

I. Các phiên bản Java LTS

Các phiên bản Java LTS (Long-Term Support) là những phiên bản mà Oracle cam kết hỗ trợ và duy trì trong thời gian dài, thường là 8 năm. Đây là các phiên bản được khuyến nghị cho các ứng dụng và hệ thống sản xuất vì sự ổn định và sự hỗ trợ dài hạn. Tính đến tháng 8 năm 2024, các phiên bản LTS của Java bao gồm:

1. **Java 8 (Java SE 8)** - Phát hành vào tháng 3 năm 2014. Đây là một trong những phiên bản phổ biến nhất và đã nhận được sự hỗ trợ rộng rãi trong nhiều năm. Hỗ trợ chính đã kết thúc vào tháng 3 năm 2022, nhưng hỗ trợ mở rộng tiếp tục cho các khách hàng có gói hỗ trợ.
2. **Java 11 (Java SE 11)** - Phát hành vào tháng 9 năm 2018. Java 11 là phiên bản LTS kế tiếp và được nhiều tổ chức chọn làm nền tảng cho các ứng dụng của mình. Hỗ trợ chính và hỗ trợ mở rộng cho Java 11 vẫn tiếp tục.
3. **Java 17 (Java SE 17)** - Phát hành vào tháng 9 năm 2021. Đây là phiên bản LTS mới nhất trước Java 21. Java 17 mang lại nhiều cải tiến và tính năng mới, đồng thời cung cấp hỗ trợ dài hạn cho các ứng dụng.
4. **Java 21 (Java SE 21)** - Phát hành vào tháng 9 năm 2023. Đây là phiên bản LTS mới nhất và tiếp tục cung cấp các cải tiến và tính năng mới cho nền tảng Java với cam kết hỗ trợ lâu dài.

II. Các nội dung

1. OOP (Lập trình hướng đối tượng)

- **Lớp và đối tượng:** Khái niệm lớp, đối tượng, và cách sử dụng chúng.
- **Kế thừa:** Nguyên tắc kế thừa, lớp cơ sở và lớp dẫn xuất.
- **Đa hình:** Overloading và overriding, phương thức ảo.
- **Đóng gói:** Sử dụng `private`, `protected`, và `public`, getter và setter.

- **Trừu tượng:** Lớp trừu tượng và giao diện (interfaces).

2. Xử lý ngoại lệ

- **Exception Handling:** Sử dụng `try`, `catch`, `finally`, và `throw` để xử lý ngoại lệ.
- **Tạo exception tùy chỉnh:** Định nghĩa các lớp ngoại lệ riêng của bạn.

3. Collections Framework

- **Các interface chính:** `List`, `Set`, `Queue`, `Map`.
- **Các lớp triển khai:** `ArrayList`, `LinkedList`, `HashSet`, `TreeSet`, `HashMap`, `TreeMap`.
- **Iterators:** Sử dụng `Iterator` và `ListIterator`.

4. Luồng và Đọc/Ghi

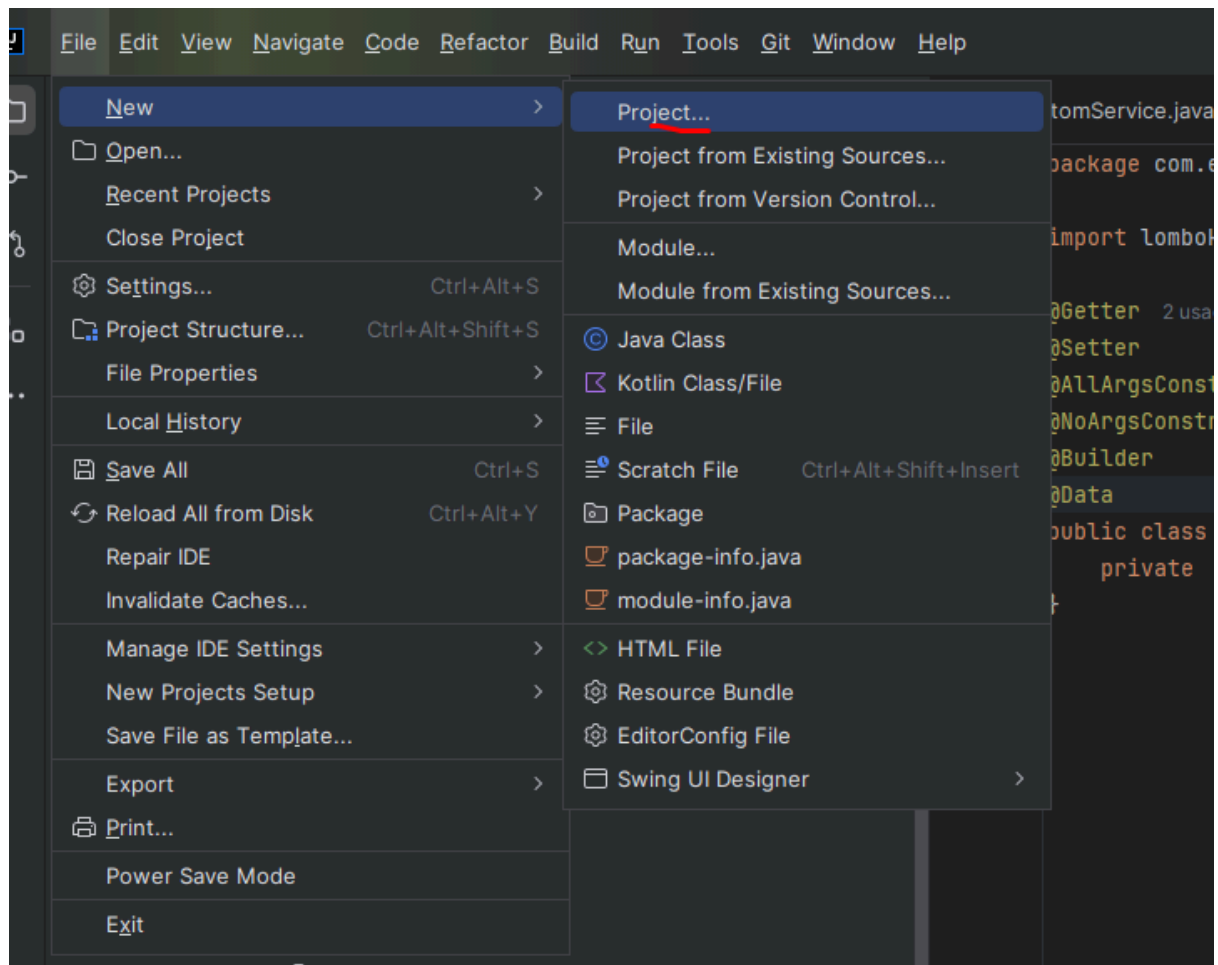
- **Input/Output (I/O):** Sử dụng các lớp trong `java.io` như `File`, `InputStream`, `OutputStream`, `Reader`, và `Writer`.
- **NIO (New I/O):** `java.nio` cho hiệu suất cao và xử lý không đồng bộ.

