

OOP ATLAS

ThS. Nguyễn Thế Hoàng | fb/giao.lang.bis

Version 21.06




© 2021 – Nguyễn Thế Hoàng – FPT University

01 Định Danh/Tên Gọi và Giá Trị – Variable & Value

Giá trị đơn (single value, primitive value)		Giá trị phức tạp, phức hợp (complex, composite, object value)	
Tên gọi/Biến	Giá trị/Dữ liệu/Thông tin	Tên gọi/Biến	Giá trị/object
VAT, discount	10%	sếp, sky	{tên = Nguyễn Thanh Tùng nghệDanh = Sơn Tùng MTP yob = 1994 bàiHát = Chúng ta của hiện tại... }
PI	3,14		
C	300.000	chiPu	{tên = Nguyễn Thùy Chi nghệDanh = Chi Pu yob = 1993 bàiHát = Anh ơi anh ở lại... }
họcPhí	27.300.000 VNĐ		
chiTiêu	10.000.000 VNĐ	miDu	{tên = Đặng Thị Mỹ Dung nghệDanh = Mida yob = 1989 phim = Thiên mệnh anh hùng... }
sốMônNợ	40/45		
diemTrungBinh	4,9	nữHoàngNộiY	{tên = Trần Thị Ngọc Trinh nghệDanh = Ngọc Trinh yob = 1989 phim = Vòng eo 56... }
lầnThi	2		
frameRate	240FPS	mình/ta/tao/ tui/this	{tên = Hoàng Ngọc Trinh bútDanh = giáo.làng, xàm yob = 2001 môn = Đa cấp dám thành công... }
age	20/thanh xuân		

© 2021 – Nguyễn Thế Hoàng – FPT University

Hàm/Function – Method/Phương Thức

<p>Hàm là gì?</p> <p>Một quy tắc/cách thức xử lí đầu vào để có đầu ra</p>		$y = f(x) = x^2$ <p>SinhTổ = máyXayÉp (tráiCâyĐưaVào) = nghiền/ép/trộn</p>
<p>Sử dụng hàm</p> <p>Gọi tên em/hàm với data vào/ra</p>		$y = f(2) = 2^2 = 4$ $y = f(3) = 3^2 = 9$ <p>SinhTổ-Cam = máyXayÉp (cam)</p> <p>SinhTổ-Cà = máyXayÉp (càRốt, càPháo, càPhê)</p>
<p>Bên trong hàm có gì?/Cấu tạo hàm</p>		$y = f(x) = x^2$

© 2021 – Nguyễn Thế Hoàng – FPT University

Hàm/Function – Method/Phương Thức (tt.)

<p>SinhTổ = máyXayÉp (tráiCâyĐưaVào) = nghiền/ép/trộn</p> <p>$y = f(x) = x^2$</p> <p>dầu ra tên hàm đầu vào</p> <p>output function name input</p> <p>returned value method name parameter/argument implement of function</p>	
<p>không-ra f(không-vào)</p> <pre>void f(void) { cần scanf() để có data mà xử lí; cần printf() để in kết quả đã xử lí; }</pre>	<p>không-ra f(có-vào)</p> <pre>void f(int a) { không nên scanf() vì đã có data a đưa vào để xử lí; cần printf() để in kết quả đã xử lí; }</pre>
<p>có-ra f(không-vào)</p> <pre>int f(void) { cần scanf() để có data mà xử lí; không nên printf() vì đã return data ra ngoài; bắt buộc return xxx-value ra ngoài qua tên hàm; } //Tên hàm là 1 biến được gán value từ return</pre>	<p>có-ra f(có-vào) //double r = sqrt(4); Math.sqrt(4);</p> <pre>int f(int a) { //IPO không nên scanf() vì đã có data a đưa vào để xử lí; không nên printf() vì đã return data ra ngoài; bắt buộc return xxx-value ra ngoài qua tên hàm; } //Tên hàm là 1 biến được gán value từ return //SOÀI CA, RE-USE CAO NHẤT, NHÚNG VÀO LỆNH KHÁC</pre>

© 2021 – Nguyễn Thế Hoàng – FPT University

02 Function/Method Exercises

Tạo mới project tên Ex1. **Submit lên LMS. Deadline: 16/5/2021 23:59**

- Viết hàm kiểm tra 1 số có phải số nguyên tố hay không?

```
public static boolean isPrime(int n) //dùng Math.sqrt(n)
```

- Viết hàm in ra các số nguyên tố trong đoạn từ 1...1000. Ví dụ: 2, 3, 5, 7, 11, ..., 997

```
public static void printPrimeList() //có ngon dùng lại/re-use hàm trên
```

- Viết hàm in ra 1000 số nguyên tố đầu tiên. Ví dụ: 2, 3, 5, 7, 11, ..., 7919

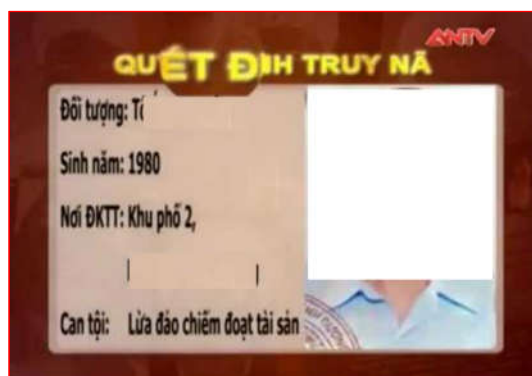
```
public static void print1000FirstPrimes() //do-while là phù hợp nhất
```

- Viết hàm nhập từ bàn phím 2 hệ số a, b đại diện cho phương trình bậc nhất một ẩn $ax + b = 0$. In ra nghiệm của phương trình này.

```
public static void solveSimpleEquation() {
    Scanner sc = ...
}
```

© 2021 – Nguyễn Thế Hoàng – FPT University

03 Đối Tượng/Object



Đối tượng: những “thứ-object-thing” quanh ta, hữu hình/vô hình, chứa đựng nhiều thông tin giúp ta mô tả được chúng, nhận diện được chúng, phân biệt được chúng, đếm được chúng, (chạm) được chúng

Đối tượng: chứa nhiều info được mô tả/nhận diện/phân biệt qua:

- Tên gọi tắt/định danh (biến phức tạp/biến object)
- Các đặc điểm (biến & value)
- Các hành vi/phương thức/hành động/behavior/method/hàm

© 2021 – Nguyễn Thế Hoàng – FPT University

Đối Tượng/Object (tt.)

