

Th.S LÊ NGỌC THẠCH

Lời nhắn

eBook "**Chạm tới GO trong 10 ngày**" này dự kiến phát hành vào ngày 31/03/2022. Bạn có thể đặt hàng ngay bây giờ với ưu đãi giảm 50% chỉ **199K**, tiết kiệm 200K. Thanh toán nhanh theo 2 cách:

① MoMo

0908550642  Lê Ngọc Thạch

 Nội dung tin nhắn: GO2021 email sdt  
Ví dụ: abc@gmail.com 0908550642  
**Email và sdt của người nhận eBook.**

Trường hợp tặng bạn bè thì ghi thông tin email và sdt của bạn.

Quét mã QR thanh toán 199K.



② Chuyển khoản

Lê Ngọc Thạch, Ngân Hàng Tiên Phong, CN HCM  
Số tài khoản: 00002888001  
Nội dung tin nhắn: GO2021 email sdt  
Vd tin nhắn: GO2021 abc@gmail.com 0908456321  
Quét mã QR để thanh toán cho:



Quét mã vạch này để giao dịch

Ngoài ra, bạn có thể đọc ngay bản nháp hiện tại với giá 0đ theo cách sau:

Cài **App MinePI** cho điện thoại tại theo link:

<https://minepi.com/thachln>

Sử dụng invitation code: **thachln**

Liên lạc với tác giả qua <https://facebook.com/ThachLN> để cung cấp account MinePI, SĐT và Email nhận nhận eBook với thông tin mã hóa đính kèm.

Lê Ngọc Thạch

# CHẠM TỚI GO TRONG 10 NGÀY

## Mục lục

Mục lục .....	2
Quy ước.....	12
Mã nguồn.....	12
Lệnh thực thi trong hệ điều hành.....	12
Đường dẫn hiện hành.....	12
Hệ điều hành Windows và Linux/Mac .....	12
Cấp dấu nhảy.....	13
Cách viết trình tự bấm chọn menu .....	13
Đường dẫn thư mục (Path) .....	13
Các từ viết tắt, tiếng Anh thường xuyên được sử dụng trong sách.....	13
Cách viết dấu chấm câu, ghi chú hình, bảng biểu.....	14
Ôn tập kiến thức cơ bản về máy tính và phần mềm.....	16
Ôn tập #1: Quá trình tiến hóa của các mô hình phần mềm.....	17
Phần mềm trên máy cá nhân.....	18
Máy vi tính cá nhân (personal computer).....	18
Giao diện console .....	19
Giao diện đồ họa (GUI – Graphics User Interface) .....	21
Phần mềm trên mạng nội bộ .....	23
Mạng nội bộ (LAN - Local Network).....	23
Phần mềm trên nền tảng mạng Internet.....	25
Mạng Internet.....	25
Phần mềm trên mạng Internet.....	25
Ôn tập #2 – Cấu trúc của phần mềm.....	27
Công thức I + P + O.....	28
Thu nhận thông tin.....	28

Xử lý thông tin .....	28
Xuất kết quả.....	28
Ví dụ.....	28
Ngày 1: Làm quen với GOLANG .....	30
Thử thách trong chương 1 .....	<b>Error! Bookmark not defined.</b>
Bài 1 – Tại sao GO ra đời .....	31
Bài 2: Ngôn ngữ lập trình GO .....	32
Biến (Variable), Cấu trúc (Structure).....	32
Variable có nghĩa là gì? .....	34
Khai báo biến (variable declaration) .....	35
Lệnh gán (assign).....	36
Bài 3 – Chuẩn bị môi trường lập trình.....	38
GO Core .....	38
Cài thêm thư viện .....	38
Visual Code .....	39
Cài GO trên Ubuntu .....	41
Bài 4 – Viết chương trình đơn giản với GO .....	42
Viết mã.....	42
Biên dịch.....	42
Chạy trực tiếp mã nguồn.....	44
Phép gán (assign).....	46
Các toán tử cơ bản.....	48
Hàm (function).....	49
Chạy chương trình có tham số dòng lệnh trong Visual Code.....	50
Lấy tham số từ dòng lệnh .....	50
Vòng lặp (loops) .....	51
Nâng cao .....	52
Bài 5 – Biểu diễn thông tin đơn giản với GO.....	54
Kiểu chuỗi (string).....	54
Xem kiểu dữ liệu của biến.....	55
Kiểu dữ liệu số (Numeric data types) .....	55
Viết chương trình Fibonacci.....	60

Mảng (arrays).....	61
Slice – Mảng không giới hạn độ dài .....	64
Maps.....	69
Thời gian (Times & dates).....	70
Tra cứu định dạng.....	72
Bài tập cuối ngày 1 .....	<b>Error! Bookmark not defined.</b>
Ngày 2 – Cách viết một phần mềm đơn giản.....	74
Bài 1 – Phần mềm đầu tiên – Cài đặt phép toán cộng.....	75
Bước 1: Xác định và viết yêu cầu.....	76
Bước 2: Làm thiết kế.....	79
Bước 3: Lập trình và kiểm thử.....	86
Bước 3.1: Kiểm thử mã nguồn.....	108
Bước 3.2: Hoàn thiện giao diện.....	110
Bước 4: Kiểm thử hệ thống.....	111
Thử thách cho bạn: #1 .....	116
Thử thách cho bạn: #2.....	116
Phiên bản 1: Hỗ trợ truyền văn bản trên dòng lệnh.....	116
Vd: lệnh.....	117
Tham khảo thêm source code bằng CSharp:.....	117
Tham khảo source code bằng C.....	118
Tham khảo thêm source code bằng Python.....	119
Thử thách cho ngày 2.....	120
Ngày 3: Biểu diễn thông tin phức hợp.....	123
Bài 1 – Biểu diễn thông tin phức hợp với GO.....	124
Cấu trúc (Structure).....	124
Kết hợp Slice và Structure.....	124
Con trỏ (Pointer).....	126
Tuples (Bộ dữ liệu).....	128
Đọc thêm và thực hành.....	131
Chuỗi (String) .....	131
Regular expressions and pattern matching .....	132
Bài 2 – Viết hàm cho cấu trúc .....	134

