تازيم ةغل افاج

1. الصفوف

- ام وه: الصف هو الوحدة الأساسية لبناء □□□□، ويعمل كقالب أو مخطط لإنشاء الكائنات. يضم البيانات □الحقول□ والسلوك □الطرق□ التي تحدد خصائص الكائنات وأفعالها.
- اذامل وه مهم: الصفوف هي حجر الزاوية في نموذج برمجة □□□□ للأشياء الموجهة □□□□□. تروج للتكثيف تجميع□ البيانات، والطرق□، إخفاء والتجريد تفاصيل□ التنفيذ□، والتقسيم، مما يجعل الكود قابل لإعادة الاستخدام وسهل الصيانة.
- فيك متي هماد‌ختسا: تنمذج الصفوف الكيانات في البرنامج مثل Person، Vehicle، أو BankAccount. يمكن أن تشمل حقول مع محددات الوصول □public، private□، وطرق لتعديل حالة الكائن. مصنعين،
- الغوص العميق:
  - يمكن أن تكون الصفوف متداخلة الصفوف□ □الداخلية□ أو مجردة الا□ يمكن أن تكون مستنثنة. □مباشرة
  - تدعم الوراثة، مما يسمح لصف أن يمدد آخر ويورث خصائصه وطرقه.
- لاثم:

```java
public class Student {
    private String name;  //
    private int age;

    //
    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    //
    public void displayInfo() {
        System.out.println("Name: " + name + ", Age: " + age);
    }
}
```

- مادختسا يقيقح: يمكن أن تكون Student جزءًا من نظام إدارة المدرسة، مع طرق لحساب الدرجات أو تتبع الحضور.

---

2. الأشياء

- ام وه: الكائن هو مثال من الصف، ينشئ باستخدام كلمة new. يمثل تحقيقًا محددًا بالقالب مع حالة خاصة به.

- **لماذا هو مهم**: الأنظمة بتنمذج تسمح فريدة. بيانات مع الأمثلة بتعدد يسمح مما الحياة، إلى الصفوف تجلب الأشياء الحقيقية. الكيانات تمثيل طريق عن المعقدة
- **كيف يتم استخدامه**: Student student1 = المثال، سبيل على وحقولها. طرقها عبر معالجتها وتتم تُنشأ الكائنات new Student("Alice", 20); يخلق كائنًا من Student.
- **الغوص العميق**:
  - المتغيرات. في لها مراجع وتخزن العشوائية، الذاكرة في تُخزن الكائنات
  - المراجع. جميع عبر انعكاسها يتم الكائن حالة في التغييرات أن يعني مما الكائنات، تمرير عند مرجعًا تستخدم ▯▯▯▯
- **مثال**:

```
Student student1 = new Student("Alice", 20);
student1.displayInfo();  // Output: Name: Alice, Age: 20
```

- **حقيقي استخدام**: العناصر أو الفردية المشتريات تمثل Product أو Order مثل الكائنات إلكترونية، تجارة نظام في للبيع.

---

3. الطرق

- **ما هو**: تؤدي أو القيم، وتعيد متغيرات، تأخذ أن يمكن الكائنات. سلوك تحدد التي الصف داخل الكود من كتل هي الطرق أفعالًا.
- **لماذا هو مهم**: مع للتفاعل الرئيسية الطريقة هي للكود. القراءة قابلية وتحسين التكرار، تقليل منطقًا، تجلب الطرق الكائن. حالة
- **كيف يتم استخدامه**: بالطريقة يبدأ ▯▯▯▯ تطبيق كل الصفوف. على ثابت بشكل أو الكائنات على تُستدعي الطرق public static void main(String[] args).
- **الغوص العميق**:
  - الفرعي▯. الصف في ▯تعديل مغطاة أو مختلفة▯ متغيرات واحد، ▯اسم الأشكال متعدد تكون أن يمكن الطرق
  - الكائن▯. ▯مستوى مستنثنة أو الصف▯ ▯مستوى static تكون أن يمكن
- **مثال**:

```
public class MathUtils {
    public int add(int a, int b) {
        return a + b;
    }

    public double add(double a, double b) {  //
        return a + b;
```

```
    }
}
//
MathUtils utils = new MathUtils();
System.out.println(utils.add(5, 3));       // Output: 8
System.out.println(utils.add(5.5, 3.2));  // Output: 8.7
```

- ماذختسا يقيقح: طريقة withdraw في BankAccount يمكن أن تحديث رصيد الحساب وتسجيل المعاملة.

