**BÀI TẬP LẬP TRÌNH JAVA CHƯƠNG 1**

**Câu 1: Mô tả các kiểu dữ liệu cơ sở trên Java**

* Kiểu logic: **boolean**.
* Các kiểu số nguyên (integer): **byte, short, char, int, long**.
* Các kiểu số thực (real number) (Cũng được gọi là kiểu dấu chấm động (floating point)): **float, double**.

**Câu 2: Mô tả các cấu trúc lệnh trên Java**

* Cấu trúc tuần tự
* Cấu trúc rẽ nhánh
* Cấu trúc lặp

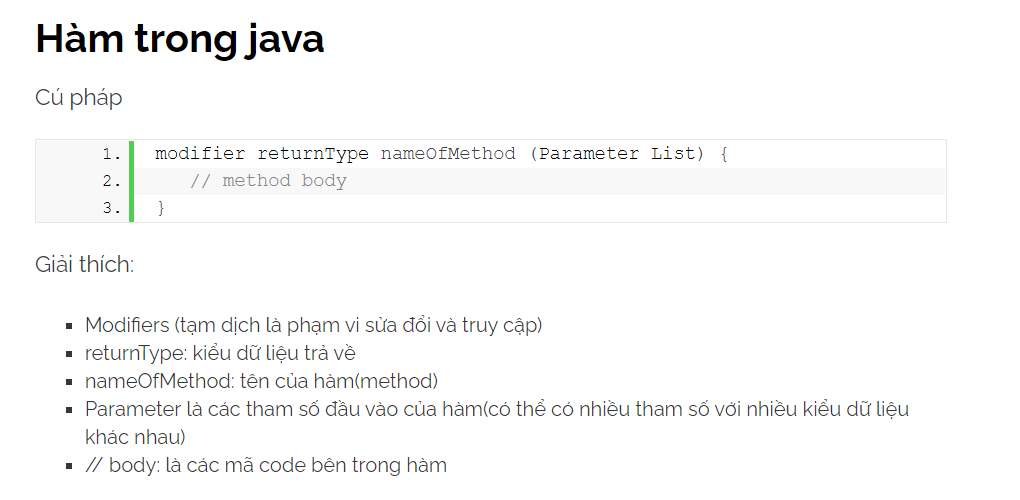
**Câu 3: Mô tả các môi trường lập trình Java**

* JVM (Java Virtual Machine): là môi trường dùng để chạy ứng dụng được viết bằng ngôn ngữ lập trình Java, giúp Java có thể chạy nhiều nền tảng (Platform) khác nhau.
* JRE (Java Runtime Environment): bao gồm các thư viện, JVM và những thành phần bổ sung để chạy những ứng dụng được viết bằng Java.
* JDK (Java Development Kit): là tập hợp JRE và những công cụ cần thiết để phát triển ứng dụng bằng Java.

**Câu 4: Liệt kê các đặc trưng của nên tảng lập trình Java**

* Đơn giản
* Thân thuộc
* Hướng đối tượng
* Bảo mật
* Hiệu suất cao
* Đa luồng
* Độc lập nền

**Câu 5: Trình bày function trên Java**

****

* Hàm có trả về kết quả: dùng từ khoá return để trả về kết qủa mà nó đã tính toán được.
* Hàm không trả về kết quả: loại hàm này chúng ta chỉ cần gọi để sử dụng.

**Câu 6: So sánh các câu lệnh lặp trên Java**

* Vòng lặp For: Là câu lệnh lặp với số lần biết trước. Điều kiện xác định trước khi thực thi khối lệnh.

Cú pháp :

for (chỉ số bắt đầu; chỉ số kết thúc; giá trị tăng mỗi bước lặp)

{

Thực hiện câu lệnh;

}

* Vòng lặp While: Là câu lệnh lặp với số lần không biết trước, điều kiện xác định trước khi thực thi khối lệnh.

Cú pháp:

While (điều kiện lặp)

{

Thực hiện câu lệnh;

}

* Vòng lặp do-while: Là câu lệnh lặp với số lần ko biết trước, điều kiện xác định sau khi thực thi khối lệnh.