Phân tích hàm calculateBMI cho struct Employee .....	134
Tổ chức thành thư viện (module) .....	136
Bài 3 – Dữ liệu dạng JSON .....	<b>Error! Bookmark not defined.</b>
Đọc dữ liệu JSON .....	<b>Error! Bookmark not defined.</b>
Chương Ngày 4 .....	<b>Error! Bookmark not defined.</b>
Thử thách cho ngày 4 .....	<b>Error! Bookmark not defined.</b>
Chương 5: Cấu trúc điều khiển .....	139
Thử thách trong chương 4 .....	140
Bài 7 – Cấu trúc rẽ nhánh .....	142
Lệnh if .....	142
Switch .....	142
Bài 8 – Vòng lặp .....	145
Vòng lặp (loops) .....	145
Vòng lặp for nâng cao .....	146
Chương 6: Làm việc với dữ liệu trên đĩa cứng .....	151
Bài 1 – Làm việc với thư mục và file .....	152
Lấy thông tin về file/thư mục .....	152
Lấy nội dung thư mục .....	154
Lấy nội dung file .....	154
Lưu file .....	154
Lưu và đọc file mã hóa .....	155
Bài 2 – Làm việc với file CSV .....	159
Bài 3 – Đọc file CSV .....	161
Bài 4 – Ghi file CSV .....	163
Bài 5 – File và cấu trúc (struct) .....	164
Cài đặt thư viện .....	164
Đọc đoạn dữ liệu binary vào mảng các struct .....	164
Đọc file CSV vào mảng các struct .....	165
Ghi mảng các struct ra file CSV .....	168
Bài 6 – Đọc file văn bản .....	169
Bài 7 – Đọc file Excel .....	170
Cài đặt .....	170

Đọc file Excel.....	170
Ghi dữ liệu ra file Excel.....	171
Chương 7: Tổ chức dự án GOLANG.....	173
Bài 1 – Tổ chức mã nguồn.....	174
Bước 1: Tạo file go.mod để mô tả tên của module.....	174
Bước 2: Tạo thư mục và file chứa hàm dùng chung.....	175
Bước 3: Viết chương trình chính.....	175
Bài 2 – Tinh chỉnh mã nguồn.....	183
Phiên bản 0.0.2.....	183
Phiên bản 0.0.3.....	184
Phiên bản 0.0.4.....	186
Bài 3 – Biên dịch dự án.....	188
Bài 3 – Tập thói quen viết phần mềm.....	189
Sử dụng logging.....	189
Bài 4 : Hàm trả về không phải là giá trị.....	193
Bài 3 – Go Packages và Functions.....	194
Anonymous function.....	194
Do it yourself:.....	195
Chương 8: Sử dụng cơ sở dữ liệu PostgreSQL.....	196
Bài 1 – Làm quen với CSDL.....	197
Bài 2 – Sử dụng PostgreSQL portable.....	198
Tải gói binary.....	198
Tạo file khởi động PostgreSQL server.....	199
Khởi động PostgreSQL server.....	200
Tương tác với PostgreSQL Server qua dòng lệnh.....	200
Tương tác với PostgreSQL Server qua web site.....	204
Tương tác với PostgreSQL Server qua web site.....	207
Bài 3 – Thực hành với PostgreSQL.....	208
Tạo một CSDL “ECP”.....	208
Nhập dữ liệu.....	212
Bài 4 – GOLANG và PGSQL.....	215
Cài thư viện.....	215

Ví dụ.....	215
Giải thích code từ ví dụ.....	216
Bài 5 – Sử dụng file cấu hình.....	219
Cài thư viện viper .....	219
Ngày 7: Sử dụng MySQL.....	220
Bài 1 - Tự chuẩn bị MySQL server .....	221
Tải MySQL.....	221
Cấu hình.....	221
Khởi tạo dữ liệu hệ thống.....	221
Thiết lập mật khẩu.....	221
Bài 2 – GOLANG và MySQL.....	224
Sử dụng thư viện go-sql-driver.....	224
Ngày 8: Sử dụng SQL Server.....	233
Bài 1 - Tự chuẩn bị MySQL server .....	234
Bài 2 – GOLANG và SQL Server .....	235
Kết nối SQL Server.....	235
Ngày 8: Interface .....	237
Bài 1 - Interface.....	238
Khái niệm .....	238
Khai báo interface.....	238
Cài đặt interface.....	239
Ngày 9 – Đảm bảo chất lượng mã nguồn .....	242
Đảm bảo mã nguồn trong sáng, dễ chỉnh sửa.....	243
Phát hiện những lỗi có thể nhìn thấy ngay.....	244
Đảm bảo các chức năng nhỏ nhất không có lỗi.....	245
Unit Testing.....	245
Dọn dẹp log files.....	247
Ngày 10: Lập trình đồng thời và song song với GO .....	249
Bài 1 – Khái niệm Concurrency và Parallelism.....	250
Tạo goroutine .....	250
Đợi hàm Goroutine chạy xong.....	252
Sử dụng channel cho goroutine.....	253