---

4. المتغيرات

- ام وه: المتغيرات تخزن قيم البيانات ويجب أن تُعلن مع نوع محدد مثل□ int، String، double□.
- اذامل وه مهم: المتغيرات هي مكان الحفظ في الذاكرة للبيانات في البرنامج، مما يتيح إدارة الحالة والحساب.
- فيك متي هماذختسا: □□□□ لديها أنواع متغيرات مختلفة:
  - المتغيرات المحلية: تُعلن داخل الطرق، مع نطاق محدود لتلك الطريقة.
  - متغيرات المثال: تُعلن في الصف، مرتبطة بكل كائن.
  - متغيرات ثابتة: تُعلن مع static، مشاركة بين جميع أمثلة الصف.
- الغوص العميق:
  - المتغيرات لديها قيم افتراضية مثل□ 0 لـ int، null للكائنات□ فقط تهيئتها يتم لم إذا للمتغيرات المستنثنة□الثابتة□.
  - □□□□ تقيد التحديد القوي، مما يمنع التخصيص غير المتوافق دون تحويل صريح.
- مثال:

```
public class Counter {
    static int totalCount = 0;   //
    int instanceCount;           //

    public void increment() {
        int localCount = 1;      //
        instanceCount += localCount;
        totalCount += localCount;
    }
}
```

- ماذختسا يقيقح: عدد تتبع المستخدمين المتصلين □ثابت□ مقابل أوقات الجلسة الفردية□ .مثال□

---

5. رماوأ قفدت مكحتال

- ام وه: رماوأ قفدت مكحتال ددحت راسم ذيفنت، جمانربال امب يف كلذ ةيطرشال if، else، switch والدورات for، while، do-while.

- اذامل وه مهم: حمست رايتخالاب، راركتالو امهو نايرورض ذيفنتل قطنم .دقعم

- فيك متي همادختسا:

  - ةيطرشال: ذفنت دوكال ءانب ىلع طورشال .ةيقطنمال
  - تارودال: رركت ىلع تانايبال وأ رركت لاعفألا ىتح متي ققحت .طرش

- صوغال قيمعال:

  - switch معدي String ذنم 7 و enum ةفاضإالاب ىلإ عاونألا .ةيلوألا
  - تارودال نكمي نأ نوكت، ةلخادتم ليدعتو breakcontinue .اهكولس

- لاثم:

```java
int score = 85;
if (score >= 90) {
    System.out.println("A");
} else if (score >= 80) {
    System.out.println("B");
} else {
    System.out.println("C");
}

for (int i = 0; i < 3; i++) {
    System.out.println("Loop iteration: " + i);
}
```

- مادختسا يقيقح: ةجلاعم ةمئاق نم تابلطال for قيبطتو تاموصخال ءانب ىلع غلبمال يلامجإال if.

---

6. ةهجاوال

- ام وه: ةهجاوال يه دقع ددحي قرطال يتال بجي نأ اهددحت فوفصال يتال .اهذفنت معدت ديرجتال ثيروتالو .ددعتمال

- اذامل وه مهم: تاهجاوال حيتت طابترالا فيعضال ددعتالو، لاكشألا امم حمسي فوفصل ةفلتخم نأ كراشت ةهجاو .ةكرتشم

- فيك متي همادختسا: فوفصال ذفنت تاهجاوال مادختساب ةملك implements. ذنم 8، نكمي نأ يوتحت تاهجاوال ىلع قرط ةيضارتفا ةتباثو عم .ذيفنت

- صوغال قيمعال:

- الطرق الافتراضية تسمح بتطور الواجهات مع الحفاظ على التوافق العكسي.
- الواجهات الوظيفية مع□ طريقة مجردة □واحدة هي مفتاح للتعبيرات اللامبدية.

- مثال:

```java
public interface Vehicle {
    void start();
    default void stop() {  //
        System.out.println("Vehicle stopped");
    }
}

public class Bike implements Vehicle {
    public void start() {
        System.out.println("Bike started");
    }
}
//
Bike bike = new Bike();
bike.start();  // Output: Bike started
bike.stop();   // Output: Vehicle stopped
```

- **تحقيق استخدام**: واجهة `Payment` لـ `CreditCard` و `PayPal` في نظام بوابة الدفع.

---

7. معالجة الاستثناءات

- **ما هو**: معالجة الاستثناءات تدير الأخطاء في وقت التشغيل باستخدام `try`، `catch`، `finally`، `throw`، و `throws`.
- **لماذا هو مهم**: تضمن الاستقرار عن طريق منع التوقفات وتسمح بالاستعادة من الأخطاء مثل ملف غير موجود أو القسمة على الصفر.
- **كيف يتم استخدامه**: الكود المخاطر يذهب في كتلة `try`، الاستثناءات المحددة يتم القبض عليها في كتلة `catch`، و `finally` تنفذ الكود التنظيف.
- **الغوص العميق**:
  - الاستثناءات هي كائنات مشتقة من `Throwable` □`Error` أو `Exception`□.
  - يمكن إنشاء استثناءات مخصصة عن طريق توسيع `Exception`.

