

Chương 1

Nhập môn làm phần mềm

Lời nhắn

Tài liệu này được trích từ eBook “Để trở thành lập trình viên Java chuyên nghiệp”. Bạn được tặng tài liệu này để chuẩn bị tham gia các dự án nhằm giúp bạn có thêm kinh nghiệm để nâng cao chất lượng công việc của mình.

Để ủng hộ tác giả, bạn thực hiện 2 việc sau (không bắt buộc)

1) Vào link dưới để cài **App MinePI**:

<https://minepi.com/thachln>

Sử dụng invitation code: **thachln**

Mục đích: đây là Mobile App giúp bạn “điểm danh” để tích lũy PI Coin (1 đơn vị để giao dịch trong tương lai).

2) Thử sử dụng trình duyệt Netbox thay vì dùng Chrome/IE/Cốc Cốc,...

Link: <https://netbox.global/r/Y5mXp>

Mục đích: Dùng Netbox hằng ngày để tích lũy NBX Coin.

PI Coin và NBX Coin ở trên có thể dùng để mua hàng trong tương lai. Trước tiên có thể mua eBook tại trang <https://ThachLN.github.io>.

Lê Ngọc Thạch

Bài 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

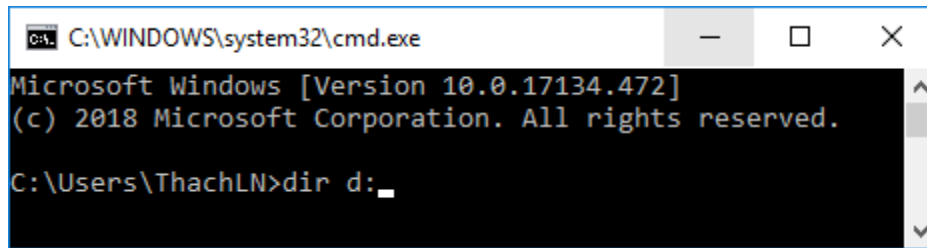
Từ những bản Hệ điều hành đầu tiên ra đời cho đến ngày nay thì việc ra lệnh cho máy vi tính thực hiện một công việc nào đó thông qua **cửa sổ gõ lệnh**¹ vẫn phổ biến.

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:
```

¹ Thuật ngữ tiếng Anh tương ứng có khác nhau trên các HĐH. Trong Windows gọi là "Prompt". Trong Mac và Linux gọi là Terminal.



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

C:\Users\ThachLN>dir d:
'mvnc' is not recognized as an internal or external command,
operable program or batch file.
```

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.

Internal command

✓ Là lệnh có sẵn trong hệ điều hành của máy tính.

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.

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:

External command

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

- .com
- .exe
- .bat
- .cmd

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:

```
path
path /?
echo %PATH%
set
set /?
cd
dir
hostname
hi
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:

```
echo $PATH
set
set -help
help set
pwd
ls
hostname
hi
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.

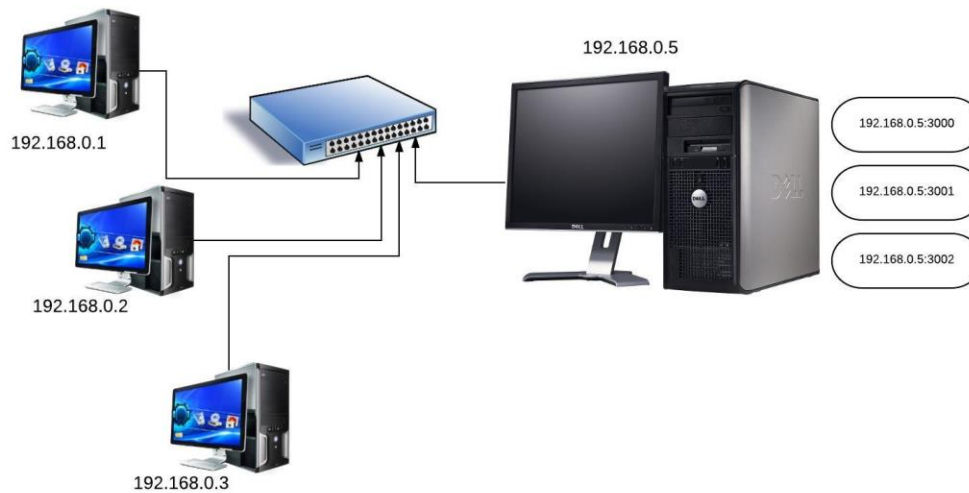
Hạn chế

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
```

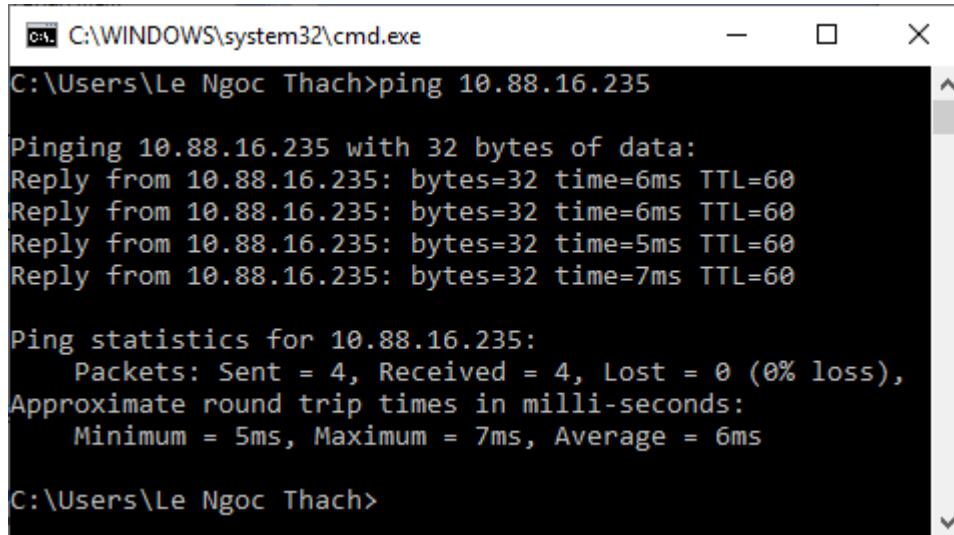
Trên Linux/Mac, gõ lệnh:

```
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.

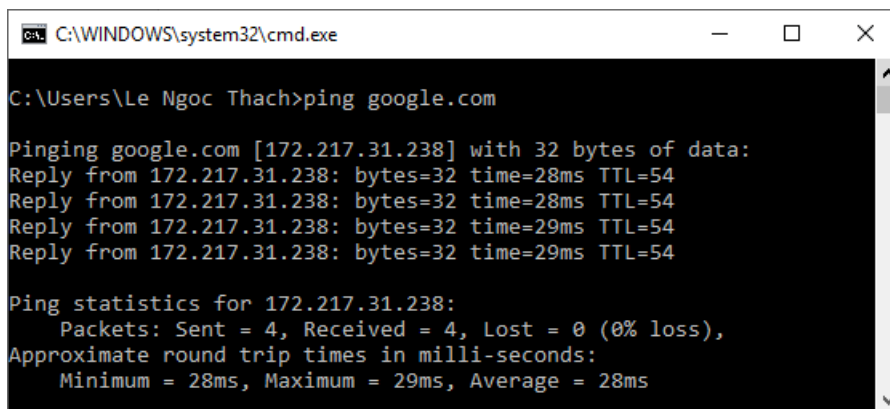
Phần mềm trên nền tảng mạng Internet

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 việc 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

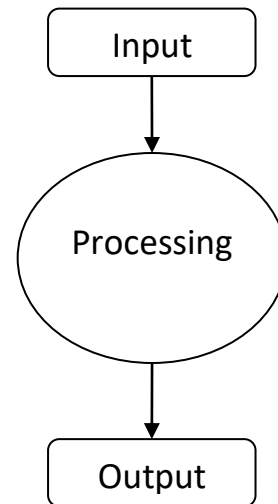
Nhờ phát minh ra Internet nên giới lập trình có cơ hội viết ra các phần mềm và để nó lên một máy server. Sau đó mua dịch vụ ánh xạ máy chủ thành một cái tên cho dễ nhớ. Người dùng có thể truy cập phần mềm này thông qua tên miền. Ví dụ bạn có thể dùng phần mềm quản lý thư điện tử của Google qua tên miền bằng cách dùng trình duyệt gõ địa chỉ <https://mail.google.com>.

Bài 2 – Cấu trúc của phần mềm

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**nput)
- 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 1:

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 (Input) rồi xử lý (Processing) bằng cách mở ra một cửa sổ để bạn gõ nội dung vào (Input).
- 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.

Bài tiếp theo sẽ giúp bạn tạo ra phần mềm để nắm vững hơn công thức I + P + O.

Bài 3 – Phần mềm đầu tiên – Cài đặt phép toán cộng

Phần này sẽ giúp các bạn hình dung rõ hơn toàn bộ quá trình làm phần mềm theo công thức IPO.