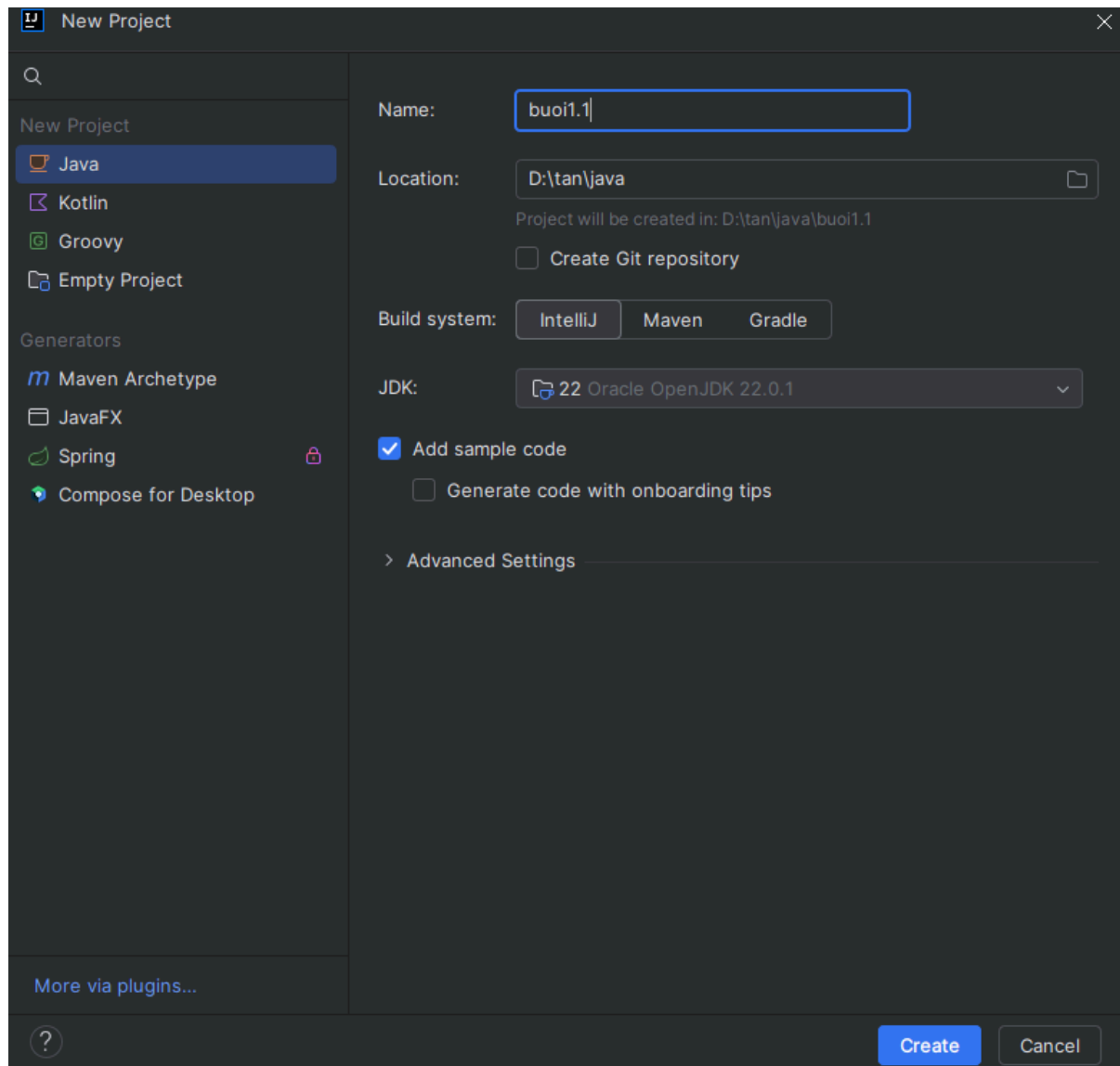
5. Đa luồng (Multithreading)

- **Thread và Runnable:** Tạo và quản lý các luồng với lớp `Thread` và interface `Runnable`.
- **Synchronization:** Đồng bộ hóa và khóa (synchronized blocks, methods).
- **Concurrency Utilities:** Sử dụng các lớp trong `java.util.concurrent` như `ExecutorService`, `CountDownLatch`, `Semaphore`.

6. Java 8 và các phiên bản mới hơn

- **Lambda Expressions:** Biểu thức lambda để xử lý hàm như đối tượng.
- **Streams API:** Xử lý dữ liệu theo cách chức năng.
- **Optional:** Xử lý các giá trị có thể null.





- VS
 - eclipse
 - java toolsuit 4
 - netbeans
 - IntelliJ: [Download IntelliJ IDEA – The Leading Java and Kotlin IDE \(jetbrains.com\)](https://www.jetbrains.com/idea/)
- ## 1. JDK (Java Development Kit)

Định nghĩa: JDK là bộ công cụ phát triển phần mềm chính thức của Java, cung cấp tất cả các thành phần cần thiết để phát triển, biên dịch, và chạy các ứng dụng Java.

