của nhân viên, cần khởi tạo đối tượng

```
Pbk=new PHONGBANKHOA();
```

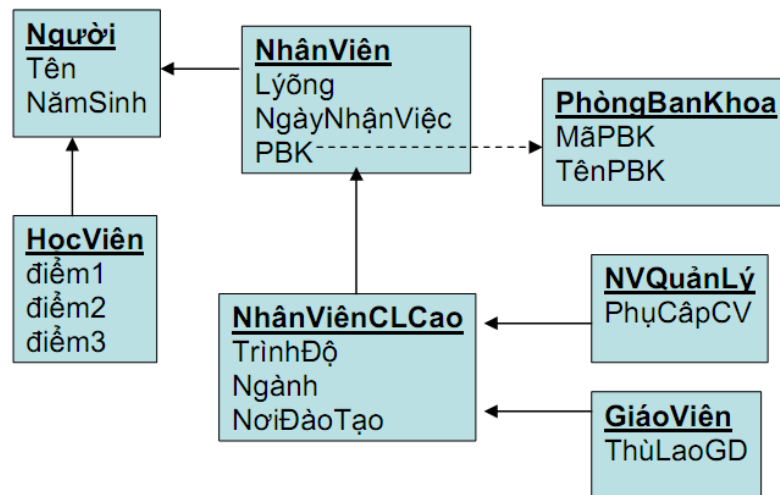
```
Pbk.nhap(); //gọi phương thức nhap() của đối tượng Pbk
```

- Tương tự với phương thức xuất dữ liệu cho một đối tượng NHANVIEN

Mở rộng

Tương tự, làm bài sau:

- Viết chương trình minh họa thiết kế sau:



Chương trình có giao diện như sau:

- Nhập một học viên
- Nhập một nhân viên quản lý
- Nhập một giáo viên
- Xuất thông tin một học viên
- Xuất thông tin một nhân viên quản lý
- Xuất thông tin một giáo viên

TUẦN 7: LỚP TRỪU TƯỢNG

Nội dung bài tập cơ bản:

Bài 1: Quản lý các đối tượng trong một học viện:

- Nhân viên quản lý (mã nv, tên nv, trình độ, chuyên môn, lương cơ bản, phụ cấp chức vụ). Lương = lương cơ bản + phụ cấp chức vụ
- Nhân viên nghiên cứu (mã nv, tên nv, trình độ, chuyên môn, lương cơ bản, phụ cấp độc hại). Lương = lương cơ bản + phụ cấp độc hại
- Nhân viên phục vụ (mã nv, tên nv, trình độ, lương cơ bản). Lương = lương cơ bản

Khái quát hóa các lớp theo sơ đồ phân cấp kế thừa để xây dựng lớp trừu tượng và lớp cụ thể.