- مثال:

```java
try {
    int[] arr = new int[2];
    arr[5] = 10;  // ArrayIndexOutOfBoundsException
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Index out of bounds: " + e.getMessage());
} finally {
    System.out.println("Cleanup done");
}
```

 حقيقي استخدام: معالجة توقيت الشبكة في تطبيق ويب.

---

8. الأنماط

 ما هو: الأنماط تسمح بالكود الآمن من حيث النوع، قابل لإعادة الاستخدام عن طريق تحديد الصفوف، الواجهات والطرق مع أنواع.

 لماذا هو مهم: تكتشف الأخطاء في نوع في وقت التجميع، تقليل الأخطاء في وقت التشغيل، وتزيل الحاجة إلى التحويل.

 كيف يتم استخدامه: شائع في المجموعات مثل List<String> الصفوف والطرق الأنماطية المخصصة.

 الغوص العميق:

- الرموز ? extends T، ? super T تسيطر على الاختلاف في النوع.
- إزالة نوع الأنماط   تبعد معلومات نوع الأنماط في وقت التشغيل للتوافق العكسي.

 مثال:

```java
public class Box<T> {
    private T content;
    public void set(T content) { this.content = content; }
    public T get() { return content; }
}
//
Box<Integer> intBox = new Box<>();
intBox.set(42);
System.out.println(intBox.get());  // Output: 42
```

 حقيقي استخدام: صف Cache<K, V> لتخزين المفتاح القيمة.

---

9. التعبيرات اللامبدية

□ **ما هو:** التعبيرات اللامبدية □□□□□ 8□□ هي تمثيلات مختصرة للوظائف المجهولة، تُستخدم عادةً مع الواجهات الوظيفية.

□ **لماذا هو مهم:** لتسهيل الكود، لتسجيل الأحداث، معالجة المجموعات، وبرمجة الوظيفة.

□ **كيف يتم استخدامه:** مع الواجهات مثل Runnable، Comparator، أو المخصصة مع طريقة مجردة واحدة.

□ **الغوص العميق:**

- □ الصيغة: `(parameters) -> expression` أو `(parameters) -> { statements; }`.
- □ تسمح □□□ب لتدفق البيانات بالشكل الوظيفي.

□ **مثال:**

```
List<String> names = Arrays.asList("Alice", "Bob", "Charlie");
names.forEach(name -> System.out.println(name.toUpperCase()));
```

□ **استخدام حقيقي:** ترتيب قائمة من المنتجات حسب السعر باستخدام `Collections.sort(products, (p1, p2) -> p1.getPrice() - p2.getPrice())`.

---

10. التعليقات

□ **ما هو:** التعليقات هي علامات بيانات مثل□ @Override، @Deprecated□ تُطبق على عناصر الكود، تُعالج في وقت التجميع أو التشغيل.

□ **لماذا هو مهم:** تزويد التعليمات للمترجمين، أو الأطر، أو الأدوات مما يزيد من التلقائية ويقلل من الكود المتكرر.

□ **كيف يتم استخدامه:** يستخدم للتكوين مثل□ @Entity في □□□□، الوثائق، أو فرض القواعد.

□ **الغوص العميق:**

- □ يمكن تعريف التعليقات المخصصة باستخدام `@interface`.
- □ سياسات الاحتفاظ □SOURCE، CLASS، RUNTIME□ تحدد عمرها.

□ **مثال:**

```
public class MyClass {
    @Override
    public String toString() {
        return "Custom string";
    }

    @Deprecated
```

```java
        public void oldMethod() {
            System.out.println("Old way");
        }
    }
```

□ ماد ختسا يقيقح: @Autowired في □□□□□□ لتسجيل التبعيات تلقائيًا.