Các thành phần chính của JDK:

- **Java Compiler (javac):** Chuyển đổi mã nguồn Java (file .java) thành bytecode (file .class) có thể được chạy trên JVM.
- **Java Runtime Environment (JRE):** Cung cấp các thư viện, các lớp, và các thành phần khác để chạy các ứng dụng Java. JRE bao gồm JVM và thư viện hỗ trợ.
- **Java Virtual Machine (JVM):** Thực thi bytecode Java. JVM là một phần quan trọng của JRE và chịu trách nhiệm chạy các ứng dụng Java.
- **Các công cụ khác:** Bao gồm **javap** (disassembler), **jar** (tạo và quản lý file JAR), **javadoc** (tạo tài liệu từ mã nguồn Java), và nhiều công cụ hỗ trợ khác.

Chức năng:

- **Biên dịch:** Chuyển mã nguồn Java thành bytecode.
- **Gỡ lỗi:** Cung cấp các công cụ để kiểm tra và sửa lỗi mã nguồn.
- **Tạo tài liệu:** Tạo tài liệu API từ mã nguồn.
- **Quản lý:** Các công cụ để quản lý các file JAR và thư viện.

2. JVM (Java Virtual Machine)

Định nghĩa: JVM là một môi trường thực thi ảo cho các ứng dụng Java. Nó chịu trách nhiệm thực thi bytecode Java, cung cấp các dịch vụ cần thiết như quản lý bộ nhớ, quản lý các luồng, và xử lý ngoại lệ.

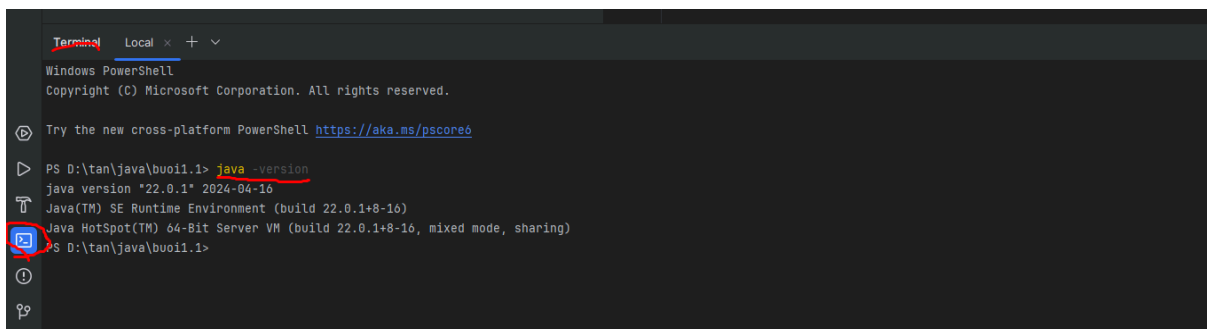
Các thành phần chính của JVM:

- **Class Loader:** Nạp các lớp vào bộ nhớ từ các file bytecode (.class), thường từ file JAR hoặc lớp.

- **Bytecode Verifier:** Kiểm tra bytecode để đảm bảo tính an toàn và hợp lệ trước khi thực thi.
- **Interpreter:** Thực thi bytecode từng lệnh một, hoặc thực thi các lệnh bytecode trực tiếp.
- **Just-In-Time (JIT) Compiler:** Biên dịch bytecode thành mã máy (native code) trong thời gian chạy để cải thiện hiệu suất.
- **Garbage Collector (GC):** Quản lý bộ nhớ tự động, giải phóng bộ nhớ không còn được sử dụng để tránh rò rỉ bộ nhớ.
- **Runtime Data Areas:** Các khu vực dữ liệu trong bộ nhớ dùng để lưu trữ thông tin khi ứng dụng Java đang chạy, bao gồm Stack, Heap, Method Area, và PC Register.

Chức năng:

- **Chạy các ứng dụng Java:** Thực thi bytecode Java, cho phép mã nguồn Java chạy trên bất kỳ nền tảng nào có JVM.
- **Quản lý bộ nhớ:** Cung cấp tính năng Garbage Collection để quản lý và giải phóng bộ nhớ.
- **Quản lý luồng:** Cung cấp cơ chế để xử lý đa luồng và đồng bộ hóa.



```

Terminal Local x + v
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\tan\java\buoi1.1> java -version
java version "22.0.1" 2024-04-16
Java(TM) SE Runtime Environment (build 22.0.1+8-16)
Java HotSpot(TM) 64-Bit Server VM (build 22.0.1+8-16, mixed mode, sharing)
S D:\tan\java\buoi1.1>
  
```

<https://docs.google.com/spreadsheets/d/1ubdw8wi88XLRhWxCqlyTjCVvuFMDSe1S1ZEPKUvJs6Y/edit?gid=0#gid=0>

```

> public class Main {
>     public static void main(String[] args) {
        // Tạo đối tượng Scanner để đọc dữ liệu từ bàn phím
        Scanner scanner = new Scanner(System.in);

        // Nhập một chuỗi
        System.out.print("Nhập tên của bạn: ");
        String name = scanner.nextLine();

        int age=0;
        boolean flag = false;
        // Nhập một số nguyên
        while (!flag) {
            try {
                System.out.print("Nhập tuổi của bạn: ");
                age = scanner.nextInt();
                flag = true; // Nếu không có lỗi, đặt validInput thành true
            } catch (InputMismatchException e) {
                System.out.println("Lỗi: Vui lòng nhập một số nguyên hợp lệ.");
                scanner.next(); // Xóa bỏ dữ liệu không hợp lệ
            }
        }

        // Nhập một số thực
        System.out.print("Nhập chiều cao của bạn (mét): ");
        double height = scanner.nextDouble();

        // Hiển thị thông tin đã nhập
        System.out.println("Tên: " + name);

        System.out.println("Chiều cao: " + height + " mét");

        // Đóng đối tượng Scanner
        scanner.close();
    }
}

```

lưu ý:

```
import java.util.InputMismatchException;
```

- Sử dụng `InputMismatchException` để bắt lỗi khi người dùng nhập dữ liệu không phải là số nguyên hoặc số thực.