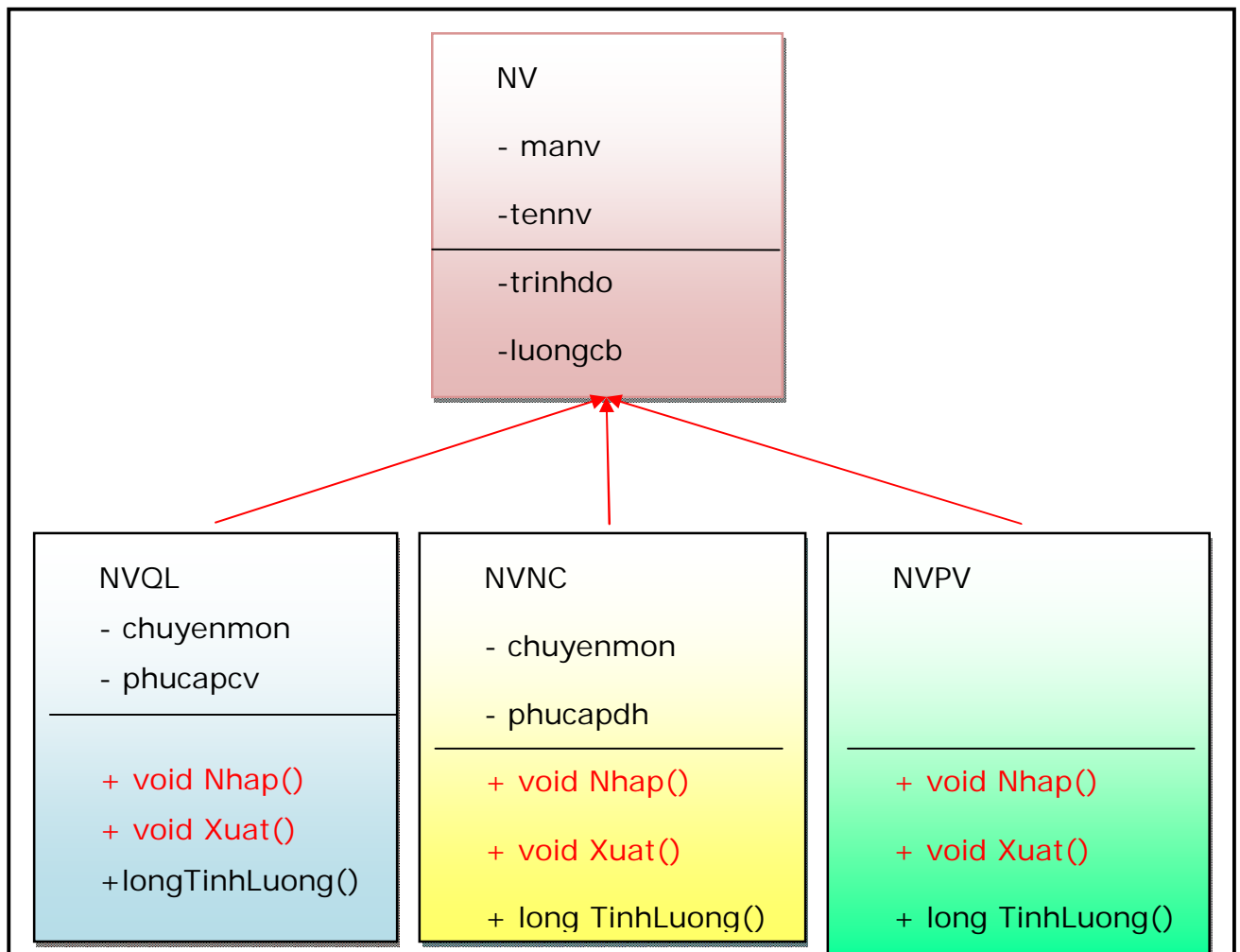
Xây dựng lớp thử nghiệm chứa phương thức main(), trong đó khai báo các đối tượng nhân viên, nhập dữ liệu và tính lương cho từng nhân viên.

Bài 2: Xây dựng lớp HOCVIEN (học viện) để quản lý danh sách nhân viên của học viện. Tính tổng lương của tất cả nhân viên có trong học viện.

Hướng dẫn

Bài 1:

1. Dựa trên các đặc điểm chung của các lớp đối tượng, ta có sơ đồ các lớp đối tượng với quan hệ kế thừa như sau (*khi làm 1 bài bất kỳ, các bạn phải tự xây dựng được sơ đồ lớp này*):



Trong các phương thức của lớp NV, phương thức nào là phương thức trừu tượng?

- Vì một nhân viên nói chung, ta chưa biết cách tính lương như thế nào nên phương thức tính lương là phương thức trừu tượng, do đó lớp NV là lớp trừu tượng.
- Tùy thuộc vào từng loại nhân viên cụ thể mà có cách tính lương khác nhau (code tưởng minh phương thức tính lương cho từng lớp con)

2. Xây dựng các lớp:

- **Lớp NV**

- Lớp NV là lớp trừu tượng
- Phương thức `Nhap()` dùng để nhập các thuộc tính mã nv, tên nv, trình độ, lương cb.

- Phương thức `Xuat()` dùng để xuất các thuộc tính mã nv, tên nv, trình độ, lương cb.
- Phương thức `TinhLuong()` là pt trừu tượng nên chỉ khai báo mà không cài đặt chi tiết.