Cú pháp:

do

{

Thực hiện câu lệnh;

}

While (điều kiện lặp);

**Câu 7: Cho ví dụ về các hàm đệ quy và không đệ quy trên Java**

Ví dụ: hàm tính giai thừa của một số nguyên

* Không đệ quy:

private static int giaiThua(int n){

Int x=1;

For (int i=2;i<=n;i++)

{

x\*=i;

}

Return x;

}

* Đệ quy:

private static int giaiThua(int x){

if (i==0||i==1) return 1;

else

return i\*giaiThua(i-1);

}

**Câu 8: Tóm tắt kiểu dữ liệu String**

Thông thường, string là một chuỗi các ký tự. Nhưng, trong java string là một đối tượng biểu diễn một nối tiếp của các ký tự. Lớp java.lang.String được sử dụng để tạo đối tượng string.

Có 2 cách để tạo đối tượng String:

* Sử dụng string literal
* Sử dụng từ khóa new

Cách khởi tạo chuỗi trong Java:

* String s = new String();
* String s1 = "Xin chao";
* char chars[] = {’i’, ’B’, ’y’, ’t’, ’e’, ’C’, ’o’, ’d’, ’e’};
* String s2 = new String(chars);

**Câu 9: Cho ví dụ về các lớp bao trên Java**

//AutoBoxing

int i = 100;

Integer ib = i;

//AutoUnboxing

int iu = ib;

public static void main(String[] args) {

Integer integer = Integer.valueOf("121");

System.out.println(integer);

System.out.println("Integer to String: " + integer.toString()); }

**Câu 10: So sánh Java với các ngôn ngữ lập trình C, C++**

Điểm chung: đều dựa trên ngôn ngữ C, có nhiều nét tương đồng về từ khóa như: public, private, static và các kiểu dữ liệu nguyên thủy (ngoại trừ boolean), các toán tử. Đều hỗ trợ lập trình hướng đối tượng.

Điểm khác: Java loại bỏ biến con trỏ, thứ đã tồn tại ở C/C++, Java hỗ trợ các luồng (thread) tự nhiên.

**Câu 11: Lập trình tính tổng 1 + 2 + .... + n**

import java.util.Scanner;

public class baitapchuong1 {

private static Scanner scanner = new Scanner(System.in);

public static int DEC\_10 = 10;

public static void main(String[] args) {

System.out.print("Nhập số nguyên dương n = ");

int n = scanner.nextInt();

System.out.printf("Tổng của các chữ số "

+ "của %d là: %d", n, tong(n));

}

private static int tong(int n) {

var tong=0;

for (int i = 0;i <= n;i++) {

tong+=i;

}

return tong;

}

}**Câu 12: Lập trình tính n!**

private static int giaiThua(int i) {

if (i==0||i==1) return 1;

else

return i\*giaiThua(i-1);

}

**Câu 13: Lập trình hàm kiểm tra số nguyên n có là số nguyên tố hay không**

private static boolean ktrSNT(int i) {

if (i<2) return false;

else

for (int j=2;j<=Math.sqrt(i);j++)

{

if (i%j==0)

return false;

}

return true;

}

**Câu 14: Lập trình hàm nhận vào họ tên, trả về Họ Tên dạng proper**

private static String proper(String s) {

StringBuffer kq= new StringBuffer();

StringTokenizer tg= new StringTokenizer(s);

while(tg.hasMoreTokens()){

String s1= tg.nextToken();

s1=s1.toLowerCase();

String s2= s1.substring(0, 1);

s1= s1.replaceFirst(s2, s2.toUpperCase());

kq.append(s1).append(" ");

}

String newS= new String(kq);//trả về chuỗi dạng proper

return newS;

}

**Câu 15: Lập trình hàm xuất ra mã nhị phân của số nguyên n**

private static void xuatMaNhiPhan(int i) {

if (i<2) System.out.print(i);

else {

xuatMaNhiPhan(i/2);

System.out.print(i%2);

}

}

**Câu 16: Phân tích câu lệnh switch thành các lệnh if, lấy ví dụ minh họa**

Lệnh switch và if đều là câu lệnh rẽ nhánh.