- Khi xảy ra lỗi, thông báo cho người dùng biết và sử dụng `scanner.next()` để loại bỏ dữ liệu không hợp lệ còn lại trong bộ đệm của `Scanner`.

Kiểu số nguyên (Integer Types):

- `byte`: 8-bit, giá trị từ -128 đến 127.
- `short`: 16-bit, giá trị từ -32,768 đến 32,767.
- `int`: 32-bit, giá trị từ -2^{31} đến $2^{31}-1$.
- `long`: 64-bit, giá trị từ -2^{63} đến $2^{63}-1$.

Kiểu số thực (Floating-Point Types):

- `float`: 32-bit, giá trị thập phân.
- `double`: 64-bit, giá trị thập phân, chính xác hơn `float`.

Kiểu ký tự (Character Type):

- `char`: 16-bit, đại diện cho một ký tự Unicode.

Kiểu boolean:

- `boolean`: Chỉ có hai giá trị `true` hoặc `false`.

Ví dụ:

```
byte byteValue = 100;
```

```
short shortValue = 20000;
```

```
int intValue = 100000;
```

```
long longValue = 1000000000000L; // Các kiểu dữ liệu số thực float  
floatValue = 10.5f;
```



```
double doubleValue = 20.99; // Kiểu ký tự char charValue = 'A'; //  
Kiểu boolean boolean booleanValue = true; // In ra các giá trị  
System.out.println("byteValue: " + byteValue);  
System.out.println("shortValue: " + shortValue);  
System.out.println("intValue: " + intValue);  
System.out.println("longValue: " + longValue);  
System.out.println("floatValue: " + floatValue);  
System.out.println("doubleValue: " + doubleValue);  
System.out.println("charValue: " + charValue);  
System.out.println("booleanValue: " + booleanValue);
```

Phương pháp ép kiểu tường minh:

```
double doubleValue = 9.78;
```

```
int intValue = (int) doubleValue; // double (8 bytes) được ép kiểu thành int (4 bytes)
```

```
byte byteValue = (byte) intValue; // int (4 bytes) được ép kiểu thành byte (1 byte)
```

Bài tập:

- Nhập xuất a và b là 2 số nguyên từ bàn phím
- tính tổng, hiệu, tích, thương và chia lấy dư
- Ép kiểu phép thương

```
int a=2,b=3;
System.out.println("a + b = "+(a+b));
System.out.println("a - b = "+(a-b));
System.out.println("a x b = "+(a*b));
System.out.println("a % b = "+(a%b));
System.out.println("a / b = "+(float)a/b);
System.out.println("a / b = "+a/(float)b);
System.out.println("a / b = "+a/(double)b);
System.out.println("a / b = "+a*1.0/b);
System.out.println("a / b = "+a/(b*1.0));
// int to float
```

Main.java ×

```
1 import java.util.InputMismatchException;
2 import java.util.Scanner;
3
4 public class Main {
5     public static int calculateSum(int a, int b) { 1 usage
6         return a + b;
7     }
8
9     public static void main(String[] args) {
10         // Main main=new Main();
11         //
12         // System.out.println("sum= " + main.calculateSum(2,3));
13         System.out.println("sum= " + calculateSum(a: 2, b: 3));
14
15         Scanner sc = new Scanner(System.in);
16
17
18         System.out.println("Nhập số a: ");
19         int a = sc.nextInt();
20         sc.nextLine();
21         System.out.println("Nhập số b: ");
22         int b = sc.nextInt();
23         sc.nextLine();
24         System.out.println("Nhập phép tính: ");
25         //cin.ignore();
26         //fflush(stdin)
27         //rewind(stdin)
28         String syntax = sc.nextLine();
29         //float rs = calculate(a,b,syntax);
30         System.out.println("KQ: ");
31         sc.close();
32
33     }
34 }
```

```

    public static int inputNumber(String name){ 2 usages
        Scanner sc = new Scanner(System.in);
        boolean validInput = false;
        int value=0;
        while (!validInput) {
            try {
                float a=0;
                double b=0;
                System.out.printf("Nhập %s của bạn: ", name);
                System.out.printf("Nhập %f của bạn: ", a);
                System.out.printf("Nhập %d của bạn: ", b);
                value = sc.nextInt();
                sc.nextLine();
                validInput = true; // Nếu nhập hợp lệ, thoát vòng lặp
            } catch (InputMismatchException e) {
                System.out.println("Lỗi: Bạn phải nhập một số nguyên hợp lệ.");
                sc.next(); // Xóa bỏ dữ liệu không hợp lệ khỏi scanner
            }
        }

        return value;
    }
}

```

```

    public static void nhapXuat(){ 1 usage
        Scanner sc = new Scanner(System.in);
        System.out.println("Nhập tên của bạn : ");
        String name = sc.nextLine();
        int age=0;
        int height=0;
        age= inputNumber( name: "tuổi");
        height=inputNumber( name: "chiều cao");

        System.out.println("Tên: "+name);

        System.out.println("Chiều cao: "+height+"m");
        sc.close();
    }

    public static void main(String[] args) {
        nhapXuat();
    }
}

```

Lưu ý:

- **Tạo một Scanner duy nhất:** Để tránh sự nhầm lẫn, nên sử dụng một đối tượng **Scanner** duy nhất trong toàn bộ chương trình và đóng nó ở cuối chương trình.
- **int:** Dùng `%d`
- **float:** Dùng `%f` cho cả nhập và in; dùng `%.2f` để hiển thị số chữ số sau dấu thập phân.
- **double:** Dùng `%f` cho cả nhập và in; dùng `%.2f` để hiển thị số chữ số sau dấu thập phân.

