# QUY CHUẨN ĐẶT TÊN:

1. PascalCase 🡪 Tên lớp. Vd: UserClass, CategoryClass…
2. camelCase 🡪 Tên hàm, tên phương thức, tên biến. Vd: getUser, …
3. UPPER\_CASE 🡪 Tên hằng số. Vd: DISCOUNT\_PERCENT, …
4. snake\_case 🡪 Tên bảng, tên cột CSDL SQL. Vd: company\_name, …
5. kabab-case 🡪 Tên file, tên UsRL . Vd: style-mobile.css, …

\* Chú ý:

- Không đặt tên biến bắt đầu bằng số.

- Không chứa ký tự đặc biệt.

- Không dùng trùng function mặc định của ngôn ngữ.

# MOOC – COURSE ONLINE:

1. Udemy (SS a Trường Acc)
2. SkillShare (Khuyết Danh)
3. Linkedin (Khuyết Danh)
4. Edx.
5. Coursera.
6. Youtube.

# FREELANCER

## Platform

1. Upwork.
2. Vlance.
3. Fiver.

## Job Title

1. Data Entry: Nhập liệu.

# CÁC PHÍM TẮT QUAN TRỌNG TRONG VISUAL CODE:

1. F12 🡪 Đi đến hàm đang dùng.
2. Shift + F12 🡪 Đi đến những nơi hàm này được dùng.
3. Ctrl + ` 🡪 Mở Terminal.
4. Ctrl + / 🡪 Comment.
5. Alt + < 🡪 Trở về
6. Ctrl + p 🡪 Tìm mở file.
7. Ctrl + p + > 🡪 Tìm command của Visual Code hoặc của Extension.
8. Ctrl + p + # 🡪 Tìm hàm
9. Ctrl + p + : hoặc Ctrl + g 🡪 Tìm hàng
10. Ctrl + f 🡪 Find.
11. Ctrl + Shift + f 🡪 Find all.
12. Ctrl + h 🡪 Find and replace.
13. Ctrl + Shift + h 🡪 Find and replace.
14. Ctrl + Shift + k 🡪 Xóa 1 dòng.
15. Alt + Click 🡪 Chọn nhiều con trỏ sửa cùng lúc.
16. Alt + Up/Down🡪 Di chuyển lên trên hoặc xuống dưới.
17. Alt + Shift + Up/Down 🡪 Copy lên trên hoặc xuống dưới.
18. Alt + Shift + I 🡪 Chọn nhiều con trỏ ở cuối hàng.
19. Ctrl + D 🡪 Tìm từng đoạn bôi đen giống nhau.
20. Ctrl + Shift + l 🡪 Tìm tất cả đoạn bôi đen giống nhau.
21. F2 🡪 Đồng bộ đổi tên hàm, biến.
22. Window + . 🡪 Thêm emoji.
23. Alt + Z 🡪 Tự động xuống dòng (Wrap text).
24. Ctrl + K + 0 🡪 fold all levels (namespace, class, method, and block)
25. Ctrl + K + 1 🡪 Namspace
26. Ctrl + K + 2 🡪 Class
27. Ctrl + K + 3 🡪 Methods
28. Ctrl + K + 4 🡪 Blocks.
29. Ctrl + K + [ hoặc Ctrl + K + ] 🡪 Current cursor block
30. Ctrl + K + J 🡪 Unfold All.
31. Ctrl + K + S 🡪 Show all shortcut.

# CÁC LỆNH CƠ BẢN CỦA TERMINAL, POWER SHELL, CMD, …

1. code . 🡪 mở visual code trong thư mục.
2. cd *tên-thư-mục* 🡪 mở thư mục con.
3. cd .. 🡪 trở về thư mục cha.
4. Clear 🡪 xóa nội dung terminal.
5. Exit 🡪 đóng terminal.
6. Gõ cmd trên thanh địa chỉ 🡪 mở terminal ở tại folder đó.

# GIT CƠ BẢN

- Lên trang gitHub/gitLab tạo 1 Remote Repository. (Lưu ý: tên Repo === tên project folder)

1. git –version 🡪 Kiểm tra phiên bản và git đã được cài chưa.
2. git init 🡪 Tạo lcal repo trong máy tính (file .git). (Nếu clone về thì ko cần init)
3. git status 🡪 Kiểm tra tình trạng code thay đổi so với repo.
4. git add *tên-file* 🡪 File đã thay đổi vào trong repo.
5. git add . 🡪 Thêm tất cả thay đổi.
6. git reset 🡪 Gỡ code thay đổi ra khỏi repo.
7. git commit -m *”Ghi-chú”* 🡪 Lưu code đã add.
8. git commit 🡪 Lưu code khi giải quyết conflict.
9. git log 🡪 Xem lại tất cả commit.
10. git log –oneline 🡪 Hiện gọn hơn (id commit ở đầu tiên).
11. git checkout *idcommit* 🡪 Chuyển đổi giữ các commi.
12. git branch -b new*branchname* 🡪 Tạo branch mới.
13. git branch -d *branchname* 🡪 Xóa branch.
14. git branch 🡪 hiện tất cả branch.
15. git checkout *branchname* 🡪 Chuyển đổi giữa các branch (mặc định: main).
16. git merge *branch1* 🡪 Gộp branch1 vào branch hiện tại.
17. git push 🡪 Đẩy code từ local repo lên remote repo. (Nếu push lần đầu cần upstream)
18. git clone *<git repo link>* 🡪 Lấy remote repo từ trên server về máy. (Nếu init thì ko cần clone về)
19. git pull 🡪 Kéo code từ remote repo về local repo.
20. git stash 🡪 Recover lại commit đã lưu