TÌM KIẾM ĐỐI TƯỢNG TRUY NÃ

Từ khóa:

Loại tội danh:

Loại truy nã:

Hệ loại tội danh:

Phạm vi truy nã:

TRUY NÃ

- Đối tượng truy nã
- Đối tượng truy nã đặc biệt
- Đối tượng truy nã quốc tế
- Đối tượng đình nã

QUẢN LÝ ĐỐI TƯỢNG TRUY NÃ

STT	Họ tên đối tượng	Ảnh đại diện	Năm sinh	Nơi DKTT	Họ tên bố/mẹ	Tội danh	Số/ ngày OD
1	Mã		1992			Tội lừa đảo chiếm đoạt tài sản	Số 06 - Ngày 14/12/2019
2	Lê		1989			Tội trộm cắp tài sản	Số 05 - Ngày 13/12/2019
3	N		1999			Trộm cắp tài sản	Số 04 - Ngày 11/12/2019
4	N		1989			Tội lừa đảo chiếm đoạt tài sản	Số 07 - Ngày 11/12/2019
5	Tô		1995			Tội mua bán trái phép chất ma túy	Số 20 - Ngày 10/12/2019

Thông tin chung

Họ và tên	Tên khác	Long
Giới tính	Nam	Năm sinh
Nơi sinh	Đ	ng
Họ khẩu thường trú	Ấp Phú	ng
Hơn đăng ký tạm trú	Ấp Phú	ng
Quốc tịch		Dân tộc
Họ tên bố		Họ tên mẹ

Đặc điểm nhận dạng

Chiều cao	Màu da
Đặc điểm mái tóc	Đặc điểm lông mày
Đặc điểm mũi	Đặc điểm tai
Đặc điểm mắt	
Đặc điểm khác	

Căn tội

Tội danh	Tội lừa đảo chiếm đoạt tài sản
Hệ loại tội danh	Ma túy
Loại truy nã	Phạm vi truy nã
Quyết định truy nã	Số 06/ Ngày 14/12/2019
Đơn vị ra quyết định	Cá huyện

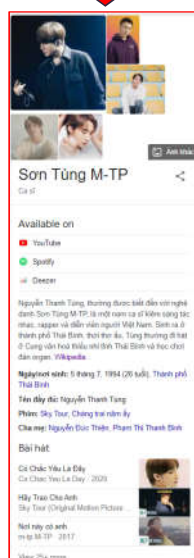
Khi phát hiện báo cho

Báo cho đơn vị	
Số điện thoại	

© 2021 – Nguyễn Thế Hoàng – FPT University

Đối Tượng/Object – Tên Đối Tượng

sếp/sky



chiPu



miDu



nữHoàngNộiY

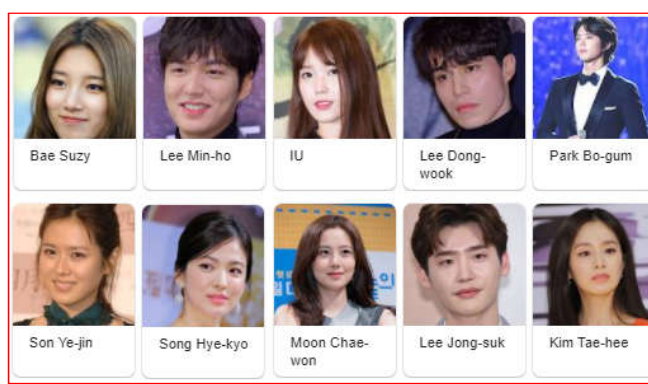
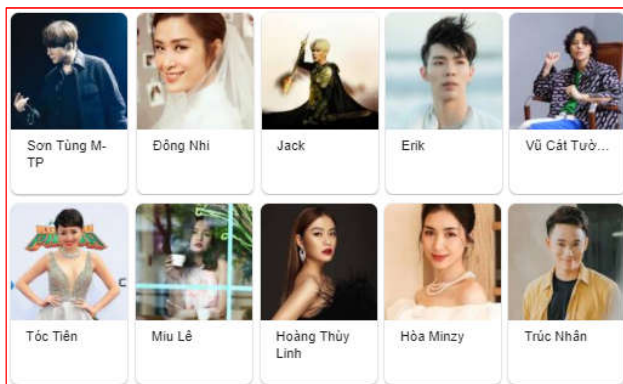


tui/this/g.l



{ tên = Hoàng Ngọc Trinh
bút danh = giáo làng, xàm
yob = 2001
môn = Đa cấp dám thành công... }

03 Đối Tượng/Object – Tên Đối Tượng – Nhóm/Class



Nhóm/Phân Nhóm/Phân Loại/
Group/Class (Classify)

Name: _____
Yob: _____
Hits: _____

CASI

Nhóm/Phân Nhóm/Phân Loại/
Group/Class (Classify)

Name: _____
Yob: _____
Movies: _____

DIỄNVIÊN

© 2021 – Nguyễn Thế Hoàng – FPT University

03 Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng



- Khuôn/Đúc/Mold/Blueprint/Prototype/Template/Form/**Class**
- Thing/**Object**/instance/vật thể/đối tượng/thể hiện/hiện thân/phần bản/nhân bản/hiện hình/bức tượng từ Khuôn

© 2021 – Nguyễn Thế Hoàng – FPT University

Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng (tt.)