Bước 1: Xác định và viết yêu cầu

Công việc phần này là bạn gặp Khách hàng để hiểu rõ Khách hàng muốn gì. Hoặc đơn giản hơn bạn chính là Khách hàng của dự án này. Để có thể hình dung ra được bối cảnh thì tôi tưởng tượng ra tình huống như sau để các bạn thực hành.

Tình huống

Bạn gặp một nhà đầu tư D yêu cầu bạn phát triển dự án “Cộng hai số lớn” với yêu cầu cụ thể như sau:

- Chương trình cho phép các học sinh tiểu học thực hiện được phép tính tổng hai số lớn (nói chung là cực kỳ lớn, về mặt lý thuyết là có thể dài vô tận).
- Ngoài chức năng nói trên thì các sản phẩm bàn giao phải có đầy đủ tài liệu mô tả Yêu cầu (Requirement), mô tả Thiết kế (Design), mô tả Kiểm thử mã nguồn² (Unit Testing), Kết quả Đánh giá mã nguồn (Checklist, Code Report).
- Mã nguồn phải trong sáng, dễ hiểu, dễ bảo trì. Tuân thủ theo chuẩn lập trình (Coding Convention) của Oracle hoặc Google.

Với yêu cầu cho dự án như trên của nhà đầu tư D thì bạn bắt tay vào công việc như thế nào?

Xác định High Level Requirement

Sau khi đã nghe nhà đầu tư D (xem như là Khách hàng của bạn) nói rõ yêu cầu như vậy thì việc đầu tiên là bạn mô tả lại tất cả nội dung đấy vào một văn bản gọi là High Level Requirement (Tài liệu yêu cầu mức cao hoặc Tài liệu yêu cầu tổng thể). Trong

² Dịch sát nghĩa là Kiểm thử đơn vị (Unit Testing) nhưng tôi thích dùng Kiểm thử mã nguồn nghe xuôi tai hơn cho các bạn học nhập môn lập trình.

Bạn có thể dùng phần mềm Word để trình bày. Tuy nhiên để tiện cho bạn theo dõi tài liệu này thì tôi sẽ dùng Excel để trình bày tài liệu. Cái tiện lợi của Excel là có nhiều sheets để trình bày nhiều loại tài liệu trong cùng một file. Bạn có thể lấy file tại link “https://github.com/mksgroup/myworkspace/blob/master/ebooks/ppj/chapter-1/A2N_High-level-requirement_vi.xlsx”.

Sheet thứ nhất “Record of change” dùng để ghi lại lịch sử thay đổi tài liệu. Sheet thứ hai “Requirement” ghi lại các yêu cầu của dự án.

Template_High level requirement.xlsx - Excel

File Home Insert Page Layout Formulas Data Review View Add-ins Team Tell me... Thach Le Share

H17

	A	B	C	D	E
1					
2	*A - Add, M - Modify, D -Delete				
3					
4	Date	Changed item	A/M/D*	Description	Version
5	21-May-2017		A	The first version	0.
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					

Page 1

Record of change Requirement

Ready

<Code>		<Project Name>		Process	
Description		Create Date	Creator	Requirement	
		Update Date	Updater		
1. Feature list					
#	Feature groups	Code	Use case name/Screen	Description	
I	<Nhóm chức năng 1>				
1	Feature 1	A01			
		A02			
2. Other requests					
<Describe here>					
3. References					
1)					
2)					

Sau khi viết đầy đủ thì được tài liệu High Level Requirement cho dự án Cộng hai số lớn như sau:

Requirement					
A2N	Cộng hai số lớn			Process	
Description	Create Date	Update Date	Creator	Requirement	
Phần mềm cộng hai số lớn	20-May-18		ThachLN		
1. Feature list					
Đây là phần mô tả chức năng cho phần mềm Cộng hai số lớn. Chương trình này giúp cho học sinh tiểu học hiểu và thực hiện từng bước của phép toán cộng.					
#	Feature groups	Code	Use case name/Screen	Description	
I	<Nhóm chức năng 1>				
1	Feature 1	A01	Cộng hai số lớn	Chỉ hỗ trợ số nguyên dương.	
		A02	Ghi lại các bước để thực hiện phép toán	Nhằm giúp cho người dùng hình dung được các bước cần thực hiện.	
2. Other requests					
Sản phẩm bàn giao có đầy đủ:					
① Tài liệu mô tả Yêu cầu (Requirement)					
② Tài liệu mô tả Thiết kế (Design)					
③ Tài liệu mô tả Kiểm thử đơn vị (Unit Testing)					
④ Tài liệu báo cáo đánh giá mã nguồn (Review code report)					
3. References					
1) Tham khảo tài liệu Coding convention của Oracle					
2) Tham khảo tài liệu Coding convention của Google.					

Page 1

Ghi nhớ

Việc đầu tiên là phải mô tả lại yêu cầu tổng thể của Khách hàng vào một tài liệu gọi là High Level Requirement.

Viết Software Requirement Specification

Bộ tài liệu thứ hai bạn³ cần viết là Software Requirement Specification, gọi tắt là SRS. Vì phần mềm này khá đơn giản và để bạn tập trung vào việc làm quen với lập trình thì tạm thời bỏ qua phần này. Chi tiết việc viết SRS sẽ được đề cập trong các chương kế tiếp.

Bước 2: Làm thiết kế

Nghe đến từ Thiết kế cho phần mềm thì bạn có thể khóps và lo lắng không biết viết như thế nào. Phần này tôi sẽ giúp các bạn làm quen dần với cách trình bày Design. Việc đầu tiên là bạn cần luyện cách trình bày thuật toán cho dự án “Cộng hai số lớn” này.

Trước khi bạn học các kỹ năng, các công cụ nghe có vẻ cao siêu trong ngành Công nghệ phần mềm thì tôi chắc là các bạn có thể diễn đạt được ý tưởng, cách làm của mình bằng tiếng mẹ đẻ.

Bạn có thể vừa đọc phần này vừa thực hành bằng cách lấy giấy bút ra trình bày theo hướng dẫn:

Cho trước thông tin đầu vào gồm có hai số 123456 và 7890. Bạn hãy mô tả lại từng bước bạn cộng như thế nào. Nếu được thì hãy khái quát hoá lên thành phương pháp (gọi là Thuật toán). Ví dụ:

$$\begin{array}{r} 123456 \\ + \\ 7890 \\ \hline \end{array}$$

Tôi tin chắc là bạn có thể tự viết được cách làm như sau:

Thuật toán phiên bản 1

Bước 1: Lấy ra số cuối cùng của chuỗi 1 (số 6)

Bước 2: Lấy ra số cuối cùng của chuỗi 2 (số 0)

Bước 3: Thực hiện Cộng 2 số vừa lấy ($6 + 0 = 6$)

³ Tôi dùng từ “Bộ” để bạn hình dung trong trường hợp phần mềm phức tạp thì có nhiều loại tài liệu, nhiều dạng (format) tài liệu khác nhau để làm nên Software Requirement Specification.

Ghi nhớ
Để trình bày thuật toán thì bước đầu tiên hãy viết bằng lời như những gì bạn nói với người đối diện.

Bước 4: Ghi nhận kết quả là 6 (vì nhỏ hơn 10 nên không phải nhớ. Nếu kết quả Bước 3 lớn hơn 10 thì ghi nhận kết quả là phần đơn vị.

Lặp lại Bước 1 đến Bước 4 cho số bên trái tiếp theo. Cụ thể trong lần lặp thứ 2:

Bước 1: Lấy ra số 5

Bước 2: Lấy ra số 9

Bước 3: Thực hiện $5 + 9$ bằng 14

Bước 4: Vì 14 lớn hơn 10 nên lấy số 4 ghi vào kết quả (ghi vào bên trái của kết quả đang có là 6). Kết quả mới là 46. Phần chục trong số 14 được ghi nhớ (nhớ 1).

Lần lặp thứ 3:

Bước 1: Lấy ra số 4

Bước 2: Lấy ra số 8

Bước 3: Thực hiện phép cộng

Thực hiện $4 + 8$ và cộng thêm nhớ 1 của lần lặp thứ 2. Kết quả là $4 + 8 + 1 = 13$.

Bước 4: Cập nhật kết quả

Lấy số 3 ghép vào bên trái của 46 ta được 346. Và tiếp tục nhớ 1.

Lần lặp thứ 4, 5, 6 sẽ tương tự như vậy. Tôi chắc là bạn tự làm được. Ở đây tôi chỉ ghi chú rõ thêm trường hợp chuỗi số hai khi đã lấy hết số (ở bước lặp thứ 5) thì số được lấy ra xem như là zero (0).

Viết ra giấy các bước này không phải là khó nhưng cũng không phải là dễ vì các bạn có thể nghĩ là nó quá quen thuộc. Nhưng nếu bạn không ghi ra hoặc ít ra là nghĩ ra một cách rõ ràng trong đầu thì khả năng bạn làm sai trong các bước tiếp theo là rất lớn.