Khảo sát thêm ví dụ GoRoutineMessage.go sau: .....	255
Chỉ định rõ channel read-only   write only .....	257
Pipeline .....	257
Bài 2 – Khái niệm Concurrency và Paralleilism .....	259
Bài 3 – Lập trình Concurrency .....	260
Bài 4: Lập trình Paralleilism .....	261
Thử thách ngày 9 .....	262
Nâng cấp chương trình ImportCSV2DB .....	262
Nâng cấp chương trình DataTransform .....	262
Thử thách ngày 10 .....	264
Nâng cao chất lượng mã nguồn .....	264
Ngày 8: GOLANG và C/C++ .....	265
Bài 1 - Lập trình C trong GO .....	266
Ngày 9: Các chủ đề mở rộng/nâng cao .....	267
Bài 1 – Viết hàm với tham số linh động .....	268
Variadic functions .....	268
Bài 2 - Crawl dữ liệu với GOLANG .....	270
Request đơn giản .....	270
Thiết lập timeout cho request .....	270
Thiết lập header .....	271
Download URL .....	273
Use substring .....	274
Bài 3 - Lập trình CUDA với GOLANG .....	276
Bài 4 - Phát triển Web Application với Beego .....	277
Cài đặt GO .....	277
Cài đặt Beego .....	277
Tạo dự án .....	277
Chạy ứng dụng .....	279
Truy cập ứng dụng .....	279
Chỉnh sửa code .....	279
Tạo API .....	281
Bài tham khảo #5 - Phát triển Web Backend với Gin-Gonic .....	282



Cài đặt .....	282
Viết Backend đơn giản.....	282
Thử thách cho bạn.....	285
Triển khai lên server Ubuntu với Nginx .....	287
Nâng cấp ứng dụng Back-end.....	287
Cài đặt các thư viện hỗ trợ web .....	292
Bài 6 - Sử dụng GOLANG trong WSL2 .....	293
Bài 7 – Sử dụng Makefile với GOLANG .....	294
Giới thiệu Makefile .....	294
Ngày 10: Tra cứu theo nhu cầu.....	296
Các API về xử lý chuỗi.....	297
Các API sử dụng GIN GO NIC .....	298
Ngày 11: Testing với GO .....	299
Biên dịch OpenCV từ mã nguồn .....	300
Cài đặt Anaconda: .....	300
Cài đặt mkl-service.....	300
Kiểm tra thông tin thiết bị GPU.....	300
Biên dịch.....	301
Ngày 12 - Blockchain .....	303
Bài 1: Ôn tập kiến thức cơ bản.....	304
Làm quen lại với kiểu Slice của byte .....	304
Thư viện bytes.....	305
Thư viện mã hóa .....	306
Mã hóa base58 .....	306
Khóa công khai và khóa bí mật .....	308
Bài 2 – Tạo cấu trúc chuỗi khối .....	310
Tạo dự án .....	310
Viết mã nguồn main.go .....	310
Bài 3 – Minh họa thuật toán đồng thuận ProofOfWork.....	314
Định nghĩa bổ sung Block.....	314
Hàm tạo Block cũ .....	314
Hàm tạo Block cải tiến.....	314

Cài đặt ProofOfWork (PoW) .....	315
Bài 4 – Lưu trữ Blockchain .....	318
Sử dụng database dạng Key-Value .....	318
Đóng gói OpenSSL .....	320
Chuẩn bị công cụ .....	320
Cài đặt tool Visual Studio 2019.....	320
Clone mã nguồn dự án OmiseGo eWallet.....	321
Lập trình wxWidget .....	323
Phụ lục .....	324
Phụ lục 3 .....	325
Lập trình giao diện với goki .....	326
Cài đặt GCC for Windows .....	326
Cài đặt thư viện goki.....	327
Viết ứng dụng.....	327
Biên dịch và chạy ứng dụng.....	328
GOLANG và QT .....	329
Cài đặt phần mềm QT .....	329
Cài đặt thư viện.....	334
Trải nghiệm lập trình.....	335
Phụ lục 4 .....	337
GOLANG và Google Sheet.....	338
Phụ lục 5 – So sánh tốc độ truy cập dữ liệu giữa GOLANG và Python .....	339
Phụ lục 5 – Sample Project.....	343
The Links 'R'; Us Project .....	344
System overview – what are we going to be building? .....	344
Selecting an SDLC model for our project .....	345
Requirements analysis .....	347
System component modeling.....	354
Summary.....	361
Building a Persistence Layer.....	363
Technical requirements.....	363
Exploring a taxonomy of database systems .....	365

Understanding the need for a data layer abstraction .....	371
Designing the data layer for the link graph component.....	372
Data-Processing Pipelines.....	411
Synchronous versus asynchronous pipelines .....	418
Building a crawler pipeline for the Links 'R' Us project .....	433
Summary .....	451
Thử thách .....	452
Thử thách sau ngày 1 .....	452
Thử thách sau ngày 4 .....	453
Thử thách sau ngày 5 .....	454
Thử thách sau ngày 6 .....	455
Thử thách sau ngày 7 .....	459
Thử thách sau ngày 8 .....	459
Thử thách sau ngày 9 .....	459
Thử thách sau ngày 10 .....	460
Final Project #1 .....	460
Final Project #2.....	464
Final Project #3.....	470
Final Project #4.....	475

## Quy ước

Một số nội dung trong tài liệu được trình bày với các định dạng khác nhau thì có ý nghĩa của nó, bạn đọc nên nắm thông tin này để tiện theo dõi.