Vd:

```
System.out.printf("Nhập giá trị cho số thực kép (double): ");  
double d = sc.nextDouble();
```

```
System.out.printf("Giá trị của số thực kép là: %.2f\n", d);
```

- **char:** Dùng `%c`

VD:

```
System.out.printf("Nhập một ký tự (char): ");
```

```
char ch = sc.next().charAt(0);
```

```
System.out.printf("Ký tự đã nhập là: %c\n", ch);
```

- **boolean:** Dùng `%b`

VD:

```
boolean bool = sc.nextBoolean();
```

```
System.out.printf("Giá trị boolean đã nhập là: %b\n", bool);
```

- **String:** Dùng `%s`

Lưu ý: `InputMismatchException` là một ngoại lệ (exception) trong Java, thuộc gói `java.util`, được ném ra khi người dùng nhập dữ liệu vào `Scanner` không phù hợp với kiểu dữ liệu mong đợi.

Bài tập; Dùng if else hoặc switch case, nhập ngày và tháng của 1 người:, cho biết cung hoàng đạo của người đó

Bạch Dương (Aries): 21/3 - 19/4

Kim Ngưu (Taurus): 20/4 - 20/5

Song Tử (Gemini): 21/5 - 20/6

Cự Giải (Cancer): 21/6 - 22/7

Sư Tử (Leo): 23/7 - 22/8

Xử Nữ (Virgo): 23/8 - 22/9

Thiên Bình (Libra): 23/9 - 22/10

Bọ Cạp (Scorpio): 23/10 - 21/11

Nhan Mã (Sagittarius): 22/11 - 21/12

Ma Kết (Capricorn): 22/12 - 19/1

Bảo Bình (Aquarius): 20/1 - 18/2

Song Ngư (Pisces): 19/2 - 20/3

```

public static String getZodiacSign(int day, int month) { 1 usage
    if (month == 1) {
        return (day <= 19) ? "Ma Kết" : "Bảo Bình";
    } else if (month == 2) {
        return (day <= 18) ? "Bảo Bình" : "Song Ngư";
    } else if (month == 3) {
        return (day <= 20) ? "Song Ngư" : "Bạch Dương";
    } else if (month == 4) {
        return (day <= 19) ? "Bạch Dương" : "Kim Ngưu";
    } else if (month == 5) {
        return (day <= 20) ? "Kim Ngưu" : "Song Tử";
    } else if (month == 6) {
        return (day <= 20) ? "Song Tử" : "Cự Giải";
    } else if (month == 7) {
        return (day <= 22) ? "Cự Giải" : "Sư Tử";
    } else if (month == 8) {
        return (day <= 22) ? "Sư Tử" : "Xử Nữ";
    } else if (month == 9) {
        return (day <= 22) ? "Xử Nữ" : "Thiên Bình";
    } else if (month == 10) {
        return (day <= 22) ? "Thiên Bình" : "Bọ Cạp";
    } else if (month == 11) {
        return (day <= 21) ? "Bọ Cạp" : "Nhân Mã";
    } else if (month == 12) {
        return (day <= 21) ? "Nhân Mã" : "Ma Kết";
    } else {
        return "Ngày hoặc tháng không hợp lệ";
    }
}
}

```

1. Bạch Dương (Aries) - 21/3 - 19/4

- Biểu Tượng: Cừu
- Nguyên Tố: Lửa
- Tính Cách: Quyết đoán, năng động, nhiệt huyết, thích chinh phục thử thách.
- Điểm Mạnh: Dũng cảm, lãnh đạo tốt, sáng tạo.
- Điểm Yếu: Cáp bách, thiếu kiên nhẫn, dễ nóng giận.

2. Kim Ngưu (Taurus) - 20/4 - 20/5

- Biểu Tượng: Bò
- Nguyên Tố: Đất
- Tính Cách: Kiên định, thực tế, yêu thích sự ổn định và an toàn.
- Điểm Mạnh: Tin cậy, chăm chỉ, có khả năng quản lý tài chính tốt.
- Điểm Yếu: Cứng đầu, đôi khi thiếu linh hoạt, có thể trở nên bảo thủ.

3. Song Tử (Gemini) - 21/5 - 20/6

- Biểu Tượng: Cặp song sinh
- Nguyên Tố: Khí
- Tính Cách: Thông minh, giao tiếp tốt, thích khám phá và học hỏi.
- Điểm Mạnh: Linh hoạt, sáng tạo, có khả năng thích nghi tốt.
- Điểm Yếu: Thiếu quyết đoán, có thể không nhất quán, dễ bị phân tâm.

4. Cự Giải (Cancer) - 21/6 - 22/7

- Biểu Tượng: Cua
- Nguyên Tố: Nước
- Tính Cách: Nhạy cảm, quan tâm, bảo vệ gia đình và những người thân yêu.
- Điểm Mạnh: Tận tâm, cảm thông, có khả năng chăm sóc tốt.
- Điểm Yếu: Dễ bị tổn thương, quá nhạy cảm, có thể trở nên sống nội tâm.

5. Sư Tử (Leo) - 23/7 - 22/8

- Biểu Tượng: Sư tử
- Nguyên Tố: Lửa
- Tính Cách: Tự tin, năng động, yêu thích sự chú ý và ngưỡng mộ.
- Điểm Mạnh: Lãnh đạo tốt, nhiệt tình, sáng tạo.
- Điểm Yếu: Kiêu ngạo, có thể trở nên quá tự phụ, yêu cầu sự chú ý quá mức.

6. Xử Nữ (Virgo) - 23/8 - 22/9

- Biểu Tượng: Xử nữ
- Nguyên Tố: Đất
- Tính Cách: Tỉ mỉ, tổ chức, phân tích, chú ý đến chi tiết.
- Điểm Mạnh: Chính xác, chăm chỉ, có khả năng giải quyết vấn đề tốt.
- Điểm Yếu: Cầu toàn, có thể trở nên quá nghiêm khắc với bản thân và người khác.