1. Khuôn (kèm phễu)
2. Đúc - construct() - tạo vật thể/object - đổ vật liệu vào
3. Xem sản phẩm - getInfo()
4. Chỉnh sửa - setInfo()

© 2021 – Nguyễn Thế Hoàng – FPT University

Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng (tt.)



1. Khuôn (kèm phễu)
2. Đúc - construct() - tạo vật thể/object - đổ vật liệu vào
3. Xem sản phẩm - getInfo()
4. Chỉnh sửa - setInfo()



© 2021 – Nguyễn Thế Hoàng – FPT University

Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng (tt.)



Class

Dog	
-	name: ???
-	weight: ???
-	hairColor: ???
-	breed: ???
-	...
+	rượtMèo(): ???
+	sủa(): ???
+	...

© 2021 – Nguyễn Thế Hoàng – FPT University

Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng (tt.)

Các đối tượng khả nghi (object) cần được theo dõi/lưu trữ thông tin, theo dõi hành vi

<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Ngáo Cỗ Weight: 50.0 kg Hair Color: Hung đỏ Breed: Ngao <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): rượt mèo rừng Sủa(): nhè nhẹ 	<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Ngáo Đá Weight: 60.0 kg Hair Color: Đen-vàng Breed: Ngao <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): rượt mèo nhà Sủa(): dữ dằn 	<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Chihuahua Weight: 1.0 kg Hair Color: Café sữa Breed: Chihuahua <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): rú mèo đi chơi Sủa(): không ra hơi
<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Chi Oa Oa Weight: 0.5 kg Hair Color: Sữa Breed: Chihuahua <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): bỏ chạy khi thấy mèo Sủa(): không ra hơi 	<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Vàng Ối Weight: 15.5 kg Hair Color: Vàng mật ong Breed: Chó ta bản địa <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): xù đẹp mèo Sủa(): không thêm sủa 	<p>Đặc điểm nhân dạng (characteristics)</p> <ul style="list-style-type: none"> Name: Bê Tô Weight: 10.5 kg Hair Color: Rắn ri Breed: Chó ta bản địa <p>Hành động (methods)</p> <ul style="list-style-type: none"> Rượt mèo(): xù đẹp mèo Sủa(): tập xong mới sủa

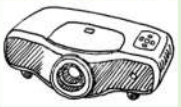





Class

Dog	
-	name: ???
-	weight: ???
-	hairColor: ???
-	breed: ???
-	...
+	rượtMèo(): ???
+	sủa(): ???
+	...

© 2021 – Nguyễn Thế Hoàng – FPT University

03 Nhân Bản/Đúc/Clone/Tạo/Construct Đối Tượng (tt.)

Các đối tượng khả nghi (object) cần được theo dõi/lưu trữ thông tin, theo dõi hành vi

		
Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ??? 	Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ??? 	Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ???
		
Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ??? 	Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ??? 	Đặc điểm nhân dạng (characteristics) <ul style="list-style-type: none"> • ???: ??? • ???: ??? • ???: ??? • ???: ??? Hành động (methods) <ul style="list-style-type: none"> • ???(): ??? • ???(): ???

Class ???

???: _____
 ???: _____
 ???: _____

© 2021 – Nguyễn Thế Hoàng – FPT University

03 Class Excercies

Tạo mới project tên Ex2. Submit lên LMS. Deadline: 23/5/2021 23:59

1. Lưu trữ thông tin các cuốn sách có trên Amazon hoặc Tiki
2. Lưu thông tin của các tài khoản gửi tiền/rút tiền ở ngân hàng nào đó.
 Tài khoản bao gồm: mã (số) tài khoản, tên chủ tài khoản, CMND/căn cước, số điện thoại, số dư (balance – mặc định 50K).
 - Tạo 5 tài khoản nào đó với số dư bất kì cho trước lớn hơn 50K
 - **Hắc não:** Hãy cho 1 vài tài khoản rút tiền. Nhớ là rút tiền thì phải để lại số dư tối thiểu 50K
 - Hãy vắn tin tài khoản - in sao kê....

© 2021 – Nguyễn Thế Hoàng – FPT University

03 Class Exercises (tt.)

Hãng sản xuất: Atlantic
Loại đồng hồ: Đồng hồ nam
Chất liệu dây: Dây da
Chất liệu mặt: Sapphire
Chất liệu vỏ: Thép không gỉ
Năng lượng sử dụng: Quartz
Độ chịu nước: 5 ATM
Đường kính: 42 mm
Bảo hành: 10 năm
Thương hiệu: Thụy Sĩ
Tư vấn và đặt hàng: 098.668.1189
Thanh toán: Trực tiếp khi nhận sản phẩm
Mã sản phẩm: AT-68750.41.25R
Giá bán: **20.862.000 VNĐ**



Hãng sản xuất: Atlantic
Loại đồng hồ: Đồng hồ nam
Chất liệu dây: Dây da
Chất liệu mặt: Sapphire
Chất liệu vỏ: Thép không gỉ
Năng lượng sử dụng: Quartz
Độ chịu nước: 5 ATM
Đường kính: 42 mm
Bảo hành: 10 năm
Thương hiệu: Thụy Sĩ
Tư vấn và đặt hàng: 098.668.1189
Thanh toán: Trực tiếp khi nhận sản phẩm
Mã sản phẩm: AT-68450.41.52
Giá bán: **12.587.500 VNĐ**