## Mã nguồn

Mã lệnh được viết và đóng khung với font chữ **Consolas**, có thanh màu vàng bên trái; và kết quả hiển thị trên màn hình được đóng trong khung màu đỏ bên dưới như sau:

```
package main

import (
    "fmt"
)

func main() {
    name := "Thạch"
    fmt.Println("Hello ", name)
}
```

```
Hello Thạch
```

## Lệnh thực thi trong hệ điều hành

Trường hợp các lệnh thực thi trong môi trường hệ điều hành (phân biệt với các lệnh, hoặc mã nguồn của chương trình thực thi trong môi trường lập trình) thì dấu hiệu có 2 thành màu vàng như sau:

```
Hello.exe "I can do"
```

## Đường dẫn hiện hành

Đôi khi lệnh được hướng dẫn có cả tên ổ đĩa và thư mục và dấu mũi tên như bên dưới (phần chữ mờ). Phần này ý nói là chạy lệnh bên phải dấu mũi tên trong thư mục hiện hành D:\MyGo.

```
D:\MyGo> go build GoArgs.go
```

## Hệ điều hành Windows và Linux/Mac

Các bạn có thể học và làm việc với GOLANG bằng máy tính chạy hệ điều hành Windows, hoặc Linux hay Mac (gọi chung là Linux/Mac). Trong tài liệu khi mô tả các chương trình đã đóng gói - ví dụ file Hello.exe thì bạn hiểu là dành cho người dùng Windows. Đối với các bạn dùng Linux/Mac thì tự hiểu là file Hello.

Đối với thư viện cũng vậy. Trên Windows thì tài liệu sẽ viết là là .dll (vd common.dll) thì các bạn dùng Linux/Mac tự hiểu là file common.o.

## Cặp dấu nháy

Trong NNL T GO, dữ liệu **dạng kí tự** được bao đóng trong cặp **dấu nháy đơn**, dữ liệu **dạng chuỗi** được bao đóng trong **dấu nháy đôi**. Trên bàn phím máy tính thì dấu **nháy trái** và **phải** là giống nhau. Tuy nhiên trong phần mềm soạn thảo văn bản như Microsoft Word thì cặp dấu nháy đơn và đôi được thay thế bằng ‘, ’’ để tăng tính thẩm mỹ. Các dấu nháy thẩm mỹ này khác với kí tự ' và " trên bàn phím (phím bên trái phím Enter).

Đôi khi bạn copy & paste mã nguồn vào các phần mềm như Microsoft Word thì các dấu nháy có thể bị “trang trí” lại như trên. Vì vậy khi copy mã nguồn từ Microsoft vào các công cụ lập trình thì hãy thay thế lại cho đúng.

Một qui ước khác liên quan đến dấu nháy đôi là khi dùng trong văn bản để bao đóng danh từ riêng, hoặc lệnh như hướng dẫn sau: *Bạn hãy thử gõ lệnh “dir” trong cửa sổ TERMINAL để xem nội dung thư mục hiện hành.* Trong câu hướng dẫn này thì lệnh dir được gõ vào cửa sổ TERMINAL **KHÔNG** bao gồm cặp dấu nháy.

## Cách viết trình tự bấm chọn menu

Khi cần trình bày thứ tự các nút bấm, hoặc các mục cần bấm trong các thao tác thì sẽ dùng dấu lớn hơn > để làm biểu tượng như là mũi tên phải. Ví dụ khi hướng dẫn bạn sử dụng phần mềm Visual Code vào menu Run, bấm vào mục “Run Without Debugging” thì sẽ viết gọn như sau:

Vào menu Run > Run Without Debugging.

## Đường dẫn thư mục (Path)

Trong Windows thì dấu cách thư mục là dấu xuyệt trái (back slash). Ví dụ: D:\MyGO\data.

Tuy nhiên ngôn ngữ GO và phần mềm lập trình Visual Code được thiết kế tương thích với các hệ điều hành khác như Macintosh, Linux. Các hệ điều hành thì dùng dấu xuyệt phải (right slash) để phân cách thư mục. Ví dụ: /mnt/d/MyGO.

Vì vậy khi trình bày đường dẫn thư mục trong câu văn thì đôi lúc dùng \, hoặc đôi lúc dùng / do dữ liệu được minh họa trên Windows hoặc Linux/Mac.

Nhưng trong mã nguồn thì đều thống nhất là dùng dấu xuyệt phải / như:

```
read.csv("D:/MyGO/HelloGO.go")
```

## Các từ viết tắt, tiếng Anh thường xuyên được sử dụng trong sách

Viết tắt	Diễn giải
----------	-----------

<b>NNLT</b>	Ngôn ngữ lập trình
<b>Windows</b>	Hệ điều hành Microsoft Windows
<b>OS</b>	Operating System: Hệ điều hành – phần mềm đặc biệt để điều khiển các thiết bị phần cứng của máy tính giúp người dùng vận hành cái máy được thuận tiện.

## Cách viết dấu chấm câu, ghi chú hình, bảng biểu

Tài liệu này sẽ hạn chế tuân thủ cú pháp viết văn thông thường khi sử dụng dấu chấm cuối câu nhưng vẫn đảm bảo người đọc hiểu đúng. Ví dụ trong các đề mục được liệt kê thì đôi lúc không cần viết dấu chấm cho nhanh. Đặc biệt là trong trường hợp có tên file cuối câu.

Trong các hình, hoặc bảng biểu thì sẽ không đánh số. Thay vào đó tôi sẽ dùng những từ như: Hình dưới đây, hình bên dưới; hoặc hình phía trước để bạn đọc có thể hiểu chính xác mà không cần phải mất thêm thời gian ghi và đánh số thứ tự các hình, bảng biểu.