Sau khi đã có văn bản ở trên thì bạn đã hình dung được cách làm như thế nào? Tuy nhiên, bạn có thể cải tiến văn bản ở trên cho gọn hơn thành một văn bản có thể xem làm thuật toán.

Thuật toán phiên bản 2

Trong phiên bản 2 này thì bạn cần hiểu một số khái niệm trong máy tính:

- Các ngôn ngữ lập trình thường biểu diễn chuỗi số gồm nhiều kí tự liên tục và được đánh số thứ tự từ 0. Ví dụ chuỗi “123456” được thể hiện bằng các ô nhớ có chỉ số như bên dưới

	1	2	3	4	5	6
Chỉ số	0	1	2	3	4	5

- Biến: là một từ hoặc cụm từ nối với nhau bởi dấu gạch chân để chứa một giá trị nào đó. Ví dụ: $a = 0$ tức là biến a được gán giá trị zero (0).
- Cách viết $a = b$ gọi là phép gán giá trị b cho a .
- Sử dụng ngôn ngữ tự nhiên (tiếng Anh) để mô tả các hành động / hoạt động (gọi là hàm hoặc function). Ví dụ:
`length(s1)` là hành động tính độ dài của biến $s1$.

Cho thông tin đầu vào $s1$ và $s2$ là hai số dạng chuỗi (String). Thuật toán cộng $s1$ và $s2$ như sau:

Bước 1: Xác định độ dài của $s1$, $s2$ và độ dài lớn nhất của $s1$, $s2$.

$len1 = \text{length}(s1)$

$len2 = \text{length}(s2)$

$maxlen = \max(len1, len2)$

Bước 2: Duyệt từng kí tự

Cho chỉ số i lặp n lần (từ 0 đến $maxlen - 1$)⁴. Mỗi lần i tăng lên

1. Mỗi bước lặp thực hiện công việc tiếp theo từ bước 3.

⁴ Tùy theo Ngôn ngữ lập trình khác nhau thì nên lặp từ 0 hoặc từ 1 cho hợp lý. Đa số các Ngôn ngữ lập trình hiện đại thì nên lặp từ 0 do cách đánh chỉ số của các kí tự trong chuỗi hoặc trong mảng là từ 0 (zero).

Bước 3: Xác định chỉ số $i1$, $i2$ đánh dấu kí tự cần lấy của chuỗi $s1$ và $s2$.

Cách tính như sau:

$$i1 = \text{len1} - i - 1$$

$$i2 = \text{len2} - i - 1$$

Việc xác định công thức trên có thể có được từ việc viết ra giấy với ví dụ ở trên: $s1 = "123456"$, $s2 = "7890"$.

Độ dài lớn nhất của 2 chuỗi là: $\text{maxlen} = 6$

khi cho $i = 0$ thì $i1$ phải bằng 5 (trong lập trình kí tự tại vị trí 5 trong chuỗi $s1$ là kí tự '6').

khi cho $i = 1$ thì $i1$ phải bằng 4. Tức là khi i tăng lên thì $i1$ giảm đi 1.

khi cho $i = 5$ (tức là đến cuối chuỗi) thì $i1$ phải bằng 0.

Nên công thức $i1$ phụ thuộc vào i đâu đó như sau

$$i1 = \text{len1} - i$$

(Bên phải dấu bằng là biểu thức có trừ i để thể hiện tính nghịch đảo: khi i tăng thì $i1$ giảm. Và khi i xuất phát là không thì $i1$ phải xuất phát từ độ dài của chuỗi 1 bên phải lấy len1 trừ bớt cho i)

Bước 4: Lấy ra kí tự của tại vị trí $i1$, $i2$ tương ứng của chuỗi $s1$, $s2$.

$$c1 = \text{st1.charAt}(i1)$$

$$c2 = \text{st2.charAt}(i2)$$

Chú ý trường hợp $i1$ âm ($i1 < 0$) thì gán $c1 = 0$ luôn chứ không thực hiện `charAt` vì sẽ gây lỗi chỉ số chuỗi không lệ.

Và tương tự nếu $i2 < 0$ thì $c1 = 0$.

Bước 5: Xác định giá trị tương ứng của kí tự (character) $c1, c2$.

$d1 = c1 - '0'$

$d2 = c2 - '0'$

Trong máy tính thì các kí tự (char) thường được biểu diễn bằng một số nguyên, tương ứng mỗi số nguyên thì máy tính sẽ vẽ lên màn hình cái chữ tương ứng để cho ta đọc⁵. Để chuyển một kí tự thành giá trị số (digit) thì bạn chỉ cần lấy giá trị mã ASCII của nó trừ cho mã ASCII của kí tự zero '0'.

Bước 6: Thực hiện cộng hai kí số $d1$ và $d2$

$t = d1 + d2$

Bước 7: Thực hiện lấy số hàng đơn vị của t và ghi nhận cờ nhớ

$resultTemp = t \% 10$ (chia lấy phần dư)

$mem = t / 10$ (chia lấy phần nguyên)

Bước 8: Ghép kết quả $resultTemp$ vào kết quả cuối cùng

$finalResult = resultTemp + finalResult$

(chú ý ghép vào bên trái và giá trị ban đầu của $finalResult$ là cuối rỗng)

Quay lại bước 2.

Bước 9: Khi kết thúc vòng lặp ở trên nếu biến nhớ mem có giá trị lớn hơn 0 thì ghép tiếp vào kết quả cuối cùng.

Nếu $mem > 0$ thì: $finalResult = mem + resultTemp$.

⁵ Xem thêm bảng mã ASCII trong máy tính (<https://vi.wikipedia.org/wiki/ASCII>). Chú kí tự ở đây tôi viết bao bởi dấu nháy **đơn** để phân biệt với chuỗi được bao bởi dấu nháy **đôi**.

Đến đây thì bạn có thể hình dung rõ hơn thuật toán cộng hai số như các bạn học sinh lớp 3 thường làm.

Tuy nhiên Thuật toán phiên bản 2 ở trên vẫn còn thiếu sót một chút ở bước 6. Do tôi cố tình trình bày theo luồng suy nghĩ bình thường để bạn hiểu thuật toán nên trong bước 6 chưa thực hiện phép cộng thêm biến nhớ nếu nó có giá trị.

Chúng ta sẽ cải tiến một chút ở bước 6, thêm bước 0 và bôi đậm các phần bổ sung để bạn tiện theo dõi trong thuật toán phiên bản 3.

Thuật toán phiên bản 3

Cho thông tin đầu vào s1 và s2 là hai số dạng chuỗi (String). Thuật toán cộng s1 và s2 như sau:

Bước 0: Chuẩn bị các biến và kiểu dữ liệu, giá trị ban đầu cho chương trình

String s1: chuỗi lưu số thứ nhất.

String s2: chuỗi lưu số thứ hai.

String finalResult = "" : Chuỗi lưu kết quả cuối cùng.

int len1: số nguyên lưu độ dài của s2.

int len2: số nguyên lưu độ dài của s1.

int maxlen: số nguyên lưu độ dài lớn nhất trong len1 và len2.

char c1: kí tự tạm thời trong quá trình duyệt chuỗi s1

char c2: kí tự tạm thời trong quá trình duyệt chuỗi s2

int d1: giá trị số nguyên của c1

int d2: giá trị số nguyên của c2.

int t: số nguyên lưu tổng tạm của từng kí số d1, d2.

Ghi nhớ
Cứ trình bày thuật toán một cách tự nhiên như là nói. Sau đó sẽ cải tiến dần để có nội dung cô đọng hơn.

**int mem = 0: giá trị của cờ nhớ trong quá trình cộng.
Khởi động bằng 0.**

Bước 1: Xác định độ dài của s1, s2 và độ dài lớn nhất của s1, s2.

len1 = length(s1)

len2 = length(s2)

maxlen = max(len1, len2)

Bước 2: Cho chỉ số i lặp n lần (từ 0 đến maxlen - 1)⁶. Mỗi lần i tăng lên 1.

Bước 3: Xác định chỉ số i1, i2 đánh dấu kí tự cần lấy của chuỗi s1 và s2.

Cách tính như sau:

$i1 = len1 - i - 1$

$i2 = len2 - i - 1$

Việc xác định công thức trên có thể có được từ việc viết ra giấy với ví dụ ở trên: s1 = "123456", s2 = "7890".

Độ dài lớn nhất của 2 chuỗi là: maxlen = 6

khi cho i = 0 thì i1 phải bằng 5 (trong lập trình kí tự tại vị trí 5 trong chuỗi s1 là kí tự '6').

khi cho i = 1 thì i1 phải bằng 4. Tức là khi i tăng lên thì i1 giảm đi 1.

khi cho i = 5 (tức là đến cuối chuỗi) thì i1 phải bằng 0.

Nên công thức i1 phụ thuộc vào i đâu đó như sau

$i1 = len1 - i$

Ghi nhớ

Cần phải viết ra giấy, vẽ hình minh họa ngay khi có thể để dễ kiểm chứng nội dung của thuật toán.

