📌 **Danh sách từ khóa lý thuyết thường gặp trong đề thi OOP Java**

Từ khóa	Mô tả/Ý nghĩa
class	Khai báo lớp trong Java
extends	Kế thừa lớp cha trong Java
@Override	Ghi đè phương thức của lớp cha
private, public, protected	Các phạm vi truy cập của thuộc tính/phương thức (Encapsulation)
get/set (getter/setter)	Các phương thức để truy xuất hoặc cập nhật giá trị thuộc tính
constructor	Hàm khởi tạo đối tượng
overload (Overloading)	Nạp chồng phương thức: cùng tên, khác tham số
override (Overriding)	Ghi đè phương thức của lớp cha
super	Gọi đến constructor hoặc phương thức của lớp cha
ArrayList	Lưu trữ danh sách đối tượng động
Polymorphism	Tính đa hình: một hành vi có thể có nhiều hình thức khác nhau
Encapsulation	Tính đóng gói: che giấu thông tin bên trong lớp
Inheritance	Tính kế thừa: lớp con kế thừa thuộc tính/phương thức từ lớp cha
Abstraction	Tính trừu tượng: ẩn chi tiết, chỉ giữ lại phần quan trọng
Scanner	Lớp dùng để nhập dữ liệu từ bàn phím (thường dùng trong thực hành)
main method	Điểm bắt đầu của chương trình Java
System.out.println()	Xuất dữ liệu ra màn hình

📘 CÂU HỎI LÝ THUYẾT DẠNG TỰ LUẬN (LẤY VÍ DỤ TỪ PHẦN 2)

🔷 Câu 1. class

(1.0 điểm)

Để tạo một lớp các đối tượng trong Java dùng từ khóa nào? Cho ví dụ tạo lớp BacSi có các thuộc tính maBS, hoTen, mucLuong.

Gợi ý: lớp BacSi trong phần mềm quản lý bệnh viện.

🔷 Câu 2. extends

(1.0 điểm)

Trong Java, để kế thừa một lớp cha ta dùng từ khóa gì? Cho ví dụ lớp BSToanTG kế thừa từ lớp BacSi.

Gợi ý: BSToanTG extends BacSi trong bài quản lý bác sĩ toàn thời gian.

🔷 Câu 3. @Override

(1.0 điểm)

Chú thích @Override dùng để làm gì trong Java? Trong ví dụ sau, chỉ ra phương thức nào được ghi đè:

```
public class BacSi {
    public double luongHangThang() {
        return mucLuong;
    }
}

public class BSToanTG extends BacSi {
    @Override
    public double luongHangThang() {
        return getMucLuong() + luongKham + luongXetNghiem;
    }
}
```

🔷 Câu 4. private, public, protected (Encapsulation)
(1.0 điểm)
Tính đóng gói trong Java được thể hiện qua các từ khóa nào? Cho ví dụ trong lớp BenhNhan với thuộc tính private và phương thức public.
Gợi ý: lớp BenhNhan và các lớp kế thừa trong phần mềm quản lý bệnh nhân.

🔷 Câu 5. get/set
(1.0 điểm)
Tại sao nên sử dụng các phương thức getter/setter trong Java? Viết ví dụ cho lớp BacSi sử dụng getHoTen() và setHoTen().

🔷 Câu 6. constructor
(1.0 điểm)
Hàm khởi tạo (constructor) trong Java là gì? Có thể có nhiều constructor trong một lớp không? Cho ví dụ lớp BNNgoaiTru có constructor dùng super(...) để khởi tạo lớp cha BenhNhan.

🔷 Câu 7. overload (Overloading)
(1.0 điểm)
Nạp chồng phương thức là gì? Trong ví dụ sau, chỉ ra phương thức nào được nạp chồng:

```
public class BacSi {
    public void hienThi() {
        System.out.println("Bac si thuong");
    }

    public void hienThi(String chucVu) {
        System.out.println("Bac si " + chucVu);
    }
}
```

🔷 Câu 8. override (Overriding)
(1.0 điểm)
Ghi đè phương thức là gì? Trong bài toán quản lý bác sĩ, hãy nêu phương thức được ghi đè trong lớp BSBanTG.

🔷 Câu 9. super
(1.0 điểm)
Từ khóa super có tác dụng gì? Viết đoạn mã tạo constructor cho lớp BNNoiTru sử dụng super() để gọi constructor của lớp BenhNhan.