Thương Hiệu	Casio
Số Hiệu Sản Phẩm	SHE-3806SPG-7A.LOR
Xuất Xứ	Nhật Bản
Giới Tính	Nữ
Kính	Mineral Crystal (Kính Cứng)
Máy	Quartz (Pin)
Bảo Hành Quốc Tế	1 Năm 4.646.000 đ
Bảo Hành Tại Hải Triều	
Đường Kính Mặt Số	44.5mm x 39.3mm
Bé Dây Mặt Số	8.6 mm
Niêng	
Dây Dẻo	
Màu Mặt Số	Trắng
Chống Nước	5 ATM
Chức Năng	Lịch Ngày – Lịch Thứ – Đồng hồ 24h



- Mã sản phẩm:SD01056
- Kiểu dáng:Giày xăng đan
- Chất liệu:Da tổng hợp
- Độ cao:1cm
- Màu sắc:Xanh lá-Đen-Nâu
- Kích cỡ:34-35-36-37-38-39

Mã SP : SD01056

BIG SALE 200,000đ



JUNO

Chất liệu da tổng hợp, êm ái
Kiểu dáng chiến binh cách điệu, thời trang
Có khóa kéo phía sau để dễ dàng mang vào

- Mã sản phẩm:BB03007
- Kiểu dáng: Giày búp bê
- Chất liệu: Da tổng hợp
- Độ cao:1cm
- Màu sắc:Đỏ-Đen-Hồng
- Kích cỡ:34-35-36-37-38-39

Mã SP : BB03007

BIG SALE 200,000đ



Chất liệu da bóng thời trang
Giày mũi nhọn, cắt hình chữ V nổi bật
Quai ôm cổ chân tạo điểm nhấn nữ tính

Giày tây nam công sở da bóng ROZALO RM52686

Thông tin sản phẩm 4332.000 **đ199.000**

Mã sản phẩm: RM52686

Chất liệu: Da PU

Đế: Cao su

Size: 39, 40, 42, 42, 43.

Màu sắc: Nâu, Đen

Tính năng: Táng chiều cao



© 2021 – Nguyễn Thế Hoàng – FPT University

04 Access Specifier/Access Modifier/Quyền Được Sờ

Đứng ngoài Object – Nhìn thấy gì?

+ **tóc:** Hường_____

+ **da :** Trắng_____

+ **? :** _____

Đứng ngoài Object – Không nhìn thấy gì?

- **vòngEo :** _____

- **tiền :** _____

- **địaChỉ :** _____

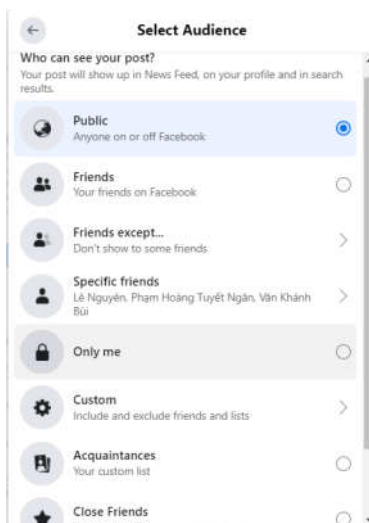
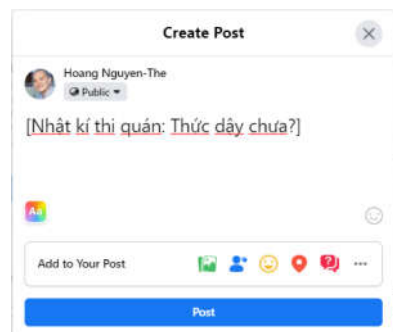
- **? :** _____



© 2021 – Nguyễn Thế Hoàng – FPT University

04

Access Specifier/Access Modifier/Quyền Được Sờ (tt)



Quyền được “sờ” thông tin trong một khu vực/một object

- private: cấm sờ hiện vật.
Không thấy từ bên ngoài

+ public: cho không biểu không.
Cứ chạm nếu muốn

protected: họ hàng, bà con
giấu nhau làm gì

~ default: bạn thân cùng phòng

© 2021 – Nguyễn Thế Hoàng – FPT University

05

Chốt hạ 1: Class – Object – Biến Object – Chạm/Sờ

GIÁO.LÀNG KHẨU QUYẾT

1. BIẾN LÀ TÊN GỌI ĐẠI DIỆN CHO 1 GIÁ TRỊ ĐƠN GIẢN (PRIMITIVE) HOẶC PHỨC TẠP (COMPOSITE, COMPLEX, OBJECT)
2. CÓ BIẾN LÀ CÓ VÙNG NHỚ ĐƯỢC CẤP TRONG RAM (Biến primitive hay biến object đều được cấp vùng RAM tương ứng)
3. CÓ NEW LÀ CÓ VÙNG NHỚ MỚI ĐƯỢC CẤP TRONG RAM (Vùng nhớ NEW – vùng nhớ bộ – còn gọi là vùng nhớ object – vùng nhớ ENCAPSULATION – chứa toàn bộ thông tin của object gồm đặc điểm và hành vi được CLONE từ Khuôn-Class. Toán tử new...() TRẢ VỀ địa chỉ/tọa độ vùng nhớ object NEW vừa tạo, và gửi địa chỉ này cho “biến object/biến con trỏ” giữ giữ). **CÓ THỂ CLONE KÈM THÊM 1 LẦN VÀ DUY NHẤT 1 LẦN 1 VÙNG NHỚ STATIC DÙNG CHUNG!!! Static** cũng có thể được tạo ra khi nó được triệu gọi/sử dụng lần đầu tiên bởi **Tên-Class.chấm-static**
4. BIẾN PRIMITIVE LƯU VALUE NGAY TRONG VÙNG NHỚ ĐƯỢC CẤP (Tên tắt của giá trị đơn sẽ được cấp 1 số byte tùy data type – ví dụ int 4, long 8. ON/OFF của transistor của vùng RAM sẽ biểu diễn giá trị cần lưu trữ)
5. BIẾN OBJECT THAM CHIẾU – TRỎ THẰNG VÀO VÙNG NEW ĐƯỢC CẤP – VÙNG OBJECT (Biến object được xem là biến “CON TRỎ” “trỏ” thẳng vào vùng RAM “bộ” vừa được NEW. “Trỏ” nghĩa là nắm/lưu lại tọa độ/địa chỉ của vùng NEW)
6. SỜ VÀO VÙNG NEW ĐƯỢC CẤP QUA BIẾN CON TRỎ CHẤM (.) (Biến object trỏ đến tọa độ vùng NEW/vùng object được cấp. Vùng NEW có rất nhiều thông tin được quyền “sờ chạm”. Thông tin này được CLONE theo thiết kế Khuôn-Class lúc đúc ra object. Thích chạm sờ gì thì chấm cái đó)
7. AI CÓ NHIỀU THÔNG TIN NHẤT, KỂ ĐÓ NÊN XỬ LÝ THÔNG TIN NÀY THAY VÌ ĐỂ KỂ KHÁC (Hàm gắn với đối tượng/class để xử lý chính những info mà nó đang chứa – ENCAPSULATION)