⁶ Tùy theo Ngôn ngữ lập trình khác nhau thì nên lặp từ 0 hoặc từ 1 cho hợp lý. Đa số các Ngôn ngữ lập trình hiện đại thì nên lặp từ 0 do cách đánh chỉ số của các kí tự trong chuỗi hoặc trong mảng là từ 0 (zero).

(Bên phải dấu bằng là biểu thức có trừ i để thể hiện tính nghịch đảo: khi i tăng thì $i1$ giảm. Và khi i xuất phát là không thì $i1$ phải xuất phát từ độ dài của chuỗi 1 bên phải lấy len1 trừ bớt cho i)

Bước 4: Lấy ra kí tự của tại vị trí $i1$, $i2$ tương ứng của chuỗi $s1$, $s2$.

$c1 = s1.charAt(i1)$

$c2 = s2.charAt(i2)$

Chú ý trường hợp $i1$ âm ($i1 < 0$) thì gán $c1 = 0$ luôn chứ không thực hiện charAt vì sẽ gây lỗi chỉ số chuỗi không lệ.

Và tương tự nếu $i2 < 0$ thì $c2 = 0$.

Bước 5: Xác định giá trị tương ứng của kí tự (character) $c1$, $c2$.

$d1 = c1 - '0'$

$d2 = c2 - '0'$

Trong máy tính thì các kí tự (char) thường được biểu diễn bằng một số nguyên, tương ứng mỗi số nguyên thì máy tính sẽ vẽ lên màn hình cái chữ tương ứng để cho ta đọc⁷. Để chuyển một kí tự thành giá trị số (digit) thì bạn chỉ cần lấy giá trị mã ASCII của nó trừ cho mã ASCII của kí tự zero '0'.

Bước 6: Thực hiện cộng hai kí số $d1$ và $d2$ và **biến nhớ**

$t = d1 + d2 + \text{mem}$

Bước 7: Thực hiện lấy số hàng đơn vị của t và ghi nhận cờ nhớ

$\text{resultTemp} = t \% 10$ (chia lấy phần dư)

$\text{mem} = t / 10$ (chia lấy phần nguyên)

⁷ Xem thêm bảng mã ASCII trong máy tính. Chữ kí tự ở đây tôi viết bao bởi dấu nháy **đơn** để phân biệt với chuỗi được bao bởi dấu nháy **đôi**.

Bước 8: Ghép kết quả resultTemp vào kết quả cuối cùng

`finalResult = resultTemp + finalResult`

(chú ý ghép vào bên trái và giá trị ban đầu của finalResult là cuối rỗng)

Quay lại bước 2.

Bước 9: Khi kết thúc vòng lặp ở trên nếu biến nhớ **mem** có giá trị lớn hơn 0 thì ghép tiếp vào kết quả cuối cùng.

Nếu `mem > 0` thì: `finalResult = mem + resultTemp`.

Đọc Javadoc - Kiểm tra khả năng cài đặt thuật toán

Nếu bạn đã học lập trình thì phần này sẽ không mấy khó khăn. Tuy nhiên đối với các bạn mới làm quen lập trình, mới làm quen với Java thì có thể hơi khó một chút.

Mục đích của phần này là giúp bạn luyện tập kỹ năng **đọc tài liệu lập trình Javadoc**⁸ để kiểm tra các thao tác (tức là các lệnh) trong thuật toán có thể thực hiện bằng ngôn ngữ lập trình Java như thế nào.

Khi hãng Oracle (trước đây là hãng Sun) cung cấp cho chúng ta thư viện và công cụ Java thì họ cũng cung cấp luôn bộ tài liệu để tra cứu các method của từng class trong bộ phát triển phần mềm gọi là JDK (Java Development Kit). Bạn có thể tra cứu bằng cách search Google từ khoá “javadoc 8”, mở link

“<https://docs.oracle.com/javase/8/docs/api/index.html?overview-summary.html>”⁹.

Một cách khác là hãy download bộ javadoc của jdk về máy để tra cứu cho nhanh. Hãy search Google với từ khoá “download javadoc jdk 8”, mở link để download

Ghi nhớ
Cần lưu bộ Javadoc JDK vào máy và luyện tập cách tranh cứu.

⁸ <https://docs.oracle.com/javase/8/docs/api/>

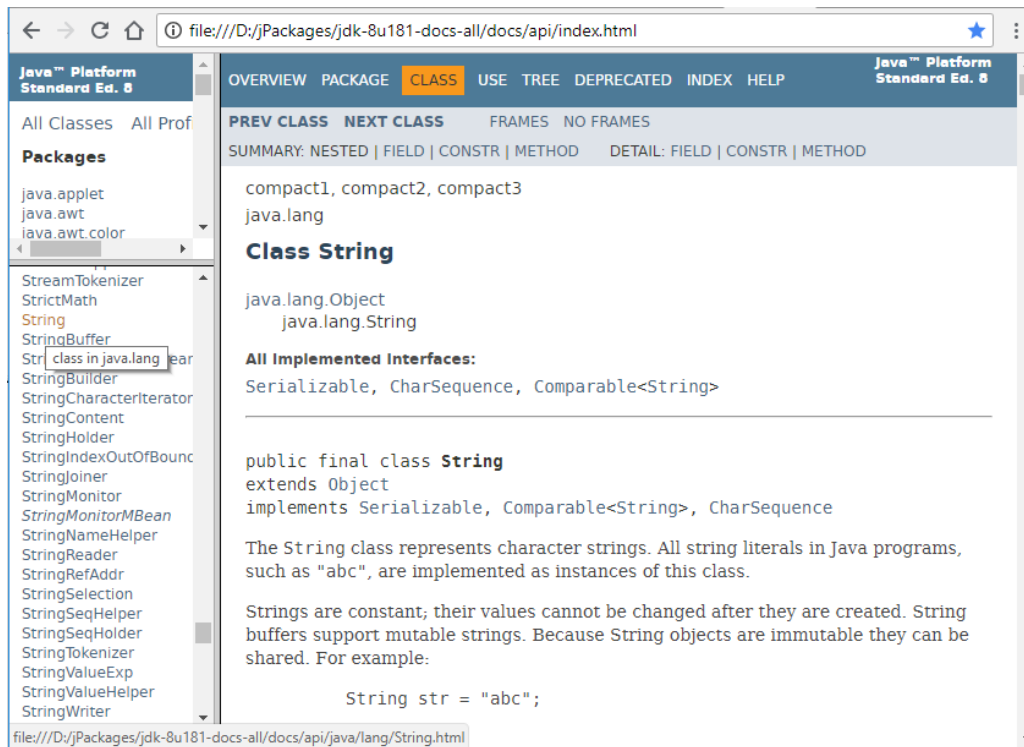
⁹ Vì lý do tương thích trong các dự án nên tại thời điểm viết ebook này tôi khuyến nghị các bạn dùng JDK 8 (dù đã có JDK9 và JDK10).

“<http://www.oracle.com/technetwork/java/javase/documentation/jdk8-doc-downloads-2133158.html>” như hình bên dưới:



Tải và giải nén file “jdk-8u181-docs-all.zip” vào thư mục D:\jPackages. Dùng trình duyệt mở file “D:\jPackages\jdk-8u181-docs-all\docs\api\index.html” và lưu vào Bookmarks bằng cách nhấn phím Ctrl + D .

Đối với thư viện Cộng hai số này thì bạn cần tra các API của class String.



API bạn cần tra thực hiện được các chức năng sau:

- Lấy ra kí tự tại một vị trí cho trước trong chuỗi (Returns the char value at the specified index).
- Lấy độ dài (số kí tự) của chuỗi (Returns the length of this string).

Thiết kế giao diện (User Interface)

Trong yêu cầu của nhà đầu tư thì họ không đề cập đến giao diện tương tác người dùng. Có nhiều giải pháp để lựa chọn cho phần này. Tuy nhiên để vấn đề trở nên đơn giản nhất có thể và giúp bạn làm quen với quy trình phát triển một dự án phát phần mềm thì tôi tưởng tượng thêm câu chuyện với nhà đầu tư như sau:

Bạn đã trao đổi với nhà đầu tư để có sản phẩm sớm và đưa ra thử nghiệm thì cần giới hạn chức năng cho phiên bản 1.0. Trong đó phần giao diện tương tác giữa người dùng thì nên dùng phím để nhập lệnh thôi. Sau khi phần mềm được cài đặt phù hợp

vào máy tính thì người dùng có thể gõ lệnh như bên dưới để chạy chương trình.

sum <n1> <n2>

Nếu gõ “sum” mà không có tham số hoặc có một tham số là /? thì xem được hướng dẫn sử dụng như sau:

sum						/?
Cú pháp sử dụng:	sum	<n1>	<n2>			
<n1>:	là	số	thứ	nhất		
<n2>:	là	số	hạng	thứ	hai	

Ví dụ: sum 1 2

Chú ý:

Như vậy trong giai đoạn hình dung thiết kế giao diện cho phần mềm, bạn đã gặp Khách hàng để thống nhất thêm về yêu cầu.