* If:
  + if( [điều kiện] ) { Khối lệnh 1} //Câu dạng thiếu
  + if( [điều kiện] ) { Khối lệnh 1} else {Khối lệnh 2} //Câu dạng đủ
* Switch:

switch(bieu\_thuc){

gia\_tri case :

//Cac lenh

break; //tuy y

case value :

//Cac lenh

break; //tuy y

//Ban co the co so luong lenh case bat ky nao.

default : //tuy y

//Cac lenh

}

* Biến được sử dụng trong một lệnh switch chỉ có thể là byte, short, int hoặc char.
* Giá trị cho một case phải giống kiểu dữ liệu của biến trong switch và nó phải là hằng số.
* Khi gặp một lệnh *break* thì switch kết thúc, và luồng điều khiển nhảy tới dòng tiếp theo lệnh switch.
* Không phải mọi case đều cần một break. Nếu không có lệnh break xuất hiện, luồng điều khiển sẽ đi qua các case sau đó tới khi gặp một lệnh break.
* Ví dụ minh họa:

A=10;

Switch(A):{

Case 10: system.out.println(“gioi”);break;

Case 8: system.out.println(“kha”); break;

Case 5: system.out.println(“trung binh”); break;

Default: system.out.println(“kem”);

}

Chuyển qua câu lệnh if, chúng ta có:

A=10;

If (A==10) system.out.println(“gioi”);

Else If (A==8) system.out.println(“kha”);

Else If (A==5) system.out.println(“trung binh”);

Else system.out.println(“kem”);

**Câu 17: Phân tích câu lệnh try catch, lấy ví dụ minh họa**

Khối lệnh try-catch dùng để xử lý ngoại lệ (lỗi) của chương trình đang thực hiện. Khi xảy ra lỗi, ngoại lệ, Java thông thường sẽ dừng thực thi chương trình và đưa ra một thông báo, hay nói cách khác là Java ném ra một exception – Ngoại lệ. Xử lý ngoại lệ (Exception Handling) trong java là một cơ chế xử lý các lỗi runtime để có thể duy trì luồng bình thường của ứng dụng. Quá trình xử lý exception được gọi là catch exception (bắt ngoại lệ), nếu Runtime System không xử lý được ngoại lệ thì chương trình sẽ kết thúc. Cú pháp:

try {

// code có thể ném ra ngoại lệ

}

catch(Exception\_class\_Name ex) {

// code xử lý ngoại lệ

}

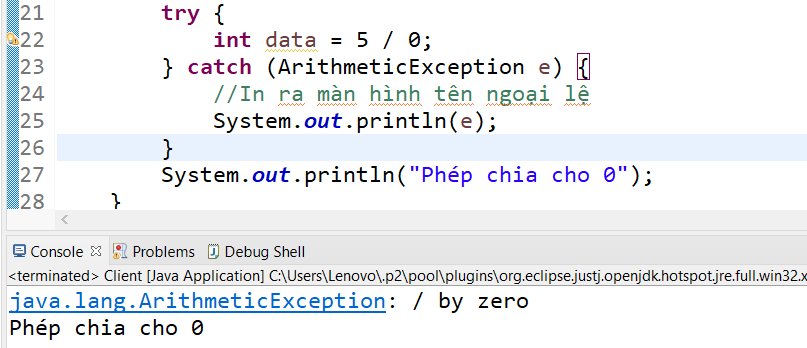
finally {

// code trong khối này luôn được thực thi

}

\*có thể có nhiểu khối catch nhưng chỉ có 1 khối try duy nhất.

Ví dụ minh họa



**Câu 18: Biến đổi cấu trúc lệnh lặp for thành lệnh lặp điều kiện cho trước bằng ví dụ cụ thể**

Ví dụ tính tổng các số nguyên từ 1 tới một số n bất kì, giả sử n=5:

* Đối với cấu trúc lặp for:

int s=0;//mảng tính tổng khởi tạo =0;

for (int i=1; i<=5; i++)

{

s+=i;

}

String b=String.format("tổng từ 1 tới 5 là: %d", s);

System.out.println(b);