7. Thiên Bình (Libra) - 23/9 - 22/10

- Biểu Tượng: Cân
- Nguyên Tố: Khí
- Tính Cách: Tìm kiếm sự cân bằng, công bằng, hòa bình, có khả năng giao tiếp tốt.
- Điểm Mạnh: Thân thiện, khéo léo trong giao tiếp, có thể giải quyết xung đột.
- Điểm Yếu: Do dự, khó đưa ra quyết định, có thể trở nên phụ thuộc vào ý kiến của người khác.

8. Bọ Cạp (Scorpio) - 23/10 - 21/11

- **Biểu Tượng:** Bọ cạp
- **Nguyên Tố:** Nước
- **Tính Cách:** Sâu sắc, mạnh mẽ, bí ẩn, có khả năng cảm nhận và hiểu biết sâu về cảm xúc.
- **Điểm Mạnh:** Quyết tâm, trung thành, có khả năng phục hồi tốt.
- **Điểm Yếu:** Ghen tuông, có thể trở nên quá nghi ngờ, khó mở lòng.

9. Nhân Mã (Sagittarius) - 22/11 - 21/12

- **Biểu Tượng:** Cung tên
- **Nguyên Tố:** Lửa
- **Tính Cách:** Thích phiêu lưu, tự do, có tinh thần lạc quan và yêu thích khám phá.
- **Điểm Mạnh:** Năng động, hào phóng, có tư duy mở rộng.
- **Điểm Yếu:** Thích mạo hiểm, có thể thiếu kiên nhẫn, không chú trọng đến chi tiết.

10. Ma Kết (Capricorn) - 22/12 - 19/1

- **Biểu Tượng:** Dê núi
- **Nguyên Tố:** Đất
- **Tính Cách:** Chăm chỉ, có tổ chức, tham vọng, có khả năng quản lý tốt.
- **Điểm Mạnh:** Có kế hoạch rõ ràng, kiên nhẫn, có khả năng làm việc chăm chỉ.
- **Điểm Yếu:** Có thể trở nên quá nghiêm khắc, thiếu sự linh hoạt, dễ cảm thấy áp lực.

11. Bảo Bình (Aquarius) - 20/1 - 18/2

- **Biểu Tượng:** Người mang nước
- **Nguyên Tố:** Khí
- **Tính Cách:** Sáng tạo, độc lập, có tư tưởng tiên bộ, quan tâm đến xã hội.
- **Điểm Mạnh:** Tư duy sáng tạo, nhân văn, có khả năng tư duy độc lập.
- **Điểm Yếu:** Có thể trở nên xa cách, khó gần, không dễ hòa nhập với cảm xúc của người khác.

12. Song Ngư (Pisces) - 19/2 - 20/3

- **Biểu Tượng:** Hai con cá
- **Nguyên Tố:** Nước
- **Tính Cách:** Nhạy cảm, giàu trí tưởng tượng, dễ đồng cảm và cảm nhận sâu sắc.
- **Điểm Mạnh:** Thấu hiểu, sáng tạo, có khả năng giúp đỡ người khác.
- **Điểm Yếu:** Có thể trở nên mơ mộng, dễ bị tổn thương, thiếu quyết đoán.

Link câu lệnh if:

https://www.w3schools.com/java/java_conditions.asp

- **Bài tập 2: Thần số học, tính số đời(số chủ đạo): Nhập ngày tháng năm từ bàn phím**

Số đời (Life Path Number) được tính từ ngày tháng năm sinh bằng cách cộng tất cả các chữ số trong ngày, tháng và năm sinh, rồi tiếp tục rút gọn kết quả về một chữ số từ 1 đến 9 (hoặc số 11, 22, 33 - các số thầy phong).

Ví dụ:

- Ngày sinh: 15
- Tháng sinh: 8
- Năm sinh: 1990

Cách tính:

1. Cộng tất cả các chữ số: $1 + 5 + 8 + 1 + 9 + 9 + 0 = 33$
2. Rút gọn số 33 thành một chữ số: $3 + 3 = 6$

Số đời trong ví dụ trên là 6.

```

public static int calculateLifePathNumber(int day, int month, int year) { 1 usage
    int sum = day + month + year;
    return reduceToSingleDigit(sum);
}

// Phương thức rút gọn số đến chữ số duy nhất hoặc số thấy phong
private static int reduceToSingleDigit(int number) { 1 usage
    while (number > 9 && number != 11 && number != 22 && number != 33) {
        number = sumOfDigits(number);
    }
    return number;
}

// Phương thức tính tổng các chữ số
private static int sumOfDigits(int number) { 1 usage
    int sum = 0;
    while (number > 0) {
        sum += number % 10;
        number /= 10;
    }
    return sum;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    System.out.print("Nhập ngày sinh (dd): ");
    int day = sc.nextInt();

    System.out.print("Nhập tháng sinh (mm): ");
    int month = sc.nextInt();

    System.out.print("Nhập năm sinh (yyyy): ");
    int year = sc.nextInt();

    int lifePathNumber = calculateLifePathNumber(day, month, year);

```

Ý Nghĩa Các Số

Số 1 - Người Lãnh Đạo

- **Tính cách:** Độc lập, quyết đoán, có khả năng lãnh đạo và sáng tạo. Người mang số 1 thường là những cá nhân tự tin, không ngại thử thách và có khả năng đạt được mục tiêu lớn.
- **Điểm Mạnh:** Sáng tạo, tự tin, có khả năng lãnh đạo tốt.
- **Điểm Yếu:** Có thể trở nên cứng đầu, không dễ làm việc nhóm.

Số 2 - Người Hòa Giải

- **Tính cách:** Nhạy cảm, hòa đồng, chú trọng đến sự hòa hợp và cộng tác. Người mang số 2 thường giỏi trong việc xây dựng các mối quan hệ và làm việc nhóm.
- **Điểm Mạnh:** Tinh tế, có khả năng làm hòa, nhạy cảm với cảm xúc của người khác.
- **Điểm Yếu:** Dễ bị tổn thương, có thể thiếu tự tin.

