NAMA : FARZAD SHINAVI PASARIBU
NIM : 4124550009
KELAS: PSIK 24A
MATKUL: PBO (UTS)

1) jelaskan prinsip OOP (Encapsulation, Inheritance, Polymorphism, Abstraction) beserta analogi nyata

* Encapsulation: Mengembunyikan data, hanya bisa diakses melalui method tertentu.
  Contoh nyata: mesin ATM, kita tidak tahu proses internalnya, hanya boleh antarmuka.
* Inheritance : Pewarisan sifat dari class induk ke anak class.
  contoh nyata: Anak mewarisi fitur orang tuanya.
* Polymorphism: Satu fungsi bisa berbeda perilaku tergantung objek.
  Contoh nyata: Tombol "print" bisa mencetak dokumen atau gambar tergantung objek.
* Abstraction: Mengembunyikan detail kompleks, hanya menampilkan yg penting
  Contoh nyata: Mengendarai mobil tanpa tahu cara kerja mesinnya.

2) kelebihan java versi terbaru dalam konteks pengembangan berbasis OOP

* Record Pattern : - memudahkan ekstraksi nilai-nilai objek record tanpa banyak boiler plate code.
  - Mempermudah pembacaan data dari objek tanpa getNama() getUmur() secara eksplisit.
* Sealed class : - membatasi pewarisan class agar hanya class tertentu yg bisa mengasi sub class, menjaga arsitektur OOP tetap terkontrol.

3) Perbedaan class dan object beserta contohnya

: * class adalah blueprint/template yang mendefinisikan variabel atau method dan hanya dibuat sekali namun bisa di pakai berkali-kali. contoh: Formulir pendaftaran mahasiswa.
* object adalah instansi nyata yang dibuat dari class, setiap object punya nilai untuk atribut nya. contoh: Formulir yg sudah diisi data tertentu

Class	Object
class Mahasiswa {	// mahasiswa mhs1 = new mahasiswa();
String nama;	mhs1.nama = "Farzad";
int nim;	mhs2.nama = "4124550009";
}	

4) Encapsulation pada class BankAccount
gunakan atribut balance sebagai private dan akses dengan getBalance()/deposit()/withdraw(). ini mencegah perubahan langsung balance sembarangan. Encapsulation penting untuk menjaga data sensitif dan mencegah kesalahan atau manipulasi.

Contoh program

```
Public Class BankAccount {
  Private double balance;

  Public BankAccount(double initialBalance) {
        balance : initialBalance;
  }
  Public void deposit(double amount) {
     if (amount > 0) balance += amount;
  }
  Public void withdraw(double amount) {
   if (amount > 0 && amount <= balance) balance -= amount;
  }
  Public double getBalance() {
     return balance;
  }
}
```

5) Bagaimana mekanisme constructor chaining bekerja pada pewarisan java. Apa yg terjadi jika constructor pada superclass tidak dipanggil secara eksplisit.
: Constructor chaining terjadi saat constructor class menggunakan keyword super(). jika tidak dipanggil secara eksplisit, java otomatis memanggil constructor default superclass (tanpa parameter). jika superclass tidak memiliki constructor default, maka akan terjadi error kompilasi.

6) jelaskan bagaimana penggunaan interface mendukung konsep polymorphism, dan berikan contoh penggunaanya dalam sistem pemesanan makanan online
: interface mendukung polymorphism karena memungkinkan berbagai class berbeda mengimplementasikan method yang sama dengan cara masing-masing.

```
interface Bayar {void bayar();}

Class Gopay implements Bayar {
```

```
Public void bayar() { System.out.println("Bayar dengan GoPay"); }
}
Public class Main {
    Public static void main (String[] args) {
    Bayar b = new GoPay();
     b.bayar();
  }
}
```

7) Perbandingan abstract class, interface, sealed class:

Abstract class : Digunakan jika ada logika umum

Interface class : Cocok untuk kontrak tanpa implementasi awal

Sealed class : Membatasi subclass hanya pada class tertentu. Contoh: untuk tipe Pengguna (Admin, User, Guest) gunakan sealed class. Untuk sistem pembayaran gunakan interface. Gunakan abstract class untuk kendaraan yg berbagi logika dasar.