- **Lớp NVQL**

- Lớp NVQL kế thừa từ lớp NV
- Trong phương thức `Nhap()`, gọi phương thức nhập của lớp cha và nhập cho các thuộc tính chuyên môn, phụ cấp chức vụ
- Phương thức `Xuat()` cũng gọi phương thức xuất của lớp cha và xuất các thuộc tính chuyên môn, phụ cấp chức vụ
- Phương thức `TinhLuong()`: viết code tương minh Lương = Lương cơ bản + phụ cấp chức vụ

- **Lớp NVNC**

- Tương tự lớp NVQL, nhưng Lương = Lương cơ bản + phụ cấp độc hại

- **Lớp NVPV**

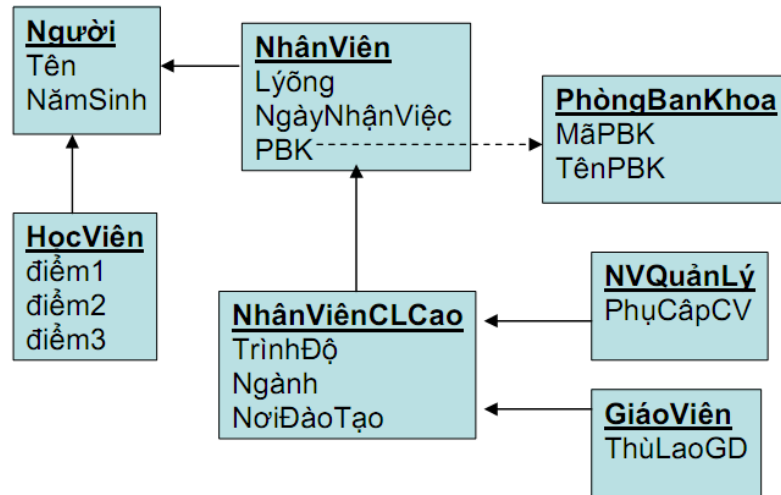
- Lớp NVPV kế thừa lớp NV, đặc biệt nó không có thêm thuộc tính nào khác so với lớp NV nên không cần viết lại phương thức `Nhap()`, `Xuat()`
- Chỉ viết lại phương thức `TinhLuong()`, viết code tương minh Lương = Lương cơ bản

3. Cài đặt lớp thử nghiệm: Tạo các đối tượng nhân viên khác nhau, nhập xuất dữ liệu cho các nhân viên này và tính lương cho từng nhân viên

Mở rộng

Tương tự, làm các bài sau:

Viết chương trình minh họa thiết kế sau:



Chương trình có giao diện như sau:

1. Nhập một học viên
2. Nhập một nhân viên quản lý
3. Nhập một giáo viên
4. Xuất thông tin một học viên
5. Xuất thông tin một nhân viên quản lý
6. Xuất thông tin một giáo viên