© 2021 – Nguyễn Thế Hoàng – FPT University



Chốt hạ 1: Class – Object – Biến Object – Chạm/Sờ (tt.)

GIÁO.LÀNG KHẨU QUYẾT | CÁC HỆ QUẢ

1. **BIẾN PRIMITIVE CHỈ TÓN 1 VÙNG RAM** (ON-OFF vùng RAM này chứa luôn value cần lưu trữ. Tên biến chính là value cần dùng, KHÔNG CHẤM)
2. **OBJECT TÓN 2 VÙNG RAM** (1 vùng RAM là vùng NEW - chính là object là object chứa đặc tính và hành vi. 1 vùng RAM khác là biến con trỏ/biến object trỏ vùng NEW object. Chạm sờ vùng NEW dùng biến object CHẤM. ON-OFF trong biến con trỏ/biến object, value của biến con trỏ chính là tọa độ/địa chỉ vùng NEW)
3. **BIẾN NÀO KHAI BÁO TRONG HÀM THÌ NÀM TRONG STACK-SEGMENT**
4. **OBJECT/VÙNG NEW NÀM TRONG HEAP-SEGMENT**
5. **MẢNG – KẾ THỪA – INTERFACE** sẽ bổ sung riêng sau...

© 2021 – Nguyễn Thế Hoàng – FPT University



Chốt hạ 2: RECIPE for COOKING a JAVA APP

1. **NHẬN DIỆN, MÔ TẢ ĐƯỢC CÁC OBJECT** xuất hiện trong bài toán qua **đặc điểm** (biến kèm value-data) và **hành vi/hành động** xử lý trên các đặc điểm đó. Ví dụ: đặc điểm **yob** thì có hàm **getAge()**. Đối tượng Student x có đặc điểm **id, name, yob, gpa**, hành vi **doQuiz(), showProfile()**
2. **NHẬN DIỆN GOM NHÓM OBJECT – CLASS**, là những cụm object chia sẻ chung các đặc điểm/hành vi **nổi bật** khác với các Nhóm khác. Ví dụ Nhóm Ca Sĩ có đặc điểm nổi bật: **bàiHits**. Nhóm chính là một dạng phân loại hay còn gọi là **Class**. Đặt tên cho Nhóm/Class này. Thiết kế Class/Khuôn gồm **đặc điểm** (biến chứa chỗ cho value) và **hành vi** (hàm xử lý tổng quát trên biến)
3. **TẠO MỚI PROJECT | TẠO MỚI CLASS/KHUÔN** bắt buộc đặt ở một **package**/ngăn tủ/kho nào đó
4. **TẠO KHOẢNG TRỐNG CHỨA CHỖ TRONG CLASS/KHUÔN**. Khoảng trống chính là các đặc điểm của object trong tương lai sẽ được tạo ra hay đúc từ cái Khuôn/Class. Chúng còn được gọi là **field/instance variable/property/attribute/state/characteristic**. Nên để chúng là **private**
5. **TẠO PHỄU/CONSTRUCTOR** dùng để nhận vật liệu bên ngoài đổ vào các khoảng trống field trong Khuôn/Class để đúc đối tượng. Có thể có nhiều constructor tương ứng với nhiều cách đúc tượng khác nhau. Phễu gọi kèm lệnh **new**. Mỗi lần gọi phễu mỗi lần new một lần object mới được tạo ra. **PUBLIC CHO PHỄU**
6. **TẠO CÁC HÀNH ĐỘNG GETX() SETX()**, cho phép xem/lấy và sửa/cập nhật thông tin - value của đối tượng đã được đúc được new, được đổ qua phễu trước đó. Các hàm này có thể gọi đi gọi lại nhiều lần nhưng phải đi kèm theo tên tắt/biến object và dấu CHẤM. **PUBLIC CHO GET() SET() MẶC ĐỊNH**, có thể tùy chỉnh