الإضافية الأساسية الميزات

قيمعتل ،كمهف كيلإ ديزملا نم تازيملا ةعئاشلا يف □□□□ عم تاحرش :ةلصفم

11. الجداول

□ ام وه: الجداول يه تاعومجم ةبترم ةتباث مجحلا نم رصانعلا نم سفن .عونلا

□ اذامل وه مهم: رفوت ةقيرط ،ةطيسب ةلاعف نيزختل و لوصولا ىلإ ميق .ةددعتم

□ فيك متي هماد ختسا: نلعُي ىلع وحنلا يلاتلا `type[] name = new type[size];` وأ ئشنُي .ةرشابم

□ لاثم:

```java
int[] numbers = {1, 2, 3, 4};
System.out.println(numbers[2]);  // Output: 3
```

□ ماد ختسا يقيقح: نيزخت ةمئاق نم تاجرد ةرارحلا .عوبسأل

12. المتعددة الأنواع

□ ام وه: عاونألا ةددعتملا ددحت ةعومجم ةتباث نم ميقلا ،ةتباثلا اًبلاغ عم ميق وأ قرط .ةطبترم

□ اذامل وه مهم: نسحت نامأ عونلا ةيلباقو ةءارقلا ةنراقم تباوثلاب .ماخلا

□ فيك متي هماد ختسا: مدختسي ديدحتل تائفلا ةددحملا لثم ،مايألا ،لاوحألا وأ .ةلاحلا

□ لاثم:

```java
public enum Status {
    PENDING("In progress"), APPROVED("Done"), REJECTED("Failed");
    private String desc;
    Status(String desc) { this.desc = desc; }
    public String getDesc() { return desc; }
}
//
System.out.println(Status.APPROVED.getDesc());  // Output: Done
```

□ ماد ختسا يقيقح: ليثمت ةلاح تابلطلا يف ماظن ةراجت .ةينورتكلإ

13. التدفقات □□□□□ 8□□

- □ ام وه: التدفقات توفر نهجًا وظيفيًا لتدوير، المجموعات تدعم العمليات مثل `filter`، `map`، و `reduce`.
- □ اذامل وه مهم: تسهيل معالجة البيانات، تدعم التوازي، وتحسن تعبيرية الكود.
- □ فيك متي همادختسا: أنشئ تُ من المجموعات باستخدام `.stream()` وتتم ربطها مع العمليات.
- □ لاثم:

```java
List<Integer> nums = Arrays.asList(1, 2, 3, 4, 5);
int sum = nums.stream()
              .filter(n -> n % 2 == 0)
              .mapToInt(n -> n * 2)
              .sum();
System.out.println(sum);  // Output: 12 (2*2 + 4*2)
```

- □ مادختسا يقيقح: تجميع بيانات المبيعات حسب المنطقة.

14. المصنعين

- □ ام وه: المصنعين هي طرق خاصة تُستدعى عند إنشاء، كائن تُستخدم لتهيئة حالة الكائن.
- □ اذامل وه مهم: يضمن أن تبدأ الكائنات مع بيانات، صالحة ويقلل من أخطاء التهيئة.
- □ فيك متي همادختسا: يُعلن باسم، الصف مع متغيرات اختيارية.
- □ لاثم:

```java
public class Book {
    String title;
    public Book(String title) {
        this.title = title;
    }
}
```

- □ مادختسا يقيقح: تهيئة كائن `User` مع اسم المستخدم وكلمة المرور.

15. الوراثة

- □ ام وه: الوراثة تسمح لصف فص□ □يعرف أن يورث حقولًا وطرقًا من صف آخر صف□ □يسيئر باستخدام `extends`.
- □ اذامل وه مهم: تروج لإعادة استخدام الكود وتأسيس علاقة هرمية بين الصفوف.
- □ فيك متي همادختسا: يستخدم لإنشاء نسخة متخصصة من صف عام.
- □ لاثم:

```java
public class Animal {
    void eat() { System.out.println("Eating"); }
}
public class Dog extends Animal {
    void bark() { System.out.println("Barking"); }
}
//
Dog dog = new Dog();
dog.eat();  // Output: Eating
dog.bark(); // Output: Barking
```

 استخدام تحقيقي: صف SavingsAccount يورث من BankAccount.

الخاتمة

التعبيرات الأنماط، الاستثناءات، معالجة الواجهات، التحكم، تدفق أوامر المتغيرات، الطرق، الكائنات، الصفوف،الميزات هذه الصيانة وسهل للتوسيع، قابل قوي، كود بكتابة لك تسمح . في البرمجة عظام هيوغيرها التعليقات، اللامبدية، وثائق اقرأ :**استكشف** الميزات. هذه تجمع صغيرة برامج أكتب :**تجرب** مهاراتك: لتعميق التطبيقات. لمختلف الميزات هذه لرؤية ويب تطبيق أو مكتبة، إدارة نظام حاسبة، آلة مثل مشاريع ابني :**تطبيق** . java.util، java.lang مثل العمل. في

محددة! ميزة أي في الغوص في ترغب كنت إذا أعلمني