<https://thachln.github.io/ebooks/cham-toi-GO-trong-10-ngay.html>

## Ôn tập kiến thức cơ bản về máy tính và phần mềm

Nếu bạn viết được bài luận trả lời rành mạch các câu hỏi sau thì có thể bỏ qua phần này:

- ? Hãy nêu những khác biệt cơ bản giữa máy vi tính (computer) với các máy móc (machine) thông thường khác.
- ? Bạn hãy giải thích Phần mềm (Software) là gì và phân loại các phần mềm trên máy vi tính mà bạn đang dùng.
- ? Mạng toàn cầu (Internet) là gì? Hãy giải thích về Internet trong mối liên quan với Ứng dụng web (Web Application), và dẫn dắt từ lệnh (command) đơn giản nhất đang có trên máy tính mà bạn đang dùng.
- ? Công thức  $I + P + O$  (gọi tắt IPO) trong lĩnh vực công nghệ thông tin là gì?



## **Ôn tập #1: Quá trình tiến hóa của các mô hình phần mềm**

*Bài này giúp các bạn hình dung các loại phần mềm phổ biến trên máy tính.*

## Phần mềm trên máy cá nhân

### Máy vi tính cá nhân (personal computer)

**Máy tính** hay **máy điện toán** là những thiết bị hay hệ thống thực hiện tự động các phép toán số học dưới dạng số hoặc phép toán logic. Các **máy tính cỡ nhỏ** thường gọi là **máy vi tính**, trong số đó **máy dùng cho cá nhân** thường gọi là **máy tính cá nhân**.

Để một cái máy vi tính hoạt động được thì cần có phần mềm đặc biệt để điều khiển các thiết bị của nó gọi Hệ điều hành (Operating System). Các hệ điều hành phổ biến gồm:

- Microsoft Windows – thường được gọi tắt là Windows. Đây là hệ điều hành của hãng Microsoft. Windows như tên gọi của nó có biểu tượng là cửa sổ. Hình 1 là biểu tượng của 2 phiên bản Windows phổ biến hiện tại.

Hệ điều hành
✓ Là phần mềm đặc biệt để điều khiển máy vi tính



Hình 1: Biểu tượng Windows 7 và 10

- Macintosh - thường gọi tắt là Mac. Đây là hệ điều hành của hãng Apple.



Hình 2: Biểu tượng Quả táo của HĐH Macintosh

- Linux. Là hệ điều hành mã nguồn mở.



Hình 3: Biểu tượng Chim cánh cụt của HĐH Linux

## Giao diện console

Từ những phiên bản Hệ điều hành đầu tiên ra đời cho đến ngày nay thì việc ra lệnh cho máy vi tính thực hiện một công việc nào đó thông qua **cửa sổ gõ lệnh**<sup>1</sup> vẫn phổ biến.

### Ví dụ 1 – Gõ lệnh:

Bạn có thể yêu cầu máy tính thực hiện một lệnh có sẵn trong OS Windows bằng cách mở cửa sổ dấu nhắc lệnh (Nhấn phím Windows + R), gõ cmd. Trong cửa sổ cmd.exe, gõ lệnh:

dir d:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.17134.472]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ThachLN>dir d: _
```

Trong trường hợp bạn gõ một lệnh không có sẵn trong máy tính (vd: mvnc) thì máy tính sẽ báo câu lỗi như sau:

'mvnc' is not recognized as an internal or external command, operable program or batch file.

Dịch sát nghĩa: mvnc không được nhận diện như là một **lệnh bên trong** hoặc **bên ngoài**, một **chương trình có thể hoạt động** hoặc tập tin batch.

#### Internal command

- ✓ Là lệnh có sẵn trong hệ điều hành và được nạp sẵn vào bộ nhớ trong (RAM)

<sup>1</sup> Thuật ngữ tiếng Anh tương ứng có khác nhau trên các HĐH. Trong Windows gọi là “Prompt”. Trong Mac và Linux gọi là Terminal.

Giải thích:

- External command là lệnh bên ngoài (hệ điều hành). Tức là các phần mềm được lưu trữ trong đĩa cứng và được khai báo đường dẫn thư mục chứa nó trong biến môi trường PATH
- Tập tin batch là một tập tin văn bản có đuôi file là .bat và nội dung bên trong file gồm nhiều lệnh (batch có nghĩa là bó | khối | nhóm). Batch file này khi thực thi thì sẽ thực thi lần lượt các lệnh bên trong nó.

Lỗi trên có nghĩa là: "mvnc" không được máy tính hiểu là một lệnh hoặc một chương trình. Lý do có thể là một trong các tình huống sau:

- Nó không phải là lệnh có sẵn trong OS.
- Trong thư mục mà bạn đang gõ lệnh hoặc trong tất cả các thư mục được liệt kê trong biến môi trường PATH không có tồn tại một trong các file có thể thực thi có phần mở rộng như:
  - .com
  - .exe
  - .bat
  - .cmd
  - ...

#### External command

✓ Là lệnh bên ngoài hệ điều hành và không có sẵn trên bộ nhớ trong. Muốn máy tính hiểu lệnh này thì đường dẫn thư mục chứa nó phải được khai báo trong biến môi trường PATH.

#### Bài tập thực hành

1) Trong Windows, mở cửa sổ lệnh "cmd" gõ, quan sát, chụp hình kết quả và ghi chú hiểu biết hoặc suy đoán của các bạn vào một tài liệu để giải thích các lệnh sau:

1. path
2. path /?
3. echo %PATH%
4. set
5. set /?
6. cd

7. dir

8. hostname

9. hi

10. myname

- 2) Trong Linux/Mac, mở cửa sổ lệnh “terminal” gõ, quan sát, chụp hình kết quả và ghi chú hiểu biết hoặc suy đoán của các bạn vào một tài liệu để giải thích các lệnh sau:

11. echo \$PATH

12. set

13. set –help

14. help set

15. pwd

16. ls

17. hostname

18. hi

19. myname

## Giao diện đồ họa (GUI – Graphics User Interface)

Nếu dùng máy tính mà chỉ gõ lệnh không thôi thì rất khó cho người không chuyên. Vì vậy các nhà làm phần mềm nghĩ ra cách để sáng tạo các phần mềm có giao diện đồ họa để người dùng tương tác với máy vi tính dễ dàng hơn.

Các phần mềm phổ biến dạng GUI là:

- Xử lý các công việc văn phòng: Microsoft Word, Microsoft Excel, Microsoft PowerPoint...
- Công cụ viết phần mềm cho dân lập trình: Microsoft Visual Studio, Eclipse, v.v...
- Soạn thảo văn bản đơn giản như: Notepad, Notepad Plus



### Bài tập thực hành

- 1) Hãy google tìm phần mềm “Notepad++” để vào trang chủ “<https://notepad-plus-plus.org/>”. Sau đó tải và cài Notepad++ vào máy tính của bạn.

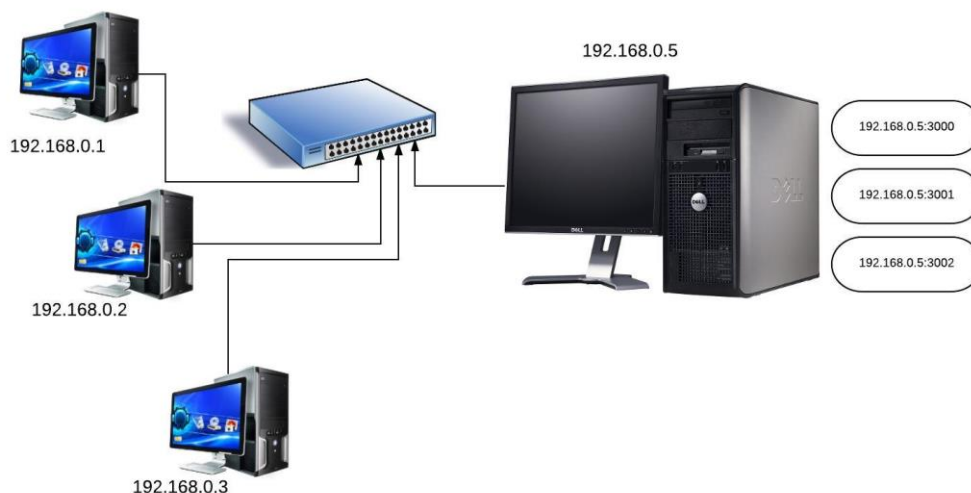
### Hạn chế

Các phần mềm dạng Console hoặc GUI được cài đặt trên máy tính có ưu điểm là người dùng có thể bật máy tính và dùng ngay vì nó đã được cài sẵn trên máy. Tuy nhiên sẽ bất lợi cho các nhà sản xuất phần mềm khi cần nâng cấp phiên bản mới. Thông thường chúng ta phải tải và cài bản nâng cấp khi có phiên bản mới.

## Phần mềm trên mạng nội bộ

### Mạng nội bộ (LAN - Local Network)

Trong một tổ chức có nhiều máy tính được kết nối với nhau thì việc khai thác sức mạnh của các máy tính là cần thiết.



Hình 4: Một mạng máy tính đơn giản

Trong hình 4, các đường kẻ mũi tên chỉ sự kết nối giữa máy tính với một thiết bị trung tâm. Kết nối này có thể là dây cáp (cable) hoặc sóng không dây (Wireless). Phổ biến là Wifi.

Khi các bạn ra quán café kết nối vào Wifi là xem như bạn đã kết nối với mạng nội bộ của quán café.

Để xác định được máy tính của bạn trong một mạng thì người ta dùng địa chỉ IP Address (Internet Protocol Address). IP Address tương tự như địa chỉ nhà của bạn để giúp người đưa thư gửi thư đến đúng nhà bạn.

Để biết địa chỉ máy tính của bạn trong mạng thì gõ lệnh:

```
ipconfig
```

Trên Linux/Mac, gõ lệnh:

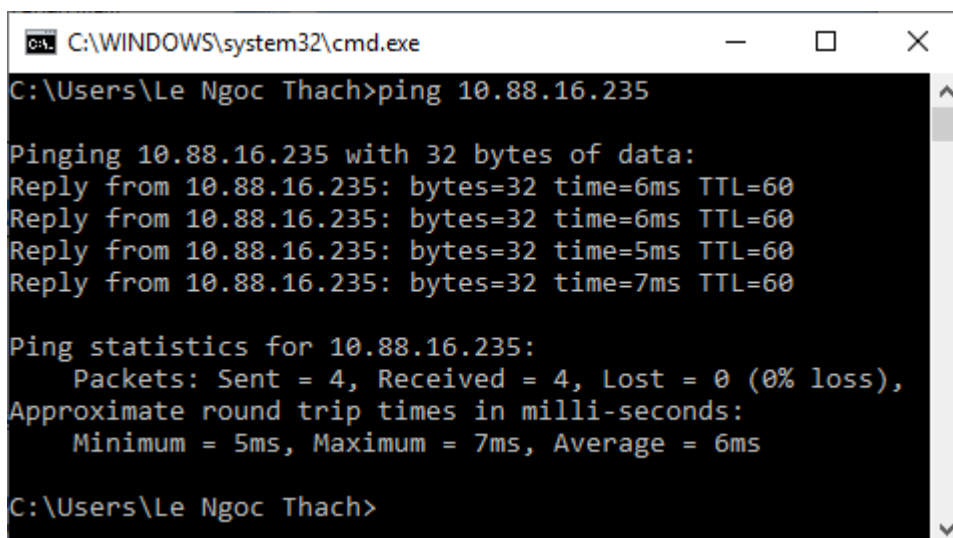
```
ifconfig
```

Để kiểm tra xem máy tính của mình có thể kết nối với một máy tính khác trong mạng nội bộ hay không thì dùng lệnh:

```
ping <ip address>
```

Ví dụ: Để kiểm tra xem máy tính của bạn có thể kết nối với máy tính có địa chỉ IP là 10.88.16.235 thì bạn gõ lệnh:

```
ping 10.88.16.235
```



```
C:\WINDOWS\system32\cmd.exe
C:\Users\Le Ngoc Thach>ping 10.88.16.235

Pinging 10.88.16.235 with 32 bytes of data:
Reply from 10.88.16.235: bytes=32 time=6ms TTL=60
Reply from 10.88.16.235: bytes=32 time=6ms TTL=60
Reply from 10.88.16.235: bytes=32 time=5ms TTL=60
Reply from 10.88.16.235: bytes=32 time=7ms TTL=60

Ping statistics for 10.88.16.235:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 7ms, Average = 6ms

C:\Users\Le Ngoc Thach>
```

#### Bài tập thực hành

- 1) Hãy mượn máy tính của đồng nghiệp hoặc một người đang kết nối vào mạng (mạng dây hoặc mạng Wifi) gõ lệnh “ipconfig”. Ghi lại IP Address của máy tính đó.  
Chú ý:
  - Nếu dùng mạng Wifi thì hãy quan sát địa chỉ của mạng Wifi (Tìm dòng nào có chữ tương tự: Wireless LAN adapter Wi-Fi)
- 2) Ngồi trên máy tính của mình gõ lệnh: ping <địa chỉ ip máy tính của đồng nghiệp>

Một bước cải tiến trong lĩnh vực phần mềm là thay vì phát triển các ứng dụng để chạy trên một máy vi tính – dùng giao diện CONSOLE hoặc GUI, thì giới lập trình có thể phát triển phần mềm đặt trên một cái máy nào đó (gọi là server) trong mạng. Người dùng có thể ngồi trên một máy tính khác nhưng vẫn có thể dùng được phần mềm trên server.



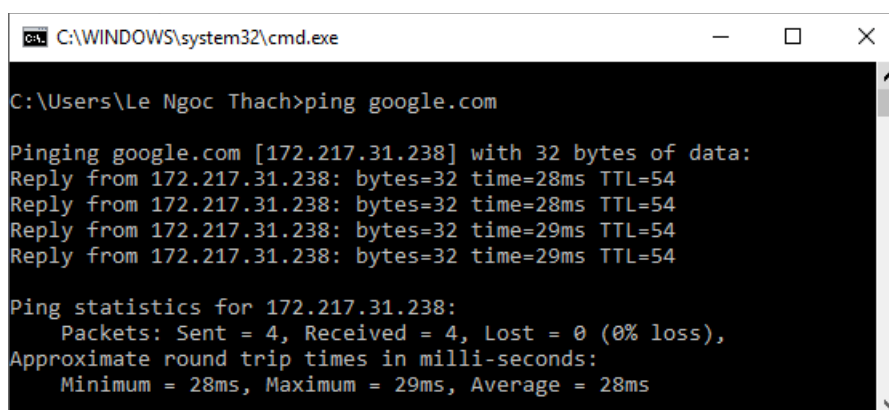
## Phần mềm trên nền tảng mạng Internet

### Mạng Internet

Trong phần trên các bạn đã biết khái niệm mạng cục bộ (LAN). Bây giờ tưởng tượng tất cả các mạng LAN được đấu nối với nhau (qua đường truyền điện thoại, hoặc cáp quang, hoặc vệ tinh, ...) thì khả năng là tất cả các máy tính trên thế giới có thể liên lạc được với nhau. Mạng khổng lồ này gọi là mạng Internet.

Trên mạng Internet người ta vẫn dùng địa chỉ IP để liên lạc với nhau. Tuy nhiên địa chỉ IP thì rất khó nhớ. Người ta phát minh ra cách ánh xạ địa chỉ IP thành một cái tên để dễ nhớ hơn gọi là tên miền (domain name).

Để biết địa chỉ IP của tên miền thì bạn dùng lệnh ping <tên miền>. Ví dụ: ping google.com



```
C:\WINDOWS\system32\cmd.exe

C:\Users\Le Ngoc Thach>ping google.com

Pinging google.com [172.217.31.238] with 32 bytes of data:
Reply from 172.217.31.238: bytes=32 time=28ms TTL=54
Reply from 172.217.31.238: bytes=32 time=28ms TTL=54
Reply from 172.217.31.238: bytes=32 time=29ms TTL=54
Reply from 172.217.31.238: bytes=32 time=29ms TTL=54