# HOW TO CODE

1. Have a clear GOAL.
2. DON’T copy code.
3. DO tasks/ project/ practices.
4. Create your own style.
5. Calmdown.
6. Learn what needed. (Học cái gì mới phải biết ích lợi để làm gì)
7. Multitasking >< Asynchronous \*

# SEO SKILL

1. Back links 🡪 Link từ nơi khác.
2. Slug key 🡪 [Www.Webname.Com/***Slug-Slug-Slug***/](http://www.webname.com/slug-slug-slug/)
3. H1 key 🡪 Tối đa 1 H1 trong page để làm tiêu đề.
4. Content is king.
5. Key được lập lại nhiều lần trong page.
6. Alt key, src name key….

# GLOSSARY / TERMINOLOGY

1. Attributes 🡪 Thuộc tính (Dùng trong CSS)
2. Properties 🡪 Đặc tính (Object.Property = value)
3. Methods 🡪 Phương thức (function của object - Object.Method())
4. Elements 🡪 Phần tử (id, class, tag HTML – DOM Js)
5. Parameter 🡪 Tham số
6. Argument 🡪 Đối số
7. Variable scope 🡪 Phạm vi biến – Global, Public, Private,…. Toàn cục, nội bộ…
8. API 🡪 Application Programming Interface.
9. Encode >< Decode.
10. Stringify >< Parse.
11. Callback Hell 🡪 Cần điều kiện của hàm trước đó để chạy hàm phía sau.
12. Chain 🡪 Object.Property.Method().Method().Property.Property…..

# HTML

1. ID > Class > Tag

# CSS

1. Selector 🡪 Cú pháp thứ tự để chon
2. Inline > Internal > External

# JAVASCRIPT

1. Closure (js) 🡪 Hàm có thể truy cập biến của hàm khác.
2. DOM (js) 🡪 API liên kết giữa Js và browser html

## Dom

1. *property:* document.querySelector(‘*Tên .class/#id/selector*’).*element*
2. *property:* document.querySelectorAll(‘*Tên .class/#id/selector*’).*element*
3. *property*: document.createElement(‘Tên *tag*’) – thêm một thẻ tag mới
4. *element*: textContent – nội dung text trong thẻ tag html
5. *element*: innerHTML – nội dung code html trong thẻ tag html­­­­­
6. *element*: insertAdjacentHTML – thêm nối tiếp nội dung code html
7. *element*: classlist – thêm/xóa class.
8. *element*: prepend/append – di chuyển vào đầu/cuối trong thẻ tag code html
9. Dom tree: nodes
10. NodeList: No update. Vd: document.querySelectorAll
11. HTMLCollection: Update. Vd: document.getElementsByTagName

## Array

1. Methods: slice, splice, reverse, concat, join.
2. Methods: foreach, map, filter, reduce.

## Function

1. Regular/Declaration: functionName(parameter1, parameter2, parameter3){ *code* }
2. Expression: const fuctionName = function(parameter1, parameter2, parameter3) { *code* }
3. Arrow: functionName = (parameter1, parameter2, parameter3) => { code }
4. Callback function: Hàm được gán làm tham số của hàm khác

## Class (OOP)

1. Object Class
2. Function Class
3. CS6 Class

## Asyncronous

1. Callback.
2. Promise.
3. Async and Await.
4. Pending / Fulfilled >< Rejected. (3 types)
5. Resolve >< Rejec. (Promise)
6. Catch >< Then / Final (Promise)
7. Try >< Catch (Async Await)
8. Promise.all (Promise / Async Await)

# JAVA

## Theory

1. OOP 🡪 Hướng đối tượng
2. Complied 🡪 Biên dịch qua ngôn ngữ máy (Có thể chạy trên nhiều loại thiết bị, hệ điều hành)
3. Auto Memory Management 🡪 Quản lý bộ nhớ, con trỏ (tự xóa khi không dùng)
4. Case sensitive 🡪 Upper Case and Lower Case are different.
5. JDK 🡪 Java Development Kit
6. JVM 🡪 Java Virtual Machine
7. IDE 🡪 Integrated Development Environment
8. Divide and Conquer 🡪 Chia để trị ??
9. Control Flow / Scope

## Static 🡪 static variables, static methods, static blocks, static class, import static.

1. Lưu trong vùng nhớ heap – có static riêng.
2. Có thể sự dụng Access Modifier.
3. Phạm vi truy cập trong block được khai báo hoặc thông qua block chứa nó (Class.)
4. Không bị xóa trong trong phạm vi block được khai báo.
5. Ngược lại là non-static.