* Đối với cấu trúc lặp điều kiện cho trước, ví dụ như while(có thể áp dụng với do-while):

int s=0, i=1;//mảng tính tổng khởi tạo =0, chỉ số bắt đầu là 1

while (i<=5)

{

s+=i;

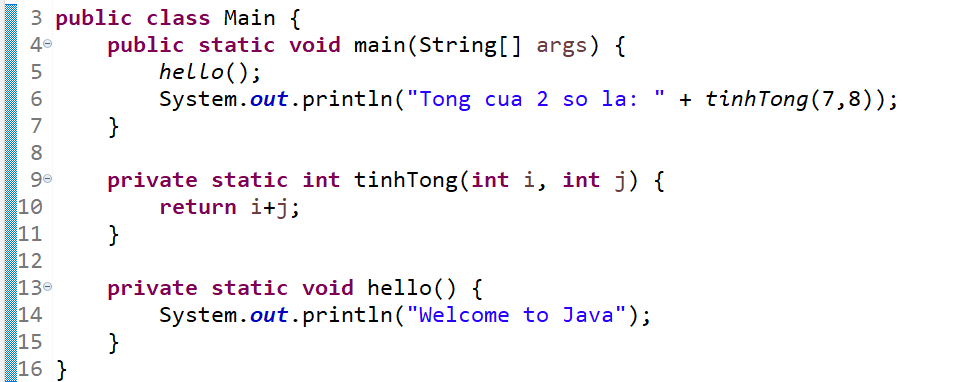
i++;

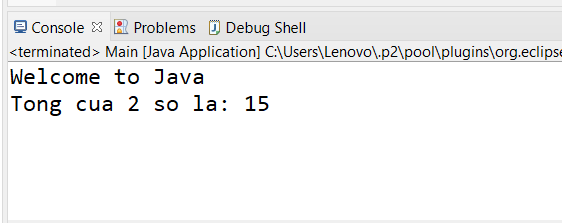
}

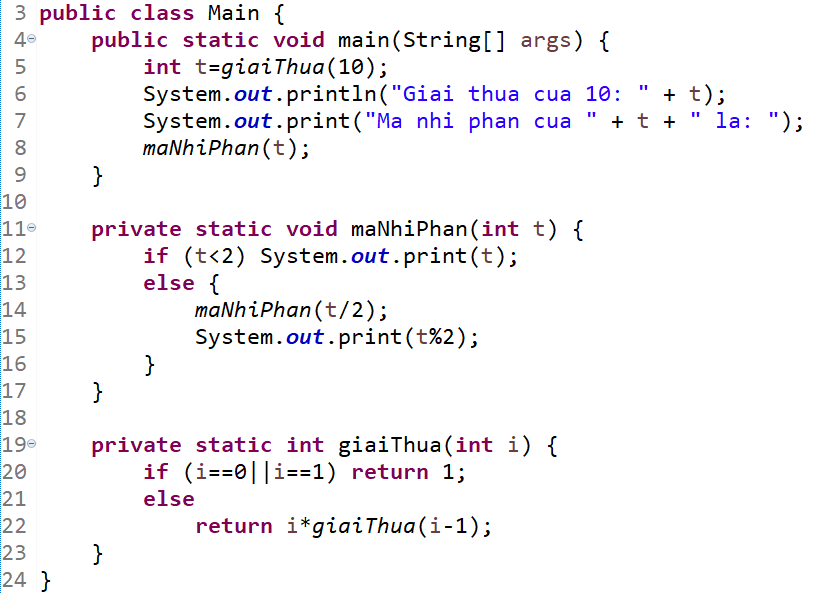
String b=String.format("tổng từ 1 tới 5 là: %d", s);

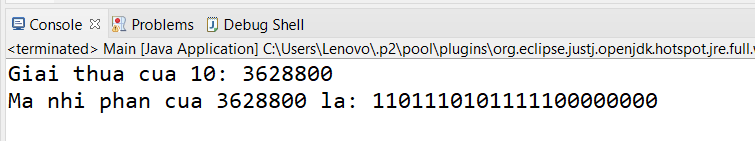
System.out.println(b);