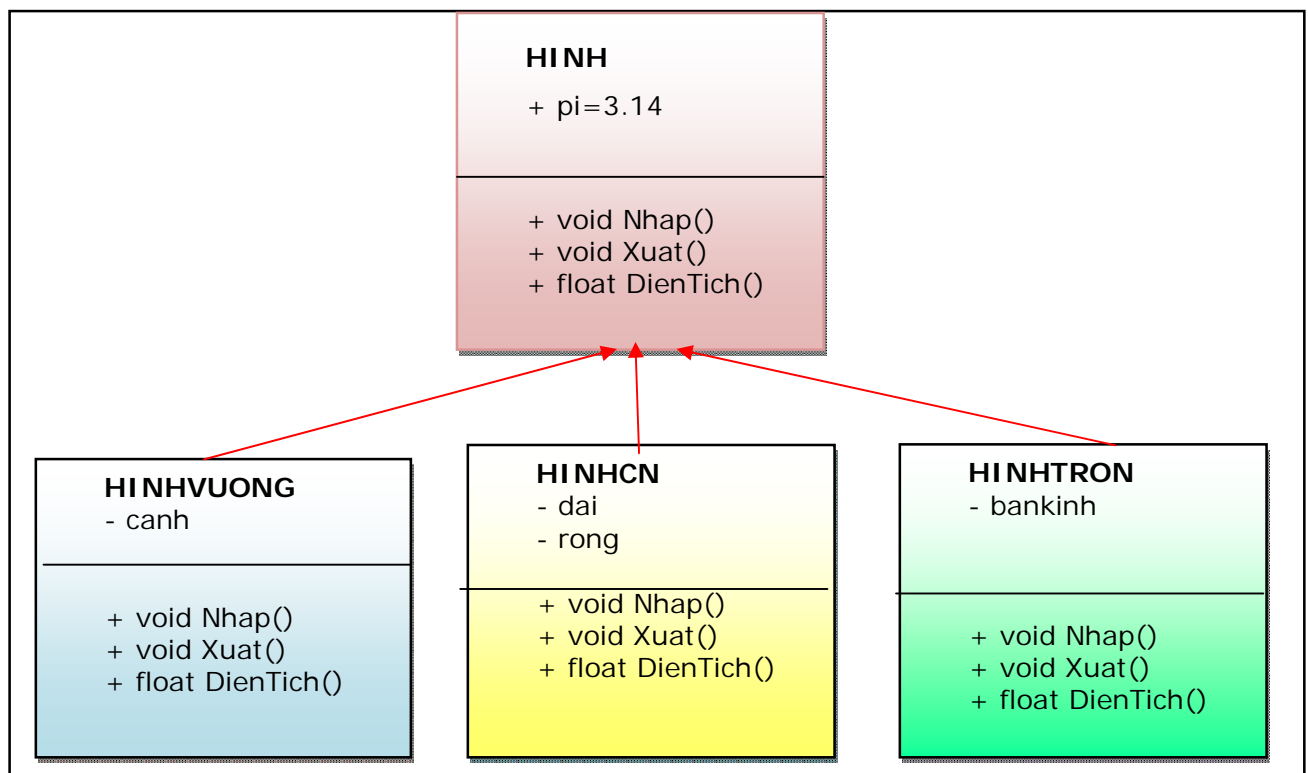
TUẦN 8: Interface, Package

Nội dung bài tập cơ bản:

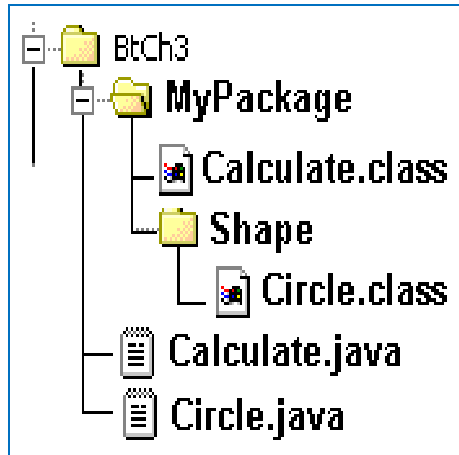
Bài 1: Cho sơ đồ phân cấp như hình bên dưới. Yêu cầu:

Xây dựng các lớp có thực hiện giao diện HINH.

Xây dựng lớp thử nghiệm chứa phương thức main(), trong đó khai báo một **mảng các đối tượng hình**, thể hiện **tính đa hình (polymorphism)** bằng cách cho phép lựa chọn nhập thông tin cho mỗi phần tử trong mảng là hình vuông, hình chữ nhật hay hình tròn, tính diện tích của mỗi hình trong mảng.



Bài 2: Xây dựng các gói theo thiết kế sau:



```
public class Calculate
{ public static double Volume(double l,
                               double w, double h)
    {return l*w*h;}
  public static double Add(double n1, double n2)
    { return n1+n2;}
}
```

```
public class Circle
{ double r;
  public Circle(double rr) { r=rr;}
  public double Circumference()
    { return 2*Math.PI*r;}
  public double Area() { return
    Math.PI* r*r; }
}
```

Hướng dẫn

Bài 1:

1. Vì mỗi loại hình có thành phần dữ liệu riêng, có diện tích được tính theo công thức khác nhau nên trong sơ đồ trên **HÌNH là một giao diện** chứa:
 - Thành phần dữ liệu là hằng số pi: **public float pi=3.14;**
 - Các phương thức chưa được cài đặt chi tiết. **Chi tiết các phương thức sẽ được cài ở từng lớp con cụ thể.**
2. Xây dựng lớp thử nghiệm chứa phương thức main():
 - Khai báo mảng các đối tượng hình
 - Thể hiện tính đa hình bằng cách cho phép người dùng lựa chọn sẽ nhập hình nào: *(xem tiếp trang sau)*.
 - Viết chương trình sử dụng 2 lớp trong 2 gói này

```
//nhập số lượng hình: Nhập n
//bạn tự code...

HINH ds[]=new HINH[n]; //khai báo mảng
//nhập danh sách hình
for(int i=0; i<n; i++) {
    System.out.println ("Chon loai hình se nhap:");
    System.out.println ("1: hình vuông");
    System.out.println ("2: hình chu nhật");
    System.out.println ("3: hình tròn");

    int chon=0;
    //nhập chọn lựa của người dùng...bạn tự code :D
    switch(chon){
        case 1: ds[i]=new HINHVUONG();
                ds[i].nhap();
                break;
        case 2: ds[i]=new HINHCHUNHAT();
                ds[i].nhap();
                break;
        case 3: ds[i]=new HINHTRON();
                ds[i].nhap();
                break;
        default: System.out.println ("Ban phai chon 1 trong 3 loai tren");
                break;
    }
}
for(int k=0; k<n; k++)
{
    ds[k].xuat();
    System.out.println ("Dien tich: " + ds[k].DienTich());
}
```

Bài 2: Tạo thư mục BTCh3.

Tạo 2 file source code: Calculate.java và Circle.java lưu trong thư mục BTCh3.

- Trong file Calculate.java, ta khai báo gói MyPackage bằng câu lệnh:
 - `package MyPackage;`
 - Như vậy khi biên dịch sẽ tạo ra file Calculate.class nằm trong thư mục MyPackage.
- Theo đề bài, ta thấy Shape là gói con của gói MyPackage, và gói Shape chứa file Circle.class. Do đó, trong file Circle.java, ta khai báo gói Shape bằng câu lệnh:
 - `package MyPackage.Shape;`
 - Như vậy, khi biên dịch sẽ tạo ra file Circle.class nằm trong thư mục Shape – là thư mục con của thư mục MyPackage.

- Viết chương trình sử dụng 2 lớp trong 2 gói này

```
TestMyPackage.java
1 import java.io.*;
2 import MyPackage.Calculate;
3 import MyPackage.Shape.Circle;
4 class TestMyPackage
5 { public static void main(String args[])
6   { System.out.println(Calculate.Volume(3,4,5));
7     Circle C= new Circle(5);
8     System.out.println(C.Area());
9   }
10 }
```


TUẦN 9: Exception, Garbage collection

Nội dung bài tập cơ bản:

1. Bẫy lỗi tổng quát. Tạo một file tên là Exception_1.java như sau:

```
Exception_1.java
1 // Exception_1.java
2 import javax.swing.*;
3 class Exception_1
4 { public static void main (String args[])
5 {   int n;
6     try
7     { n = Integer.parseInt(JOptionPane.showInputDialog(null, "Nhập số nguyên", ""));
8     }
9     catch (Exception e)
10    { System.out.println(e.toString());
11      e.printStackTrace();
12    }
13    finally
14    { System.out.println("Have good fun");
15    }
16 }
17 }
```

Chạy và xem kết quả. Giải thích kết quả: kết quả này do đâu gây ra, mỗi dòng kết quả xuất hiện là do dòng code nào xuất.

2. Bẫy lỗi cụ thể. Tạo một file Exception_2.java có nội dung như sau:

```
Exception_1.java
1 // Exception_1.java
2 import javax.swing.*;
3 class Exception_1
4 { public static void main (String args[])
5 {   int n;
6     try
7     { n = Integer.parseInt(JOptionPane.showInputDialog(null, "Nhập số nguyên", ""));
8     }
9     catch (java.lang.NumberFormatException e1)
10    { System.out.println(e1.toString());
11      e1.printStackTrace();
12    }
13    finally
14    { System.out.println("Have goo fun");
15    }
16 }
17 }
```

Chạy và xem kết quả. Giải thích kết quả: kết quả này do đâu gây ra, mỗi dòng kết quả xuất hiện là do dòng code nào xuất.