Ping statistics for 172.217.31.238:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 28ms, Maximum = 29ms, Average = 28ms
```

### Phần mềm trên mạng Internet

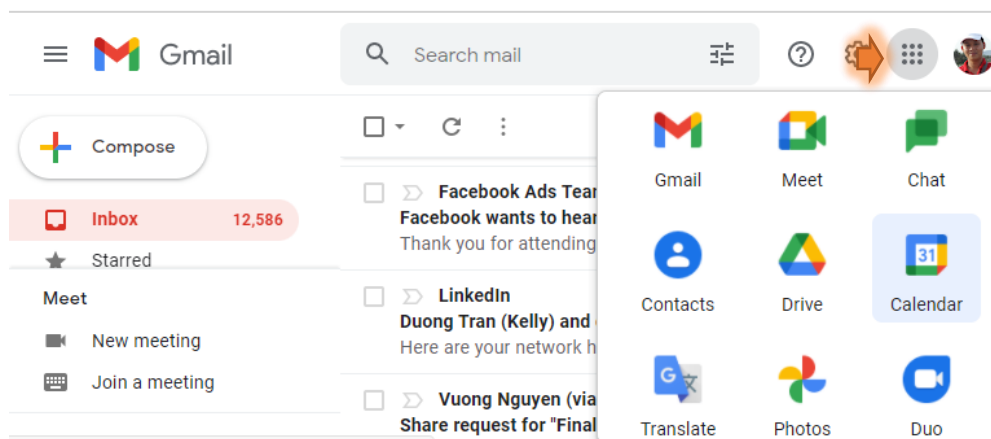
Nhờ phát minh ra Internet nên giới lập trình có cơ hội viết ra các phần mềm và cài nó lên một máy server. Sau đó mua dịch vụ ánh xạ máy chủ thành một cái tên cho dễ nhớ. Người dùng có thể truy cập phần mềm này thông qua tên miền.

Ví dụ: bạn có thể dùng phần mềm quản lý thư điện tử của Google qua tên miền bằng cách dùng trình duyệt gõ địa chỉ <https://mail.google.com>. Sau đó

bấm vào biểu tượng



chỗ hình mũi tên để truy cập các ứng dụng khác của Google.



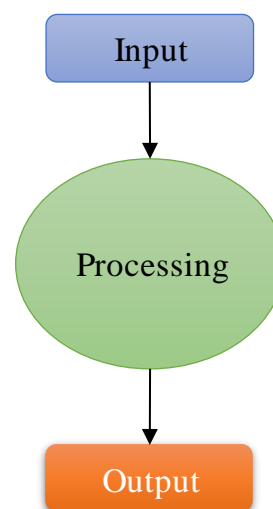
## **Ôn tập #2 – Cấu trúc của phần mềm**

*Bài này giúp bạn ghi nhớ 3 thành phần cơ bản trong hệ thống phần mềm. Đây là kim chỉ nam giúp bạn làm các bài tập cũng như xây dựng các công cụ trong tài liệu, hoặc khóa học đi kèm một cách đầy đủ, tự tin hơn.*

## Công thức I + P + O

Một phần mềm (PM) hoặc hệ thống thông tin (HTTT) nói chung gồm 3 phần:

- Thu nhận thông tin (**I**ntput)
- Xử lý thông tin thu nhận được (**P**rocess)
- Xuất kết quả (**O**utput)



### Thu nhận thông tin

Thông thường thì người dùng sẽ **cung cấp thông tin** (nhập liệu) hoặc **ra lệnh cho phần mềm** thông qua bàn phím và chuột, cao cấp hơn qua micro (nói)

### Xử lý thông tin

Thông tin được người dùng cung cấp sẽ được biến đổi, tổng hợp theo một hoặc nhiều mục đích nào đó.

### Xuất kết quả.

Thông tin sau khi được xử lý có thể được trình bày cho người dùng biết, đồng thời có thể được lưu trữ lại để dùng về sau.

Cách thức trình bày phổ biến là:

- Hiển thị ra màn hình máy tính
- In ra giấy
- Lưu thành file trên máy tính để người dùng có thể tự xem bằng các phần mềm khác như: Word, Excel, PDF,...

### Ví dụ

Bạn có thể dùng phần mềm xử lý bảng tính (Spreadsheet processing) như Excel, Open Office Calculator để nhập liệu và lưu thành file chứa thông tin của bạn bè mình. Sau khi bạn mở máy tính lên, khởi động phần mềm lên, quá trình IPO diễn ra như sau:

- Bạn **ra lệnh** cho phần mềm mở một tài liệu mới để bắt đầu soạn thảo bằng cách bấm phím tổ hợp phím Ctrl + N.
- Phần mềm Excel nhận lệnh Ctrl + N (**I**ntput) rồi xử lý (**P**rocessing) bằng cách mở ra một cửa sổ để bạn gõ nội dung vào (**I**ntput).

- Trong quá trình bạn gõ nội dung (Input) bạn có thể ra lệnh để Excel thực hiện (Processing) như: định dạng chữ đậm, nghiêng, ... sao chép (copy & paste), cắt dán (cut & paste), ...
- Trong quá trình bạn soạn nội dung thì phần mềm tự động lưu tài liệu vào đĩa cứng với tên file tạm (Output) với các kí tự đặc biệt để đề phòng trường hợp có sự cố.
- Sau khi soạn xong nội dung, bạn ra lệnh cho phần mềm lưu lại công sức của mình thì nhấn tổ hợp phím Ctrl+S. Phần mềm sẽ xử lý (Processing) bằng cách hiển thị hộp thoại để yêu cầu bạn cung cấp thông tin đường dẫn và tên file muốn lưu. Tiếp theo phần mềm sẽ lưu nội dung tài liệu với tên mà bạn đã cung cấp vào đường dẫn tương ứng (Output).

Như vậy bạn thấy rằng các hoạt động IPO diễn ra liên tục và đan xen với nhau tùy theo ý đồ thiết kế, sắp xếp chức năng của người lập trình.