Programowanie obiektowe w JAVA

- Klasy i obiekty
- Hermetyzacja
- Dziedziczenie
- Polimorfizm
- Kompozycja

Klasy i obiekty

Klasa to szablon (plan) dla tworzenia obiektów. Definiuje atrybuty (zmienne) oraz metody (funkcje), które mogą być używane przez obiekty tej klasy.

Obiekt to instancja klasy. Tworzymy go za pomocą słowa kluczowego **new.**

```java
public class Main {
    public static void main(String[] args) {
        // Tworzenie obiektu klasy Osoba
        Osoba osoba1 = new Osoba("Jan", 25);
        osoba1.przedstawSie();  // Wywołanie metody na obiekcie
    }
}
```

Przykład klasy w Java:

```java
public class Osoba {
    // Atrybuty
    private String imie;
    private int wiek;

    // Konstruktor
    public Osoba(String imie, int wiek) {
        this.imie = imie;
        this.wiek = wiek;
    }

    // Metoda
    public void przedstawSie() {
        System.out.println("Cześć, mam na imię " + imie + " i mam " + wiek + " lat.");
    }
}
```

Hermetyzacja

Hermetyzacja to zasada polegająca na ukrywaniu wewnętrznych szczegółów implementacji obiektu oraz udostępnianiu tylko niezbędnych metod. Dzięki temu zapewniamy większą kontrolę nad danymi oraz zapobiegamy ich niekontrolowanemu modyfikowaniu.

Modyfikatory dostępu:

- **private**
- **public**
- **protected**

Przykład Hermetyzacji w Java:

```java
public class Osoba {
    private String imie;  // Ukryty atrybut
    private int wiek;     // Ukryty atrybut

    // Konstruktor
    public Osoba(String imie, int wiek) {
        this.imie = imie;
        this.wiek = wiek;
    }

    // Getter dla imienia
    public String getImie() {
        return imie;
    }

    // Setter dla wieku
    public void setWiek(int wiek) {
        if (wiek > 0) {  // Walidacja wieku
            this.wiek = wiek;
        }
    }

    public void przedstawSie() {
        System.out.println("Cześć, mam na imię " + imie + " i mam " + wiek + " lat.");
    }
}
```

Dziedziczenie

Dziedziczenie to mechanizm, który pozwala na tworzenie nowych klas na podstawie istniejących.

Klasa dziedzicząca (subklasa) przejmuje **właściwości** i **metody** klasy bazowej (superklasy), a także może dodawać własne elementy lub nadpisywać istniejące metody.

Zalety dziedziczenia:

- **Zwiększenie spójności plikacji**

Przykład Dziedziczenia w Java:

```java
// Klasa bazowa
public class Zwierze {
    public void dzwiek() {
        System.out.println("Zwierzę wydaje dźwięk.");
    }
}

// Klasa dziedzicząca
public class Pies extends Zwierze {
    @Override
    public void dzwiek() {
        System.out.println("Pies szczeka.");
    }
}

public class Main {
    public static void main(String[] args) {
        Pies pies = new Pies();
        pies.dzwiek();  // Wywołanie nadpisanej metody
    }
}
```

Polimorfizm

Polimorfizm pozwala na używanie obiektów różnych klas w taki sposób, jakby były obiektami jednej klasy bazowej.

Może to odbywać się przez nadpisywanie metod (metody w klasach dziedziczących) lub przeciążanie metod (ta sama metoda o różnych sygnaturach).

Przykład Poliforizmu w Java:

```java
public class Zwierze {
    public void dzwiek() {
        System.out.println("Zwierzę wydaje dźwięk.");
    }
}

public class Pies extends Zwierze {
    @Override
    public void dzwiek() {
        System.out.println("Pies szczeka.");
    }
}

public class Kot extends Zwierze {
    @Override
    public void dzwiek() {
        System.out.println("Kot miauczy.");
    }
}

public class Main {
    public static void main(String[] args) {
        Zwierze zwierze1 = new Pies();
        Zwierze zwierze2 = new Kot();

        zwierze1.dzwiek();  // Pies szczeka
        zwierze2.dzwiek();  // Kot miauczy
    }
}
```

Kompozycja

Kompozycja to relacja między obiektami, w której jeden obiekt zawiera instancję innych obiektów.

Kompozycja jest silniejszą formą relacji niż dziedziczenie, ponieważ oznacza, że obiekt A jest częścią obiektu B i nie może istnieć bez niego.

Zalety kompozycji:

- **Elastyczność i niezależność obiektów**
- **Modułowość aplikacji**
- **Ułatwione rozbudowywanie kodu**

Przykład Kompozycji w Java:

```java
public class Silnik {
    public void uruchom() {
        System.out.println("Silnik uruchomiony.");
    }
}

public class Samochod {
    private Silnik silnik;  // Kompozycja

    public Samochod() {
        silnik = new Silnik();  // Tworzenie obiektu Silnik w konstruktorze
    }

    public void uruchomSamochod() {
        silnik.uruchom();
        System.out.println("Samochód rusza.");
    }
}

public class Main {
    public static void main(String[] args) {
        Samochod samochod = new Samochod();
        samochod.uruchomSamochod();
    }
}
```