© 2021 – Nguyễn Thế Hoàng – FPT University

05 Chốt hạ 2: RECIPE for COOKING a JAVA APP (tt.)

7. **TAO HÀM TOSTRING()** để trả về/show ra hết thông tin đang có sẵn/value của các field đã được đổ/được gán giá trị đang nằm trong object. Có thể tùy ý chỉnh sửa lại hàm này theo ý của mình, nhưng tuyệt đối **KHÔNG** được đổi tên hàm và kiểu trả về
CÓ THỂ GENERATE CODE TỰ ĐỘNG PHỄU, GETX(), SETX(), TOSTRING() qua phím nóng Alt+Insert | Insert Code...
Khi đó ta xài **this**.
8. **TAO CÁC HÀM TỰ TUI, CỦA RIÊNG MÌNH.** Những hàm này dùng để xử lý các thông tin mà object này sở hữu hay được đổ vào. Nguyên tắc ENCAPSULATION – hàm thuộc về object. Object có **sổ-dư-tài-khoản** thì có hàm **rútTiền()**. **PUBLIC/PRIVATE** tùy chỉnh theo nhu cầu cấp quyền được sờ - **Access Specifier/Modifier!!!**
9. **ĐÚC TƯỢNG, TẠO OBJECT.** Sang bên sản diễn, mặt sân thì công **main()** | Khai báo biến object/tên tắt thuộc kiểu Nhóm/Khuôn/Class vừa tạo | Gọi toán tử **new** để tạo vùng nhớ mới dùng chứa thông tin object | Gọi phễu để đổ value vào | Nhớ **import kho.Tên-Khuôn-Class**. Ví dụ: Dog chiHu = new Dog("Chi Hu Hu", 2021, 0.5);
10. **YÊU CẦU OBJECT LÀM GÌ ĐÓ QUA CHẤM VÀ BUNG LỰA.** Dùng dấu **CHẤM** qua biến object để sờ vào bên trong vùng được **new** mà biến object đang trỏ đến. Dấu **CHẤM** để triệu gọi/giao tiếp/gửi thông điệp yêu cầu/gọi các hàm của object mình muốn tham chiếu. Hành động thuộc về object – ENCAPSULATION. Ví dụ: ngaoDa.bark(); svX.showTranscript();
11. **ÉP SÁU (F6) VÀ TẬN HƯỞNG KẾT QUẢ – XEM APP CHẠY – AHIHI ☺.** Toàn bộ code của APP, gồm Khuôn/Class và các lệnh tạo object new...(), gọi hành động của object qua biến object **CHẤM** được tải vào vùng RAM **CODE-segment**. Lệnh new...() sẽ **clone** Khuôn/Class vào vùng RAM **HEAP-segment** để hình thành nên object với đầy đủ data và hàm xử lý data. Các lệnh gọi hàm của object sẽ lần lượt chạy để cho ra kết quả xử lý như developer đã thiết kế và lập trình./

HAPPY CODE – HAPPY MONEY

© 2021 – Nguyễn Thế Hoàng – FPT University

02 Interface – Bộ Tiêu Chuẩn/Cam Kết/Quy Tắc – CLB

1. **INTERFACE:** là một **cuộc chơi** giữa các bên có liên quan, chơi theo một cách thức định trước (chung giao tiếp, protocol)
2. **INTERFACE:** là một **bộ tiêu chuẩn chung**, **liệt kê** các tiêu chí cần có, liệt kê các tiêu chuẩn, các cam kết, các khế ước mà các bên đối tác, các bên liên quan sẽ chấp nhận **thực thi** các điều khoản này ở giai đoạn sau này (**implement**). Điều này sẽ giúp các bên liên quan dễ giao tiếp, dễ chấp nhận nhau hơn, dễ liên thông, dễ chuyển giao thông tin và xử lý cho nhau. **MỤC ĐÍCH: TẠO SÂN CHƠI CHUNG!!!**
3. **INTERFACE:** là một **CLB** có các tiêu chí hoạt động "chung chung" (hành động **abstract**), gom những người "khác biệt" nhưng chỉ cần **"chung"** lý tưởng/hành động (chung hàm – không chung dữ liệu). Ai cùng chí hướng thì mời tham gia (**implement**)
4. **INTERFACE:** Bộ tiêu chí/yêu cầu – **abstract** của **CLB** (chỉ nói rằng có vậy vậy hoy, ý tưởng hoy, chưa thêm làm, chỉ là cần có cam kết hoy mà). Để anh em **hội viên** theo đó mà hành động sau (implement/cái gì cụ thể thì hội viên sẽ phải làm)
5. **INTERFACE:** là class **"CHA"** đặc biệt, chứa hàm ko có code (**abstract**). Các **"CON"** khi sinh ra sau này (hội viên CLB, các bên cùng chơi) **PHẢI** có code cho hàm abstract của CHA – **implement**. Đòi CHA nêu ý kiến, đòi CON thực thi
6. **Vì các CON có code khác nhau cho CÙNG hàm abstract của CHA.** Khi CHA gọi hàm, các CON hưởng ứng theo cách của mình, đa hình, đa xạ, 50 sắc thái, từ một ra nhiều, transformer biến hình theo CON, **POLYMORPHISM**, và CON được ưu tiên hơn CHA, gọi CHA mà CON lại chạy, CON qua mặt CHA, CHA chỉ còn là lãnh đạo tinh thần hoy – **@Override**
7. **Ai chấp nhận chơi với Interface** (có sử dụng biến kiểu Interface), ai chấp nhận sử dụng sản phẩm xuất phát từ Interface, chơi với các đối tượng đến từ 1 CLB nghĩa là kẻ đó trong tương lai đã sẽ chấp nhận chơi với các cách xử lý khác nhau nhưng đứng chung 1 tên ăn theo tên của hành động của CHA/Interface/CLB (khác **implement** nhưng **cùng tên hàm**). Phụ thuộc vào Interface là giúp ta đã dễ dàng và linh hoạt với các xử lý/hành động/ứng xử của người khác trong tương lai. **High cohesion, low/loose coupling**. Thay vì phụ thuộc vào cái cụ thể, ta phụ thuộc vào cái trừu tượng/abstract sẽ giúp ta linh hoạt với mọi cái khác biệt sẽ diễn ra sau này (implement) – **DEPENDENCY INJECTION** (xem **YouTube** giáo.làng...)