Bước 3: Lập trình và kiểm thử

Sau khi đã trình bày được thuật toán một cách rõ ràng, về cơ bản thì không có gì khuất mắt thì chúng ta bắt tay vào viết code.

Trong quá trình viết code thì có thể chia làm nhiều giai đoạn.

Giai đoạn một – cài đặt logic chính

Bạn nên tập thói quen ghi chú tài liệu cho method trước khi bắt tay viết code.

```
public class MyBigNumber {
    /**
     * Cộng hai số lớn dưới dạng chuỗi.
     * <br/>
     * Giả định 2 tham số truyền vào là đúng: chỉ chứa các kí số
     * từ '0' đến '9'.
     * @param s1 chuỗi số thứ nhất.
     * @param s2 chuỗi số thứ hai.
     * @return chuỗi có giá trị là tổng của hai số s1 và s2.
     */
    public String sum(String s1, String s2) {
        String finalResult = "";
    }
}
```

```

        return finalResult;
    }
}

```

Trong đoạn code trên tôi mô tả Javadoc cho method `sum(s2, s2)` để định hình trong đầu ý nghĩa của method mà chúng ta sẽ code. Nội dung của method `sum` lúc này chỉ có 2 dòng.

Dòng thứ nhất khai báo biến kết quả sẽ trả lại. Vì chưa biết kết quả sẽ được tính toán như thế nào nên tôi gán bằng rỗng (""). Sau đó `return` luôn.

Ghi nhớ
Trước lệnh <code>return</code> nên có dòng trắng.

Ch ú ý	<p>Phong cách lập trình:</p> <ul style="list-style-type: none"> • Trước lệnh <code>return</code> nên có dòng trắng.
-----------	--

Tiếp theo, chúng ta code vòng lặp để quét từng kí tự của hai chuỗi từ phải qua trái. Chú ý là trước khi code bạn nên ghi chú ý định (suy nghĩ) ra trước để làm.

```

public String sum(String s1, String s2) {
    String finalResult = "";

    // Quét các kí tự của chuỗi s1 và s2 từ phải qua trái

    //// Xác định độ dài của s1, s2 và độ dài lớn nhất của
    //// 2 chuỗi
    int len1 = s1.length();
    int len2 = s2.length();
    int maxLen = (len1 > len2) ? len1 : len2;

    int index1; // chỉ số của kí tự đang xét của chuỗi 1
    int index2; // chỉ số của kí tự đang xét của chuỗi 2

    //// Lặp maxLen lần
    for (int i = 0; i < maxLen; i++) {
        index1 = len1 - i - 1;
        index2 = len2 - i - 1;
    }

    return finalResult;
}

```

Đoạn code ở trên dùng một vòng lặp thực hiện maxLen lần. Mỗi lần lặp thì chỉ số i sẽ chạy từ 0 cho đến maxLen (chưa đến maxLen, tức là đến maxLen - 1).

Vì chúng ta cần lấy kí tự của chuỗi s1 và s2 từ phải sang trái. Tức là xuất phát từ **vị trí cuối cùng** của chuỗi rồi lần lượt giảm đi 1. Như vậy chúng ta cần tính 2 chỉ số index1 và index2 bởi công thức:

```
index1 = len1 - i - 1;
index2 = len2 - i - 1;
```

Ch
ú ý

Phong cách lập trình:

- Trước vòng lặp nên có dòng trắng.

Ghi nhớ

Trước vòng lặp nên có dòng trắng.

Sau khi đã có chỉ số index1 và index2 thì chúng ta sẽ lấy ra kí tự và tính kí số tương ứng.

```
public String sum(String s1, String s2) {
    String finalResult = "";

    // Quét các kí tự của chuỗi s1 và s2 từ phải qua trái

    //// Xác định độ dài của s1, s2 và độ dài lớn nhất của
    //// 2 chuỗi
    int len1 = s1.length();
    int len2 = s2.length();
    int maxLen = (len1 > len2) ? len1 : len2;

    int index1; // chỉ số của kí tự đang xét của chuỗi 1
    int index2; // chỉ số của kí tự đang xét của chuỗi 2
    char c1;    // kí tự tại vị trí index1 của chuỗi s1
    char c2;    // kí tự tại vị trí index2 của chuỗi s2
    int d1;     // kí số của c1
    int d2;     // kí số của c2

    //// Lặp maxLen lần
    for (int i = 0; i < maxLen; i++) {
        index1 = len1 - i - 1;
        index2 = len2 - i - 1;

        c1 = s1.charAt(index1);
        c2 = s2.charAt(index2);

        d1 = c1 - '0';
```

```

        d2 = c2 - '0';
    }

    return finalResult;
}

```

Sáu dòng code bôi vàng được bổ sung để xác định hai kí số d1 và d2. Để xác định kí tự tại vị trí cho trước trong chuỗi thì dùng method `charAt(vị trí)`. Để chuyển một kí tự (char) thành giá trị nguyên tương ứng với nó thì chúng ta sử dụng ý nghĩa của bảng mã ASCII. Cụ thể là ta lấy kí tự trừ đi kí tự zero '0' thì sẽ ra giá trị nguyên của kí tự. Ví dụ: '1' - '0' sẽ được 1.

Sau khi đã có d1 và d2 rồi thì việc cộng từng kí số là không có gì khó khăn.

Chú ý	<p>Phong cách lập trình:</p> <ul style="list-style-type: none"> Trong quá trình code thì sẽ có nhu cầu khai báo thêm biến. Thì không nên khai báo biến trong vòng lặp. Ở đây tôi khai báo d1 và d2 phía trước vòng lặp for. Đoạn code tính toán d1 và d2 mới thêm vào có ý nghĩa logic với đoạn code đang có. Vì vậy nên sử dụng một dòng trắng để cách ra. Cái này sẽ giúp cho code dễ đọc (readable).
-------	---

Ghi nhớ
Không khai báo biến trong vòng lặp
Giữa các khối lệnh thực hiện các nghiệp vụ / logic khác nhau nên có dòng trắng.

Chúng ta sẽ bổ sung tiếp code để thực hiện cộng hai kí số.

Code version 1

```

public String sum(String s1, String s2) {
    String finalResult = "";

    // Quét các kí tự của chuỗi s1 và s2 từ phải qua trái

    /// Xác định độ dài của s1, s2 và độ dài lớn nhất của
    /// 2 chuỗi
    int len1 = s1.length();
    int len2 = s2.length();
    int maxLen = (len1 > len2) ? len1 : len2;

    int index1;    // chỉ số của kí tự đang xét của chuỗi 1
    int index2;    // chỉ số của kí tự đang xét của chuỗi 2

```

```

char c1;          // kí tự tại vị trí index1 của chuỗi s1
char c2;          // kí tự tại vị trí index2 của chuỗi s2
int d1;           // kí số của c1
int d2;           // kí số của c2
int t;            // tổng tạm của d1 và d2;
int mem = 0;      // nhớ nếu t lớn hơn hoặc bằng 10

//// Lặp maxLen lần
for (int i = 0; i < maxLen; i++) {
    index1 = len1 - i - 1;
    index2 = len2 - i - 1;

    c1 = s1.charAt(index1);
    c2 = s2.charAt(index2);

    d1 = c1 - '0';
    d2 = c2 - '0';

    t = d1 + d2;

    // Lấy hàng đơn vị của t ghép vào phía trước kết quả
    finalResult = (t % 10) + finalResult;
    mem = t / 10;
}

// Kết thúc vòng lặp, nếu biến nhớ mem có giá trị thì
// ghép thêm mem vào phía trước kết quả
if (mem > 0) {
    finalResult = mem + finalResult;
}

return finalResult;
}

```

Phần code màu vàng ở trên thực hiện lấy hàng đơn vị của tổng (tạm) của 2 kí số ghép vào kết quả finalResult.

Để xác định giá trị biết nhớ nếu kết quả t lớn hơn hoặc bằng 10 thì chúng ta lợi dụng phép chia lấy phần nguyên (div). Trong Java sử dụng phép toán $t / 10$ và gán cho biến mem có giá trị nguyên (int).

Khi kết thúc vòng lặp thì có khả năng biến nhớ mem có giá trị. Vì vậy chúng ta phải kiểm tra nếu $mem > 0$ thì ghép tiếp vào kết quả. Chú ý là ghép vào bên trái chuỗi kết quả.

Như vậy đến thời điểm này thì chúng ta đã thực hiện xong ý tưởng chính của thuật toán. Tức là logic chính đã được cài đặt.

Nếu tinh ý thì bạn sẽ thấy đoạn code ở trên có thể chạy được với trường hợp hai chuỗi `s1` và `s2` truyền vào có độ dài bằng nhau. Nếu `s1` và `s2` có độ dài khác nhau sẽ gây ra lỗi khi chạy đến lệnh `charAt(index)` vì `index` có thể nằm ngoài phạm vi chỉ số các kí tự của chuỗi.