Hãy so sánh bẫy lỗi tổng quát với lỗi cụ thể. Lợi, hại thế nào ?

3. Lan truyền lỗi. Tạo một file tên là Exception_3.java như sau:

Exception_3.java

```

1 // Exception_3.java
2 import javax.swing.*;
3 class Exception_3
4 { static int getPos (int i, int a[])
5   { return a[i];
6   }
7   static void output(int a[], int n)
8   { for (int i=0;i<n; i++)
9     System.out.println(getPos(i,a));
10  }
11 public static void main (String args[])
12 {   int a[]={ 4, 7, 9, 12, 7 };
13     output (a,10);
14 }
15 }

```

Chạy và xem kết quả. Giải thích kết quả: kết quả này do đâu gây ra, mỗi dòng kết quả xuất hiện là do dòng code nào xuất. Từ đó vẽ lại mô hình lan truyền lỗi trong chương trình này.

4. Chặn lan truyền lỗi. Tạo một file Exception_4.java có nội dung như sau:

Exception_3.java

```

1 // Exception_3.java
2 import javax.swing.*;
3 class Exception_3
4 { static int getPos (int i, int a[])
5   { return a[i];
6   }
7   static void output(int a[], int n)
8   { try
9     { for (int i=0;i<n; i++)
10      System.out.println(getPos(i,a));
11    }
12    catch (java.lang.ArrayIndexOutOfBoundsException e)
13    { System.out.println("Out of bound!");
14    }
15  }
16 public static void main (String args[])
17 {   int a[]={ 4, 7, 9, 12, 7 };
18     output (a,10);
19     System.out.println("Have good fun!");
20 }
21 }

```

Chạy và xem kết quả. Giải thích kết quả: kết quả này do đâu gây ra, mỗi dòng kết quả xuất hiện là do dòng code nào xuất. Từ đó vẽ lại mô hình lan truyền lỗi trong chương trình này.

5. Định nghĩa Exception. Tạo một file tên là Exception_7.java như sau:

```
UserException.java
1 // User-Defined Exception Demonstration
2 class MyException extends java.lang.ArrayIndexOutOfBoundsException
3 { MyException()
4   { super ("Truy xuất ngoài tam pha song!");
5   }
6 }
7 class MyExceptionDemo
8 { static int getPos(int i, int a[]) throws MyException
9   { if (i<0 || i>=a.length) throw new MyException();
10    return a[i];
11  }
12 public static void main (String args[])
13 { int a[] = { 4,2,8,0,9 };
14   System.out.println(getPos(20,a));
15 }
16 }
```

Chạy và xem kết quả. Giải thích kết quả.

Bài tập mở rộng

Tạo lớp mô tả và thao tác trên mảng số nguyên có quản lý lỗi truy cập.

Các phương thức:

- Constructor chuẩn (cấp phát mảng có tối đa MaxN phần tử)
- Constructor có tham số là mảng các số int (sao chép mảng đã có)
- Nhập mảng
- Xuất mảng
- Sắp xếp mảng tăng dần
- Chèn x vào vị trí thứ i trong mảng (x, i nhập từ bàn phím)

TÀI LIỆU THAM KHẢO

1. Đoàn Văn Ban (2005). *Lập trình hướng đối tượng*. NXB Khoa học và Kỹ thuật, Hà Nội.
2. Hoàng Trọng Thế (2012). *Lập trình hướng đối tượng trong Java*. Đại học Sư phạm kỹ thuật Hưng Yên.
3. Trần Minh Châu, Nguyễn Việt Hà. *Giáo trình Lập trình hướng đối tượng với Java*. Đại học Công nghệ Hà Nội.
4. Deitel & Deitel (2012). *Java How to program*. Prentice – Hall.
5. Kathy Sierra, Bert Bates (2008). *Head First Java*. O'Reilly.