**Câu 19: Phân tích quá trình vận hành của chương trình Java bao gồm nhiều lời gọi hàm bằng ví dụ cụ thể**

 Đây là dạng cấu trúc tuần tự, nghĩa là thực hiện từ trên xuống dưới. Đầu tiên chương trình sẽ thực hiện hàm *hello()* (dòng 5). Hàm này sẽ xuất ra màn hình câu “Welcome to Java”. Sau đó sẽ tiếp tục thực hiện câu lệnh tiếp theo in ra tổng của 2 số, trong câu lệnh print (dòng 6) sẽ bao gồm 1 hàm tính tổng, chương trình sẽ bắt đầu gọi hàm *tinhTong()* gồm 2 tham số 7 và 8 sau đó trả về kết quả và xuất ra màn hình. Ta có kết quả như sau:



**Câu 20: Phân tích quá trình vận hành của chương trình Java bao gồm nhiều lời gọi hàm đệ quy bằng ví dụ cụ thể**

 Chương trình bao gồm 2 hàm đệ quy, là hàm tính giai thừa và hàm xuất mã nhị phân. Đầu tiên chúng ta khai báo số nguyên t và gán giá trị hàm giaiThua(10) cho t (dòng 5). Từ kết quả của t đã được gán, chúng ta tiếp tục thực hiện chuyển đổi mã nhị phân nhờ thực hiện hàm đệ quy *maNhiPhan()* (dòng 8). Sau khi thực hiện chúng ta có kết quả như sau:

Câu 21: Các thành phần của một đối tượng

* Thuộc tính: bao gồm các đặc điểm của đối tượng
* Phương thức: bao gồm các hành động mà đối tượng thực hiện

Câu 22: Giải thích thuộc tính của đối tượng

Là các đặc điểm dùng để mô tả đối tượng

Câu 23: Giải thích phương thức của đối tượng

Đại diện cho hành vi (chức năng) của một đối tượng như gửi tiền, rút tiền, ...

Câu 24: Cho ví dụ về lớp có đủ thuộc tính và phương thức

Lớp SinhVien

* Thuộc tính: Họ tên, MSSV, Ngày sinh, Quê Quán, Sđt, ...
* Phương thức: Đăng ký học phần, Đóng học phí, Đi học, Điểm danh...

Câu 25: Cho ví dụ về lớp chỉ có thuộc tính

Lớp ConNguoi

* Thuộc tính: Màu da, Giới tính, Chiều cao, Cân nặng, ...

Câu 26: Phân biệt lớp và đối tượng

Đối tượng(Object) là một thể hiện của một lớp(Class). Lớp là một mẫu hoặc thiết kế từ đó các đối tượng được tạo ra. Vì vậy, đối tượng là các thể hiện (kết quả) của một lớp.

Một lớp là một nhóm đối tượng có các thuộc tính chung.

Câu 27: Trình bày lại các yếu tố cơ bản trong lập trình hướng đối tượng trên Java

Lập trình hướng đối tượng (OOP-Object-Oriented Programming) là một phương pháp hay mô hình giúp tăng năng suất, đơn giản hóa việc bảo trì, dễ dàng mở rộng trong thiết kế phần mềm bởi việc cung cấp một vài khái niệm như:

* Đối tượng (Object)
* Lớp (Class)

4 đặc điểm của hướng đối tượng:

* Kế thừa (Inheritance)
* Đa hình (Polymorphism)
* Trừu tượng (Abstraction)
* Đóng gói (Encapsulation)

Câu 28: Lập trình hiện thực lớp Phân số và các phép toán trên phân số

Câu 29: Áp dụng thừa kế, lập trình hiện thực lớp Sinh viên, Thực tập viên, Giảng viên, Nhân viên sao cho thể hiện được các phương thức tính lương, tính điểm ở mức tổng quát

Câu 30: Lập trình hiện thực giao diện, lớp Hình, Hình chứ nhật, Hình tròn, Hình tam giác cùng thực hiện các phương thức tính chu vi, diện tích

Câu 31: Lập trình hiện thực giao diện, lớp Sản phẩm, Giỏ hàng, Khách hàng, Cửa hàng trong các bài toán thương mại điện tử

Câu 32: Lập trình hiện thực bài toán các youtuber quản lý videos của họ, youtube viewer quản lý các videos mà họ đã xem, đã lưu