## Global 🡪 *Java không có*

1. Lưu trong vùng nhớ heap (Ngẫu nhiên)
2. Có thể sự dụng Access Modifier.
3. Phạm vi truy cập trong block được khai báo hoặc thông qua object được tạo (*new class()*).
4. Bị xóa theo object được tạo bị xóa .

**Instance 🡪**

1. Khai báo mà không đặt tên cụ thể (để mặc định)

**Local 🡪**

1. Lưu trong vùng nhớ stack (Last in first out / Theo thứ tự)
2. Không sử dụng được Access Modifier.
3. Phạm vi truy cập chỉ trong block được khai báo.
4. Là non-static.

**Final** 🡪

1. Lưu trong vùng nhớ có địa chỉ cụ thể.

## Khai báo hằng số

## Không thể thay đổi chỉ có thể thay đổi bằng cách ghi đè (overriding)

## Access Modifier 🡪 private, (default), protected, public. (\*Xem ở OOP)

## Method 🡪 Phương thức (Trong class)

1. Nhóm các đoạn code thực hiện một chức năng lại và gọi ra khi cần thực thi với tên tùy chọn.
2. Chức năng tương tự như hàm nhưng hàm không dùng trong class.
3. Build-in method 🡪 những thức được dựng sẵn từ thư viện….
4. Main method 🡪 phương thức chính mà chương trình sẽ thực thi từ trên xuống dưới.
5. Static method 🡪 ??
6. Final method 🡪 ??
7. Public method 🡪 ??

## Variable/Data type

1. Primitive:

* Boolean 🡪 isDone = false
* Byte 🡪 8 Bit
* Short 🡪 2 Byte
* Int 🡪 4 Byte
* Float 🡪 4 Byte
* Long 🡪 8 Byte
* Double 🡪 8 Byte
* Char

1. Reference/Object:

* Array
* Class
* String

1. Casting 🡺 Ép kiểu.
2. Static Variable 🡪 ??

## Opeartor

1. Arithmetic 🡪 + - \* / % ++ --
2. Bitwise 🡪 & |^ ~ << >> >>>
3. Assignment 🡪 = += -= \*= /= %= &= |= ^= >>= <<=
4. Comparison/Relational 🡪 == != > < >= <=
5. Logical 🡪 && || !

## Constant

* Final

## Type Conversion

1. Implicit 🡪 Byte > Short > Int > Long > Float Double
2. Explicit 🡪 int a = (int) b

## Condition

1. If-Else If- Else
2. Switch-Case

## Loop

1. While - Do While
2. For - For Each

## Break - Continue

## Array

1. Single Dimensional 🡪 []
2. Mutil Dimensional 🡪 [][]
3. clone()
4. Static
5. Main/Void
6. Try Catch Finally Throw
7. Implements

## Comment 🡪 // or /\* \*/

## String

1. Escape Sequences 🡪 Thoát chuỗi: \n \t \s …

# OOP (Object Oriented Programming)

## ABSTRACTION 🡪 Tính trừu tượng: Ẩn các chi tiết không cần thiết, cho cái nhìn tổng thể về object.

1. Cấu trúc: Tạo ra một khung sườn dễ sử dụng hơn không cần hiểu rõ code sâu bên trong.
2. Bảo mật: Ngăn chặn sử dụng chương trình sai mục đích.

## ENSCAPSULATION 🡪 Tính đóng gói: Tạo các properties hoặc methods riêng cho object. (private)

1. Bảo mật: Ngăn đối tượng bị truy cập từ code bên ngoài vào.

## INHERITANCE 🡪 Tính kế thừa: Tái sử dụng các properties hoặc methods của object cha cho con.

1. Cải tiến: Dễ dàng nâng cấp chương trình.

## POLYMORPHISM 🡪 Tính đa hình: Ghi đè lên các methods của object cha.

1. Luôn đi chung với tính kế thừa.
2. ??

## Object

1. Property
2. Method
3. Prototype 🡪 Là các properties và methods được kế thừa (Javascript)

## Class/Function/Object 🡪 Abstraction

## Abtract Class 🡪 Abstraction

## Interface 🡪 Abstraction + Mutil Inheritance

1. Luôn là public static final.

## Access Modifier 🡪 Khả năng truy cập biến hoặc phương thức hoặc lớp. (class only Public or Default )

1. Package:

* Là folder chứa nhiều class, inerface, package khác (NameSpace).
* Có thể là built-in hoặc user defined.

1. Private 🡪 Chỉ trong class được khái báo. (Trừ inner class nếu được khai báo new)
2. (Default) 🡪 Trong class và cả package của nó và cả con của nó
3. Protected 🡪 Giống default nhưng cho phép truy cập class con của package khác nếu dùng kế thừa (extends + this.).
4. Public 🡪 Tất cả kể cả package khác.

## Final­­­ 🡪 Polymorphism

## Extends 🡪 Inheritance

## Super() 🡪 Inheritance

## this. 🡪 Instance

## new 🡪 Instance

## getter / setter 🡪 Enscapsulation

## Constructor Function 🡪 new Function() (Javascript)