Số 3 - Người Sáng Tạo

- **Tính Cách:** Vui vẻ, sáng tạo, giao tiếp tốt và có khả năng biểu đạt cảm xúc. Người mang số 3 thường yêu thích nghệ thuật và có khả năng truyền cảm hứng cho người khác.
- **Điểm Mạnh:** Sáng tạo, hòa đồng, năng động.
- **Điểm Yếu:** Có thể trở nên bề ngoài và thiếu sự tập trung.

Số 4 - Người Thực Tế

- **Tính Cách:** Thực tế, đáng tin cậy, có tổ chức và chăm chỉ. Người mang số 4 thường chú trọng đến công việc và yêu thích sự ổn định.
- **Điểm Mạnh:** Có kế hoạch, chăm chỉ, kiên nhẫn.
- **Điểm Yếu:** Có thể trở nên quá bảo thủ và khó thích nghi với sự thay đổi.

Số 5 - Người Phiêu Lưu

- **Tính Cách:** Thích mạo hiểm, tự do, năng động và yêu thích sự thay đổi. Người mang số 5 thường tìm kiếm những trải nghiệm mới và không ngại thử thách.
- **Điểm Mạnh:** Linh hoạt, thích nghi tốt, đầy năng lượng.
- **Điểm Yếu:** Có thể trở nên thiếu ổn định và dễ bị phân tâm.

Số 6 - Người Chăm Sóc

- **Tính Cách:** Tận tâm, quan tâm đến gia đình và bạn bè, và thường là người chăm sóc. Người mang số 6 thường tìm kiếm sự cân bằng trong cuộc sống và có xu hướng giúp đỡ người khác.
- **Điểm Mạnh:** Chăm sóc, có trách nhiệm, nhân ái.
- **Điểm Yếu:** Có thể trở nên quá lo lắng và chịu trách nhiệm quá nhiều.

Số 7 - Người Tìm Kiếm

- **Tính Cách:** Sâu sắc, phân tích và có xu hướng tìm kiếm sự hiểu biết và tri thức. Người mang số 7 thường rất tư duy và thích khám phá các vấn đề sâu xa.

- **Điểm Mạnh:** Phân tích tốt, trí tuệ, tĩnh lặng.
- **Điểm Yếu:** Có thể trở nên cô lập và thiếu xã hội.

Số 8 - Người Thành Công

- **Tính Cách:** Mạnh mẽ, có tham vọng và khả năng quản lý tài chính tốt. Người mang số 8 thường tập trung vào mục tiêu tài chính và sự nghiệp.
- **Điểm Mạnh:** Quyết đoán, có khả năng lãnh đạo, khả năng quản lý tài chính.
- **Điểm Yếu:** Có thể trở nên quá tập trung vào công việc và thiếu quan tâm đến mối quan hệ cá nhân.

Số 9 - Người Nhân Ái

- **Tính Cách:** Nhân ái, từ bi và có tinh thần phục vụ xã hội. Người mang số 9 thường có sự quan tâm lớn đến các vấn đề xã hội và tình nguyện giúp đỡ người khác.
- **Điểm Mạnh:** Từ bi, nhân ái, có tinh thần phục vụ.
- **Điểm Yếu:** Có thể trở nên quá hy sinh và dễ bị tổn thương.

11 - Số Thầy Phong - Người Đem Đến Ánh Sáng

- **Tính Cách:** Tinh thần mạnh mẽ, có tầm nhìn xa, và thường là những người có khả năng truyền cảm hứng mạnh mẽ. Người mang số 11 thường có sự kết nối mạnh mẽ với tâm linh và trí tuệ cao.
- **Điểm Mạnh:** Trực giác tốt, khả năng lãnh đạo, truyền cảm hứng.
- **Điểm Yếu:** Có thể trở nên dễ bị căng thẳng và dễ bị ảnh hưởng bởi cảm xúc.

22 - Số Thầy Phong - Người Xây Dựng

- **Tính Cách:** Có khả năng biến ý tưởng thành hiện thực và có tầm nhìn lớn. Người mang số 22 thường có khả năng lãnh đạo và quản lý dự án quy mô lớn.
- **Điểm Mạnh:** Tầm nhìn xa, có khả năng tổ chức và xây dựng.

- **Điểm Yếu:** Có thể trở nên quá căng thẳng với trách nhiệm lớn.

33 - Số Thầy Phong - Người Thầy Tâm Linh

- **Tính cách:** Đem đến sự chữa lành và ánh sáng cho người khác. Người mang số 33 thường là những người truyền cảm hứng và có khả năng giúp đỡ và chữa lành tâm hồn người khác.
- **Điểm Mạnh:** Tử bi, khả năng truyền cảm hứng, có ảnh hưởng mạnh mẽ.
- **Điểm Yếu:** Có thể trở nên dễ bị áp lực từ trách nhiệm lớn.

```

> public class Array1 {
>     public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, 5};
        for (int number : numbers) {
            System.out.println(number);
        }

        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[i].length; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }

        //Tinh tổng đường chéo chính
        int tong=0;
        for (int i = 0; i < matrix.length; i++) {
            tong+=matrix[i][i];
        }
        System.out.printf("Tổng đường chéo chính là: %d",tong);
        System.out.println();
        int tong1=0;
        for (int i = 0; i < matrix.length; i++) {
            tong1+=matrix[i][matrix.length-i-1];
        }
        System.out.printf("Tổng đường chéo chính là: %d",tong1);
    }
}

```

Khai báo hằng số:

```
public static final int MY_CONSTANT = 10;
```

```
public static final double PI = 3.14159; public static final String  
GREETING_MESSAGE = "Hello, World!";
```