Câu 33: Lập trình hiện thực bài toán truyền sự kiện giảm giá của cửa hàng cho các khách hàng VIP

Câu 34: Lập trình hiện thực bài toán truyền sự kiện đăng mới video của youtuber cho các viewer của họ

Câu 35: Chỉ ra sự khác biệt giữa lớp trừu tượng và giao diện

|  |  |
| --- | --- |
| Abstract class | Interface |
| Abstract class có phương thức abstract (không có thân hàm) và phương thức non-abstract (có thân hàm). | Interface chỉ có phương thức abstract. Từ java 8, nó có thêm các phương thức default và static. |
| Abstract class không hỗ trợ đa kế thừa. | Interface **có hỗ trợ đa kế thừa** |
| Abstract class có các biến final, non-final, static and non-static. | Interface chỉ có các biến static và final. |
| Abstract class có thể cung cấp nội dung cài đặt cho phương thức của interface. | Interface không thể cung cấp nội dung cài đặt cho phương thức của abstract class. |
| Từ khóa abstract được sử dụng để khai báo abstract class. | Từ khóa interface được sử dụng để khai báo interface. |
| Ví dụ:  public abstract class Shape {  public abstract void draw();  } | Ví dụ:  public interface Drawable {  void draw();  } |

Câu 36: Phân tích sự hoạt động của mô hình truyền sự kiện trên Java

1. Event source phát sính EventObject khi gặp biến cố
2. Event source truyền EventObject tới tất cả các Listener của Event source
3. Các Listener dựa trên thông tin trong EventObject để xử lý sự kiện và trả về kết quả

Câu 37: Phân tích sự hoạt động của cơ chế đa hình trên Java

Câu 38: Tổng hợp các loại quan hệ giữa các đối tượng trên Java

Câu 39: So sánh quan hệ bao gộp và quan hệ thừa kế

Câu 40: Phác thảo các giao diện và lớp đáp ứng mô hình bài toán thiết kế phần mềm giao diện đồ họa (GUI)

Câu 41: Phác thảo các giao diện và lớp đáp ứng mô hình bài toán thương mại điện tử trong đó cho phép thanh toán trực tuyến

Câu 42: Giải thích khái niệm stream trong Java

Câu 43: So sánh byte stream và character stream

* Byte Streams: Xữ lý input và output của 8 bits. Sử dụng byte stream có thể lưu trữ các ký tự, video, âm thanh, hình ảnh, v.v. Common class được dùng cho byte stream đó là InputStream và OutputStream
* Character Streams: Xữ lý input và output cho 16 bit Unicode. Sử dụng character stream chỉ có thể dùng để đọc và ghi ký tự văn bản. Common class được dùng cho character streams đó là File Reader và File Writer.

Câu 44: Diễn giải thao tác nhập số nguyên từ bàn phím

Câu 45: Diễn giải thao tác nhập số thực từ bàn phím

Câu 46: Mô tả thao tác đọc ghi file

Câu 47: Lập trình thêm, xóa, sửa, tìm kiếm file danh sách sinh viên

Câu 48: Lập trình thêm, xóa, sửa, tìm kiếm file danh sách sản phẩm

Câu 49: Lập trình truy xuất thông tin giỏ hàng trên file

Câu 50: Giải quyết bài toán tính toán trên phân số, kết quả lưu trữ vào file

Câu 51: Lập trình sắp xếp danh sách nhân viên trong file tăng dần theo lương cơ bản

Câu 52: Trình bày khái niệm thread

Câu 53: Trình bày khái niệm multi thread

Câu 54: Mô tả các trạng thái của thread

Câu 55: Mô tả các phương thức chuyển đổi trạng thái thread

Câu 56: Mô tả mức độ ưu tiên của thread

Câu 57: Mô tả các cách thức tạo thread

Câu 58: Trình bày mối liên hệ giữa start và run thread

Câu 59: Cho ví dụ extends thread

Câu 60: Cho ví dụ implements runnable

Câu 61: Cho ví dụ anonymous thread

Câu 62: Cho ví dụ wait và notify trong các thread