“Chạy thử”

Trước khi hoàn thiện các logic phụ thì tại sao chúng ta không “chạy thử” đoạn code ở trên nhỉ? Để xem thành quả của chúng ta ra sao. Chữ “chạy thử” ở đây tôi để trong dấu ngoặc kép có nghĩa chính xác là kiểm thử (Testing). Chính xác hơn là Unit Testing vì phần code này chỉ là một method, method này được xem là một đơn vị (Unit) trong phần mềm. Vì vậy kiểm thử một method gọi là Unit Testing¹⁰.

Thông thường thói quen của các bạn mới học lập trình là viết method main để chạy thử theo đúng nghĩa đen là thử. Tuy nhiên để rèn luyện kỹ năng lập trình chuyên nghiệp thì không nên chạy thử mà nên thực hiện Unit Testing để kiểm tra các tình huống theo ý đồ của chúng ta. Để làm quen với kỹ thuật Unit Testing khi lập trình Java với Eclipse thì phần tiếp theo tôi sẽ hướng dẫn các bạn test thử đoạn code cộng hai số dạng chuỗi ở trên.

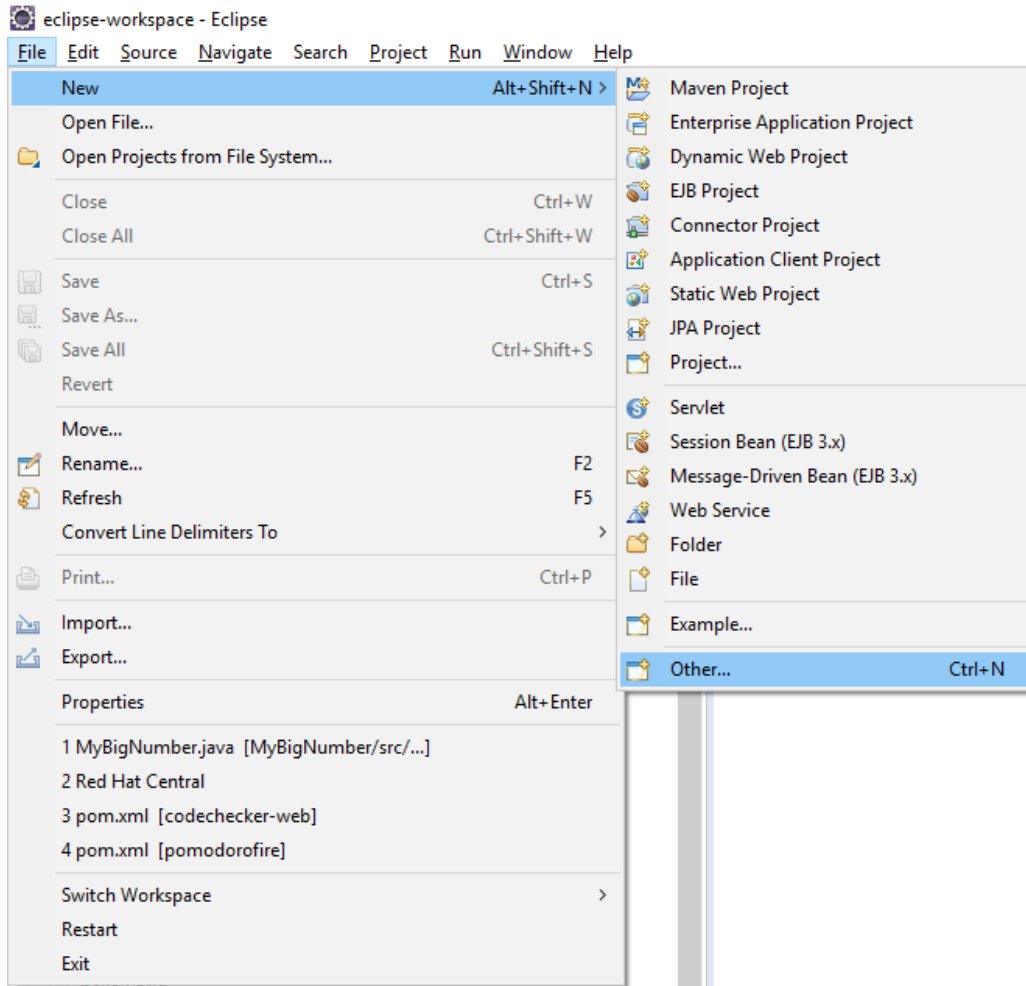
Tạo Project

Giả định bạn đã có bộ Eclipse đã cài gói Spring Tool Suite (hoặc sử dụng luôn bộ Spring Tool Suite IDE).

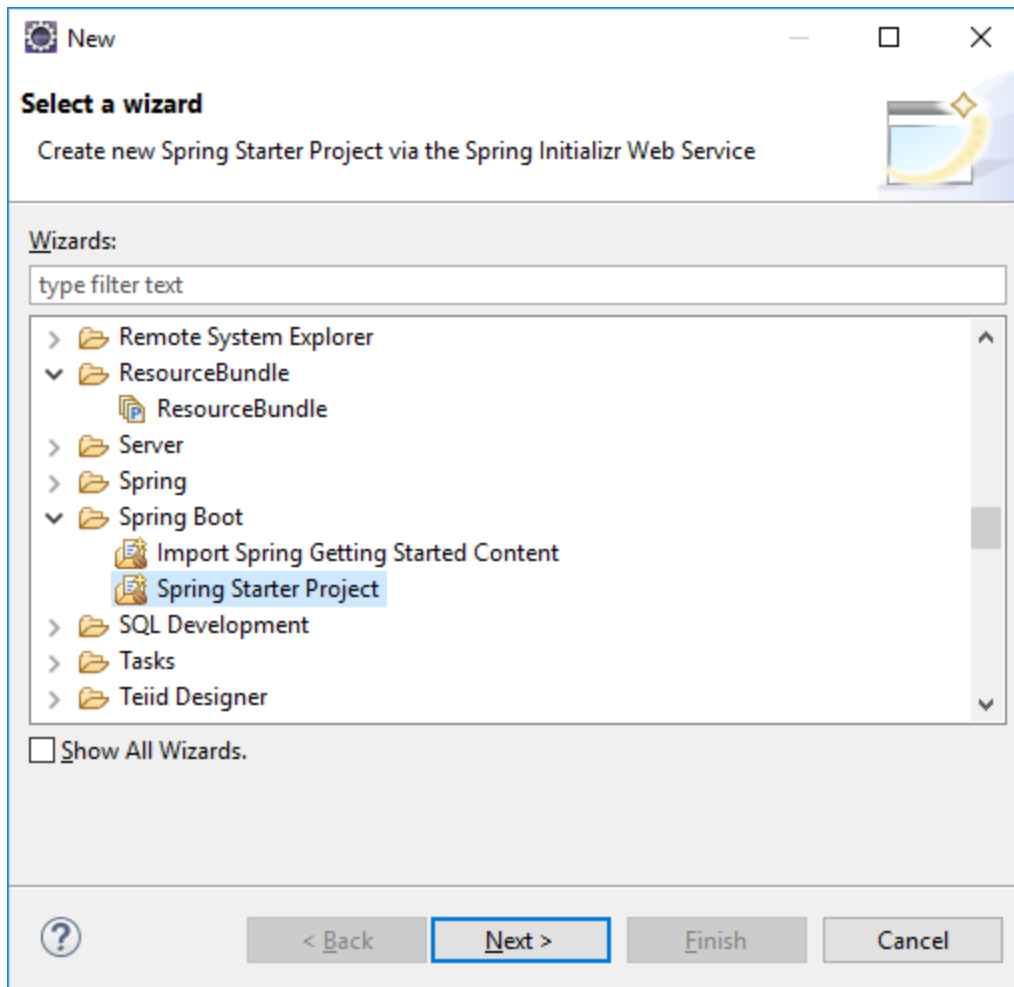
Khởi tạo project bằng Springboot.

Trong Eclipse, vào menu File > New > Other...