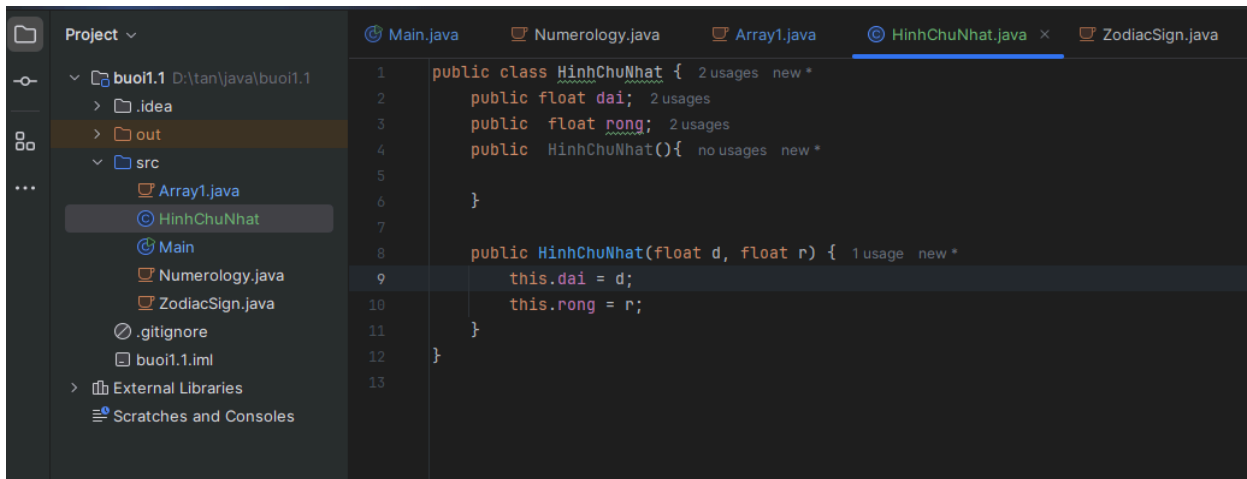
```
> public class Array1 {  * ngoctan1234 *  
    public enum Day { 2 usages new *  
        SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY 1 usage  
    }  
    public static final double PI = 3.14159; 1 usage  
    public static final double E = 2.71828; no usages  
    > public static void main(String[] args) {  * ngoctan1234 *  
  
        Day today = Day.FRIDAY;  
        double r=3;  
        System.out.println("s="+r*r*PI);  
        switch (today) {  
            case MONDAY:  
                System.out.println("Today is Monday.");  
                break;  
            case FRIDAY:  
                System.out.println("Today is Friday.");  
                break;  
            case SUNDAY:  
                System.out.println("Today is Sunday.");  
                break;  
            default:  
                System.out.println("Today is a regular day.");  
                break;  
        }  
    }  
}
```

Từ khóa **final**:

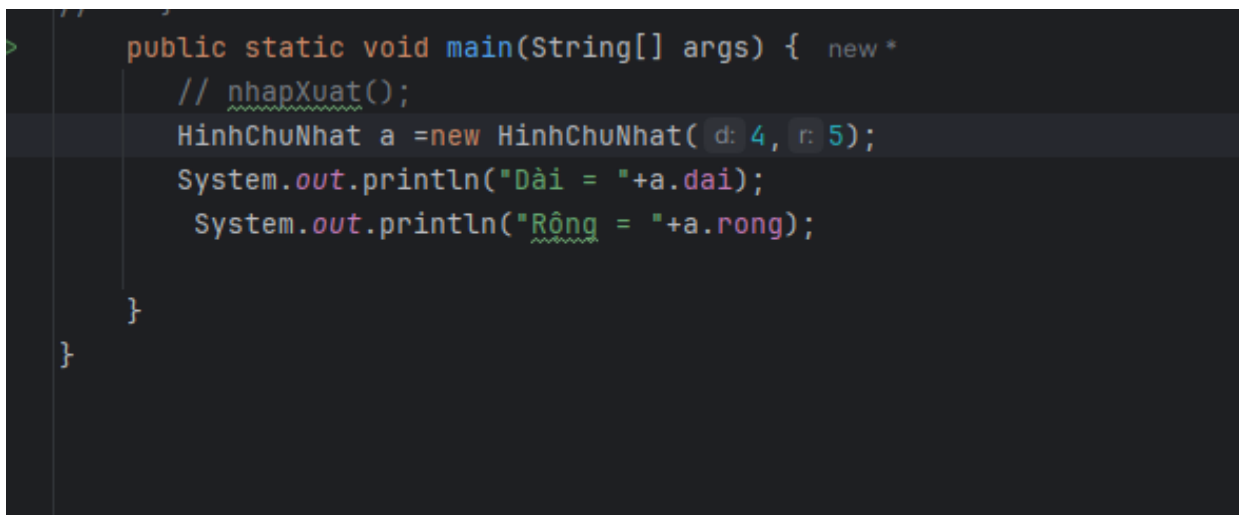
- Đảm bảo rằng giá trị của biến không thay đổi sau khi nó được gán. Nếu bạn cố gắng thay đổi giá trị của một biến đã được khai báo là **final**, bạn sẽ nhận được lỗi biên dịch.

Từ khóa **static**:

- Chỉ định rằng biến thuộc về lớp thay vì thuộc về một đối tượng cụ thể của lớp đó. Điều này cho phép bạn truy cập hằng số bằng cách sử dụng tên lớp.



```
1 public class HinhChuNhat { 2 usages new *
2     public float dai; 2 usages
3     public float rong; 2 usages
4     public HinhChuNhat(){ no usages new *
5
6     }
7
8     public HinhChuNhat(float d, float r) { 1 usage new *
9         this.dai = d;
10        this.rong = r;
11    }
12 }
13
```



```
> //
    public static void main(String[] args) { new *
        // nhapXuat();
        HinhChuNhat a = new HinhChuNhat( d: 4, r: 5);
        System.out.println("Dài = "+a.dai);
        System.out.println("Rộng = "+a.rong);
    }
}
```