© 2021 – Nguyễn Thế Hoàng – FPT University



Biểu thức Lambda – Chỉ Còn Cái Dây Nịt

1. **ĐỊNH NGHĨA BIỂU THỨC LAMBDA:** Lambda Expression là một hàm không có tên (**Anonymous Method/Function**), dùng thay thế cho **Anonymous Class** trong việc hiện thực hóa đối tượng/implement thuộc về **Functional Interface** (Interface chỉ có duy nhất một hàm **abstract**). Là một cách cài đặt code cho một object thuộc về một Functional Interface, viết **siêu ngắn gọn** đến mức bỏ luôn cả tên hàm **@Override**, chỉ còn lại cái dây nịt – chỉ còn cái **tên tham số** và cái **body of method**
2. **CÁC CÁCH CHƠI VỚI ABSTRACT/KẾ THỪA/INTERFACE:** Khai Cha new Con(); Khai Con new Con(); Khai Cha new Cha() – nếu Cha có abstract thì Anonymous Class xuất hiện. Nếu Cha-Interface chỉ có duy nhất 1 hàm abstract thì **Lambda Expression** xuất hiện
3. **CHƠI VỚI LAMBDA EXPRESSION THẾ NÀO:**
 1. Tạo Functional Interface của riêng bạn hoặc sử dụng các Functional Interface có sẵn đâu đó trong JDK, trong các thư viện của nhà người ta (**.jar**). Functional Interface là interface chỉ có 1 hàm abstract duy nhất. Ví dụ: **Comparator**, **Runnable**, ...
 2. Tạo Anonymous Inner Class là cách truyền thống, chờ IDE tool gợi ý bấm nút **chuyển đổi** sang biểu thức Lambda
 3. Chủ động sử dụng biểu thức Lambda thay cho **Anonymous Inner Class**, thực ra là ta **FOCUS** vào việc viết code cho 1 hàm abstract duy nhất, chỉ tập trung vào hàm, loại bỏ tất cả cái rườ rịa: new, tên class, tên hàm. Chỉ tập trung cái **dây nịt** còn lại: **đầu vào hàm + body của hàm** (code trong hàm)

© 2021 – Nguyễn Thế Hoàng – FPT University



Biểu thức Lambda – Chỉ Còn Cái Dây Nịt (tt.)

4. CÁC HÌNH DẠNG CỦA BIỂU THỨC LAMBDA

Có thể viết lambda expression bằng nhiều cách tùy thuộc vào cấu trúc hàm abstract cần cài đặt bên phía Functional Interface.

Template chung của biểu thức Lambda:

(tham số/đầu vào của hàm abstract) -> {code cần cài đặt cho hàm abstract}

- **Kiểu dữ liệu truyền vào:** không cần, compiler tự suy luận từ tên hàm abstract của Functional Interface
- **Dấu ngoặc tròn ở tham số của hàm ():** nếu hàm có một tham số thì có thể bỏ qua dấu ngoặc, tham số đứng một mình okie
- **Dấu ngoặc nhọn ở thân hàm { }:** nếu thân hàm hay biểu thức lambda chỉ có 1 lệnh thì có thể bỏ luôn { }
- **Lệnh return trong hàm:** nếu thân hàm chỉ có 1 lệnh thì có thể bỏ luôn **return**. Bắt buộc phải sử dụng **return** khi thân hàm/biểu thức Lambda chứa nhiều hơn 1 câu lệnh

5. CÁC CÁCH VIẾT LAMBDA TIÊU BIỂU

- `() -> expression` // Không tham số, một ngoặc tròn, một lệnh, không dấu ;
- `biến-obj-x -> expression` // Một tham số, không ngoặc tròn, một lệnh, không dấu ;
- `(biến-obj-x) -> expression` // Một tham số, một ngoặc tròn, một lệnh, không dấu ;
- `(arguments) -> expression` // Nhiều tham số cách nhau dấu phẩy, một lệnh, không dấu ;
- `(arg1, arg2, ...) -> {` // Nhiều tham số cách nhau dấu phẩy, ngoặc nhọn thân hàm {

`body-statements; // 1 hoặc nhiều lệnh, dấu ; như hàm bình thường`

`return value; // Nếu hàm cần return thì viết như hàm bình thường`

`}`

© 2021 – Nguyễn Thế Hoàng – FPT University



Biểu thức Lambda – Chỉ Còn Cái Dây Nịt (tt.)

6. TẠO CLASS RỒI VÀ TẠO OBJECT:

```
public class XYZ implements FunctionalInterfaceXXX {

    @Override
    public return-type hàm-F(argument-list) {
        body-code
        viết code, cài đặt cho hàm abstract của
        FunctionalInterfaceXXX
    }
}

FunctionalInterfaceXXX xxx = new XYZ();
```