🔷 Câu 10. ArrayList
(1.0 điểm)
ArrayList dùng để làm gì trong Java? Hãy viết đoạn mã khởi tạo danh sách ArrayList<BacSi> chứa 2 đối tượng BSToanTG.

🔷 Câu 11. Polymorphism
(1.0 điểm)
Tính đa hình là gì? Cho ví dụ đoạn mã có sử dụng:
BacSi bs = new BSToanTG(...);
System.out.println(bs.luongHangThang());
Giải thích vì sao lời gọi phương thức ở dòng trên thể hiện tính đa hình.

🔷 Câu 12. Encapsulation
(1.0 điểm)
Trình bày mục tiêu và lợi ích của tính đóng gói (Encapsulation). Dựa vào ví dụ lớp BenhNhan, chỉ ra cách ẩn thông tin và truy cập dữ liệu an toàn.

🔷 Câu 13. Inheritance
(1.0 điểm)
Thế nào là kế thừa trong lập trình hướng đối tượng? Cho ví dụ lớp BNNoiTru kế thừa lớp BenhNhan.

🔷 Câu 14. Abstraction
(1.0 điểm)
Tính trừu tượng trong Java là gì? Nếu bạn thiết kế một lớp Nguoi với phương thức trienKhaiVienPhi() mà các lớp con như BNNgoaiTru, BNNoiTru phải định nghĩa lại, bạn sẽ sử dụng kiểu lớp như thế nào?

🔷 Câu 15. Scanner

(1.0 điểm)
Lớp Scanner dùng để làm gì trong Java? Viết chương trình nhập từ bàn phím thông tin một bác sĩ (họ tên, mức lương) và in ra lương tháng.

🔷 Câu 16. main method
(1.0 điểm)
Phương thức main có vai trò gì trong chương trình Java? Viết phương thức main cho lớp App để tạo 1 đối tượng BacSi và in ra lương.

🔷 Câu 17. System.out.println()
(1.0 điểm)
System.out.println() dùng để làm gì? Viết đoạn mã in ra kết quả lương của 1 bác sĩ từ phương thức luongHangThang().

🔷 Câu 1. class
Trả lời:
Từ khóa class dùng để khai báo một lớp trong Java.
Ví dụ:
```
public class BacSi {
    private String maBS;
    private String hoTen;
    private double mucLuong;
}
```

🔷 Câu 2. extends
Trả lời:
Từ khóa extends dùng để khai báo một lớp con kế thừa từ một lớp cha.
Ví dụ:
```
public class BSToanTG extends BacSi {
    // Thuộc tính và phương thức riêng của BSToanTG
}
```

🔷 Câu 3. @Override
Trả lời:
Chú thích @Override dùng để chỉ ra rằng một phương thức đang ghi đè (override) phương thức của lớp cha.
Phương thức ghi đè trong ví dụ là:
```
@Override
public double luongHangThang() {
    return getMucLuong() + luongKham + luongXetNghiem;
}
```

🔷 Câu 4. private, public, protected (Encapsulation)

Trả lời:
Tính đóng gói thể hiện qua việc sử dụng các phạm vi truy cập:

- private: chỉ truy cập trong lớp
- public: truy cập từ bất kỳ đâu
- protected: truy cập trong cùng package và lớp con

Ví dụ:

```
public class BenhNhan {
    private String hoTen;
    public String getHoTen() {
        return hoTen;
    }
    public void setHoTen(String hoTen) {
        this.hoTen = hoTen;
    }
}
```

🔷 Câu 5. get/set

Trả lời:
Getter/setter giúp truy cập và cập nhật dữ liệu an toàn khi thuộc tính để ở chế độ private.
Ví dụ:

```
public class BacSi {
    private String hoTen;
    public String getHoTen() {
        return hoTen;
    }
    public void setHoTen(String hoTen) {
        this.hoTen = hoTen;
    }
}
```

🔷 Câu 6. constructor

Trả lời:
Constructor là phương thức đặc biệt dùng để khởi tạo đối tượng. Có thể có nhiều constructor (nạp chồng).
Ví dụ:

```
public class BenhNhan {
    private String hoTen;
    public BenhNhan(String hoTen) {
        this.hoTen = hoTen;
    }
}
public class BNNgoaiTru extends BenhNhan {
```

```
    public BNNgoaiTru(String hoTen) {
        super(hoTen);
    }
}
```