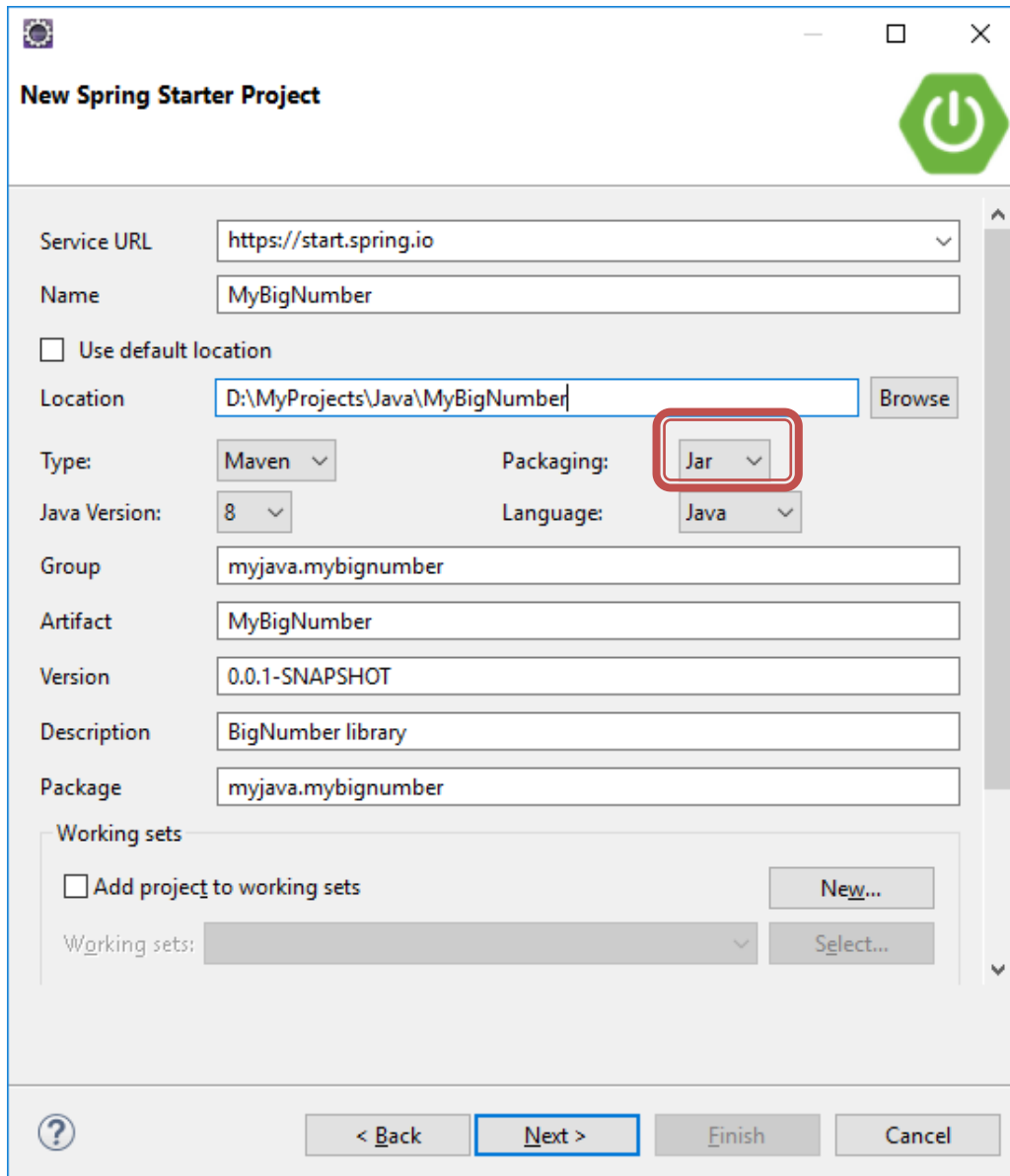
¹⁰ Nhưng nói ngược lại Unit Testing là kiểm thử method thì chưa đầy đủ nhé! Nội dung Unit Testing sẽ được bàn trong các chương kế tiếp



Hoặc nhấn phím tắt Ctrl + N để hiển thị hộp thoại tạo Project.



Chọn mục Spring Boot > Spring Starter project. Sau đó nhấn nút “Next”.



New Spring Starter Project

Service URL:

Name:

☐ Use default location

Location:

Type: Packaging: (highlighted with a red box)

Java Version: Language:

Group:

Artifact:

Version:

Description:

Package:

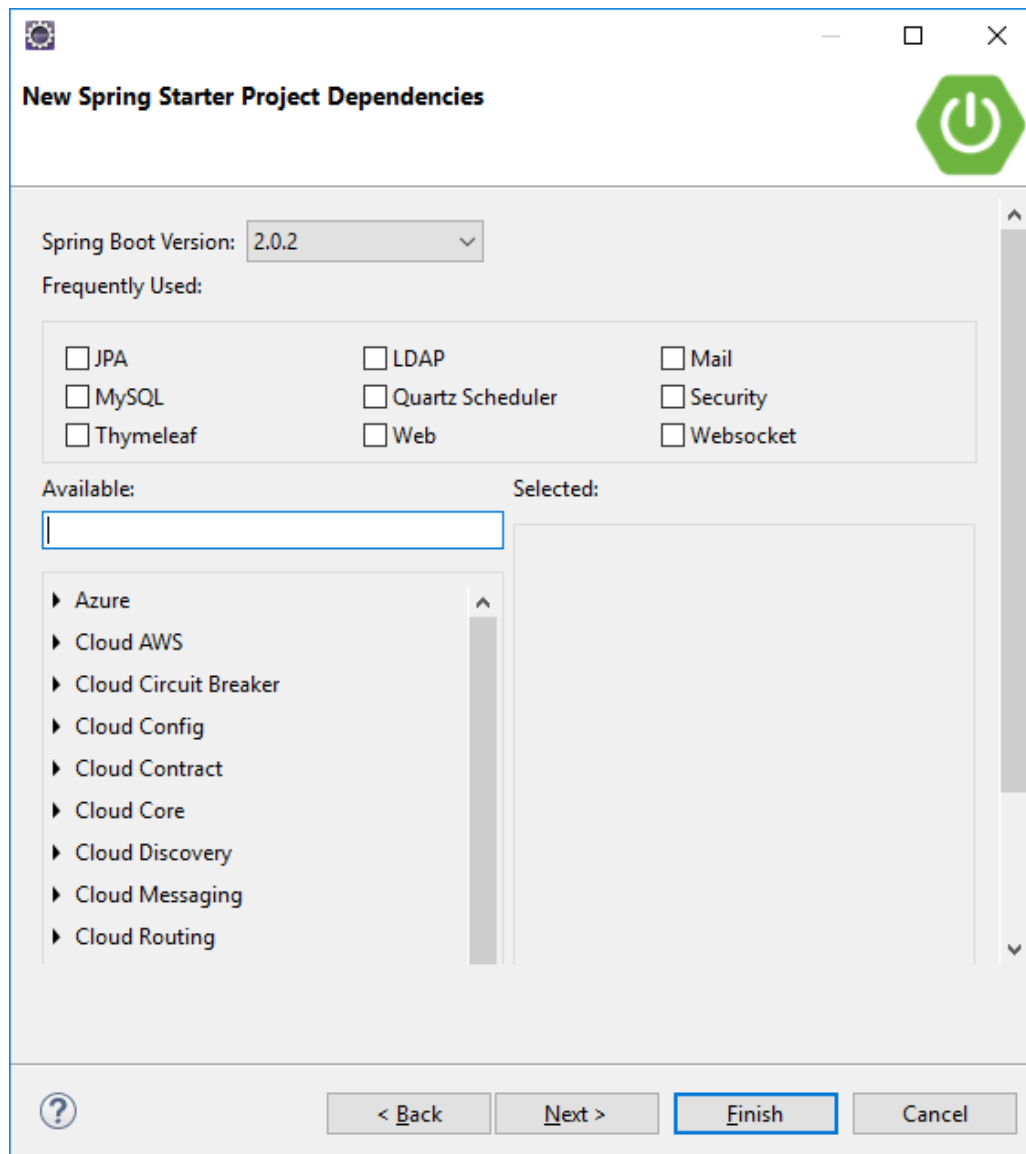
Working sets

☐ Add project to working sets

Working sets:

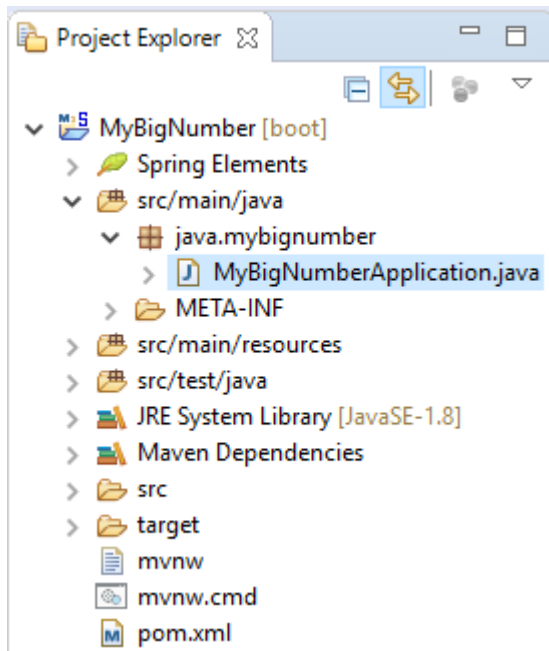
(highlighted with a blue box)

Máy tính của bạn cần phải có kết nối Internet để Eclipse kết nối tới địa chỉ <https://start.spring.io> để khởi tạo dự án. Bạn điền các thông số như trên hình. Sau đó nhấn nút “Next”.