8. BIỂU THỨC LAMBDA – DẸP LUÔN CÁI TÊN HÀM

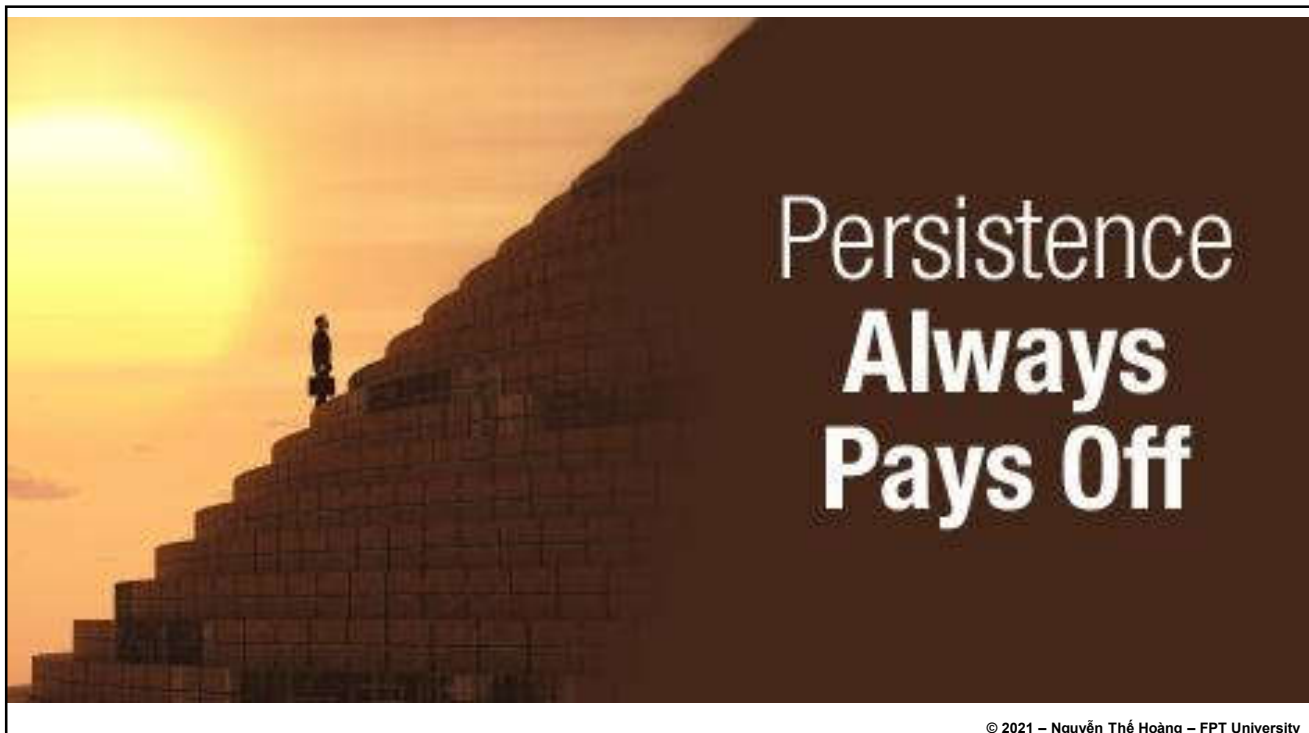
```
FunctionalInterfaceXXX xxx = (argument-list) -> {body}

FunctionalInterfaceXXX xxx = (NO-argument) -> {body}
```

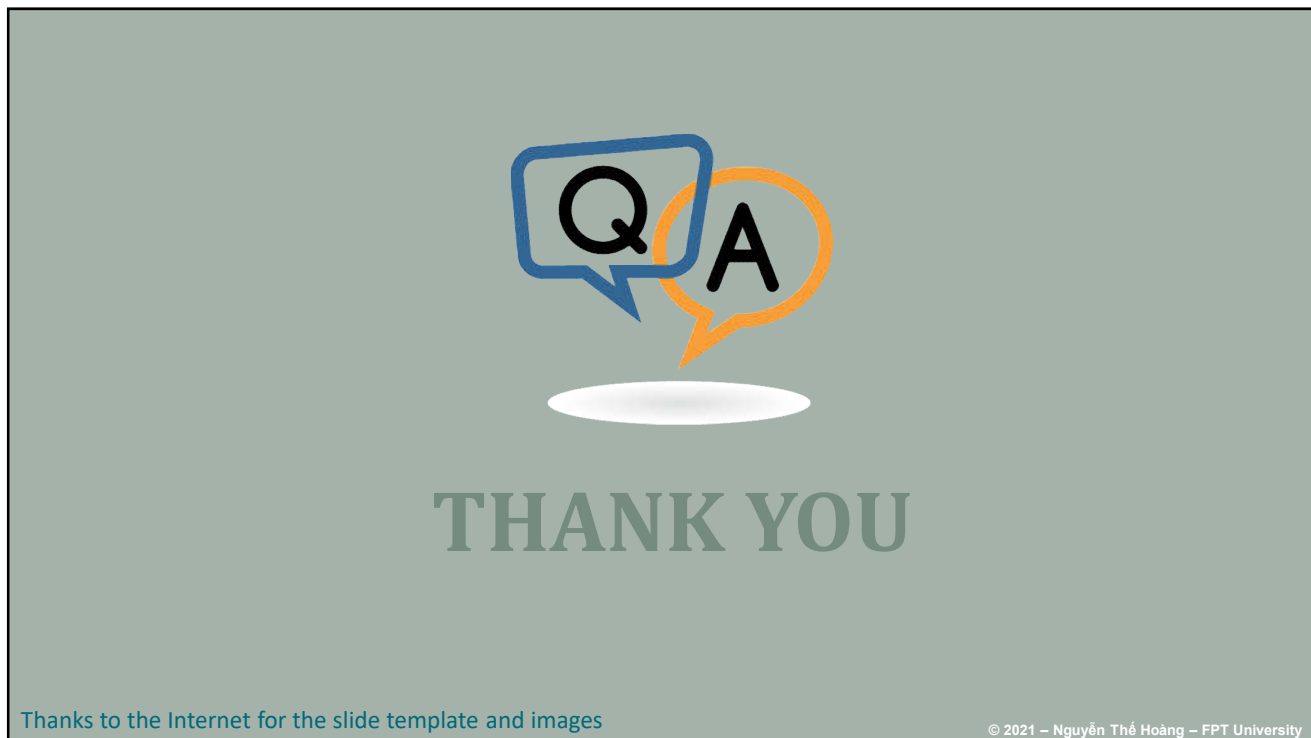
7. TẠO OBJECT QUA ANONYMOUS CLASS:

```
FunctionalInterfaceXXX xxx = new FunctionalInterfaceXXX {
    @Override
    public return-type hàm-F(argument-list) {
        body-code
        viết code, cài đặt cho hàm abstract của
        FunctionalInterfaceXXX
    }
}; //VIP semicolon
```

© 2021 – Nguyễn Thế Hoàng – FPT University



© 2021 – Nguyễn Thế Hoàng – FPT University

[illegible]