🔷 Câu 7. overload (Overloading)

Trả lời:
Overloading là việc định nghĩa nhiều phương thức cùng tên trong một lớp nhưng khác tham số.
Hàm bị nạp chồng:

```
public void hienThi() {...}
public void hienThi(String chucVu) {...}
```

🔷 Câu 8. override (Overriding)

Trả lời:
Overriding là việc lớp con định nghĩa lại phương thức đã có trong lớp cha.
Ví dụ lớp BSBanTG ghi đè:

```
@Override
public double luongHangThang() {
    return getMucLuong()*0.3 + luongNgay * soNgayLamViec;
}
```

🔷 Câu 9. super

Trả lời:
super dùng để gọi constructor hoặc phương thức của lớp cha.
Ví dụ:

```
public class BNNoiTru extends BenhNhan {
    public BNNoiTru(String hoTen) {
        super(hoTen);
    }
}
```

🔷 Câu 10. ArrayList

Trả lời:
ArrayList dùng để tạo danh sách động các phần tử.
Ví dụ:

```
ArrayList<BacSi> ds = new ArrayList<>();
ds.add(new BSToanTG("BS01", "Nguyen Van A", 1000, 500, 300));
ds.add(new BSToanTG("BS02", "Tran Thi B", 1100, 400, 250));
```

🔷 Câu 11. Polymorphism

Trả lời:
Đa hình cho phép đối tượng lớp con có thể được tham chiếu bằng lớp cha và

phương thức được gọi tùy vào đối tượng thực tế.
Ví dụ:
BacSi bs = new BSToanTG("BS01", "Van A", 1000, 500, 300);
System.out.println(bs.luongHangThang());
Mặc dù bs có kiểu BacSi, phương thức luongHangThang() của BSToanTG vẫn được gọi → thể hiện **đa hình động (runtime polymorphism)**.

◆ Câu 12. Encapsulation
Trả lời:
Mục tiêu: bảo vệ dữ liệu, kiểm soát truy cập.
Ví dụ:
```
public class BenhNhan {
    private double vienPhi;

    public double getVienPhi() {
        return vienPhi;
    }

    public void setVienPhi(double vienPhi) {
        this.vienPhi = vienPhi;
    }
}
```

◆ Câu 13. Inheritance
Trả lời:
Kế thừa là tính chất cho phép lớp con sử dụng lại thuộc tính/phương thức của lớp cha.
Ví dụ:
```
public class BNNoiTru extends BenhNhan {
    // kế thừa thuộc tính từ BenhNhan
}
```

◆ Câu 14. Abstraction
Trả lời:
Tính trừu tượng là che giấu chi tiết, chỉ hiển thị hành vi bên ngoài.
Sử dụng lớp trừu tượng (abstract class):
```
public abstract class Nguoi {
    public abstract double trienKhaiVienPhi();
}
public class BNNoiTru extends Nguoi {
    @Override
    public double trienKhaiVienPhi() {
        return ...;
```

```
    }
}
```

🔷 Câu 15. Scanner

Trả lời:

Lớp Scanner giúp nhập dữ liệu từ bàn phím.

Ví dụ:

```
import java.util.Scanner;

public class App {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Nhap ten bac si: ");
        String hoTen = sc.nextLine();
        System.out.print("Nhap muc luong: ");
        double mucLuong = sc.nextDouble();

        BacSi bs = new BacSi("BS01", hoTen, mucLuong);
        System.out.println("Luong thang: " + bs.luongHangThang());
    }
}
```

🔷 Câu 16. main method

Trả lời:

main() là điểm bắt đầu của chương trình Java.

Ví dụ:

```
public class App {
    public static void main(String[] args) {
        BacSi bs = new BacSi("BS01", "Nguyen Van A", 1000);
        System.out.println("Luong: " + bs.luongHangThang());
    }
}
```

🔷 Câu 17. System.out.println()

Trả lời:

Dùng để in dữ liệu ra màn hình console.

Ví dụ:

```
BacSi bs = new BacSi("BS01", "Nguyen Van A", 1000);
System.out.println("Luong cua bac si: " + bs.luongHangThang());
```