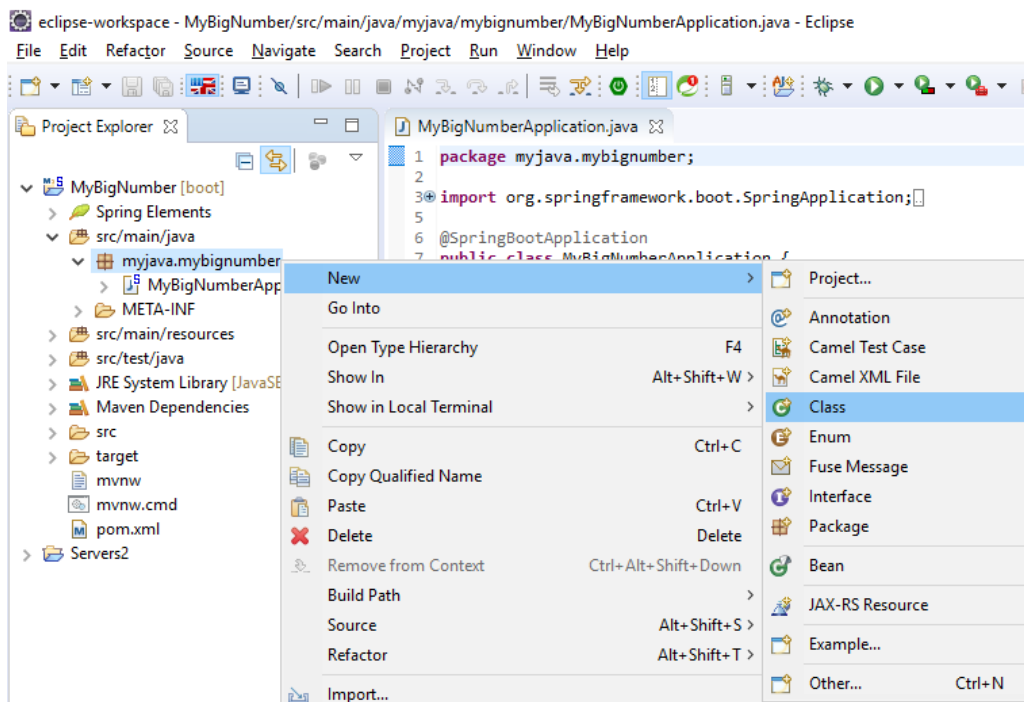
Phiên bản Spring Boot Version tại thời điểm viết tài liệu này là 2.0.2. Bạn nhấn tiếp nút “Next” và “Finish”.

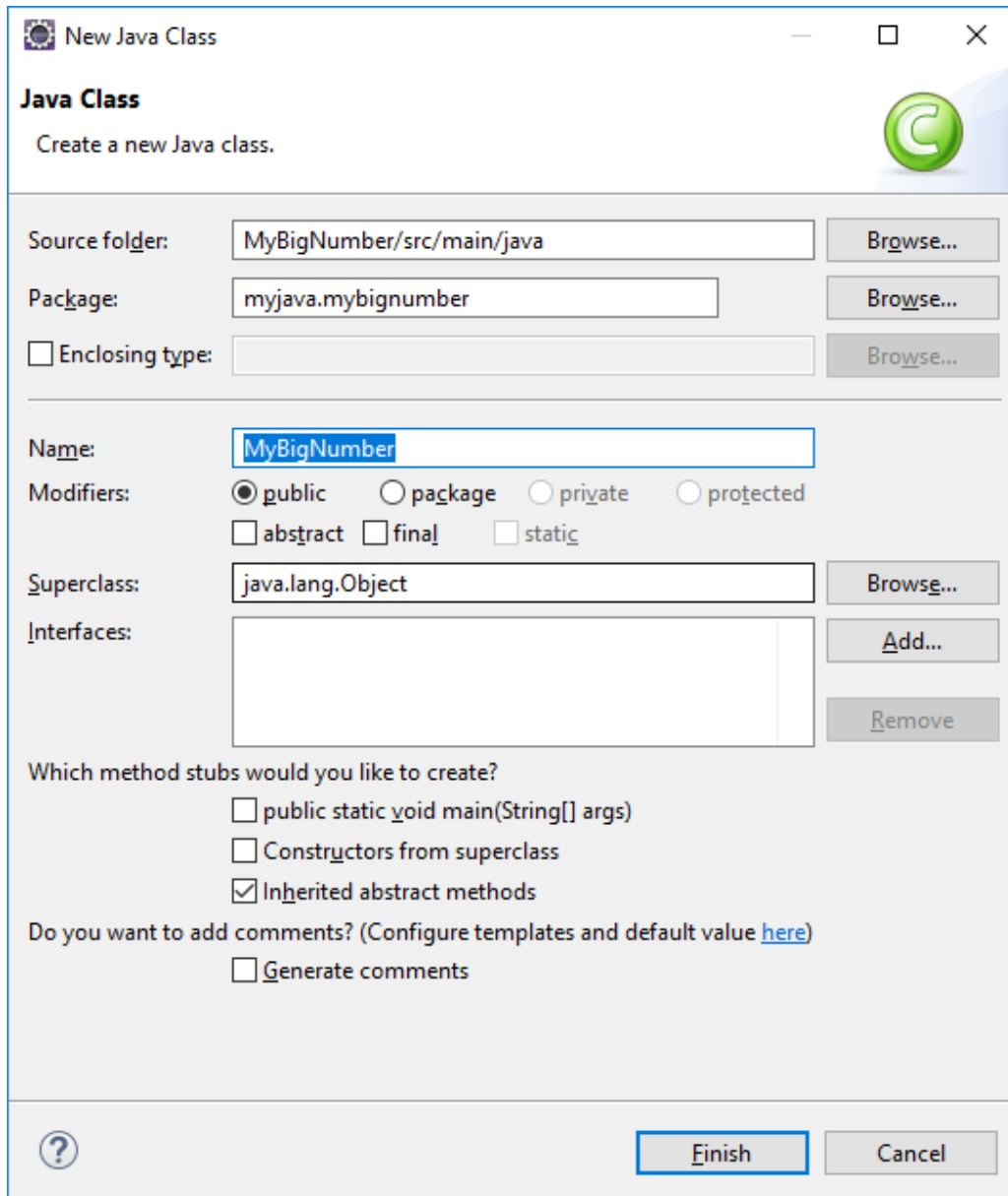
Eclipse sẽ khởi tạo project với khung code mẫu có một file mã nguồn “MyBigNumberApplication.java” như bên dưới:



Tạm thời bạn chưa quan tâm đến file mã nguồn này mà tập trung vào khởi tạo lớp `MyBigNumber` của chúng ta.

Bạn nhấp phải chuột vào package `myjava.mybignumber` ở khung bên trái, chọn menu `New > Class`.





New Java Class

Java Class
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

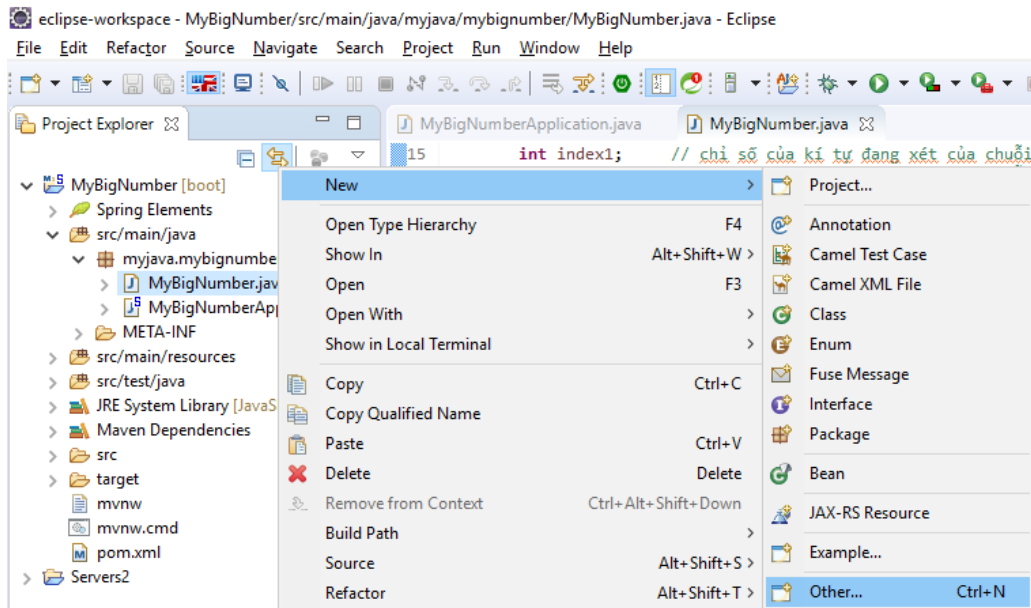
Điền tên class sẽ tạo vào mục Name: MyBigNumber.

Sau đó nhấn nút “Finish” và bắt đầu code method sum như tôi đã hướng dẫn ở trên. Bạn có thể copy và paste từ [source version 1](#).

Tạo Test Case

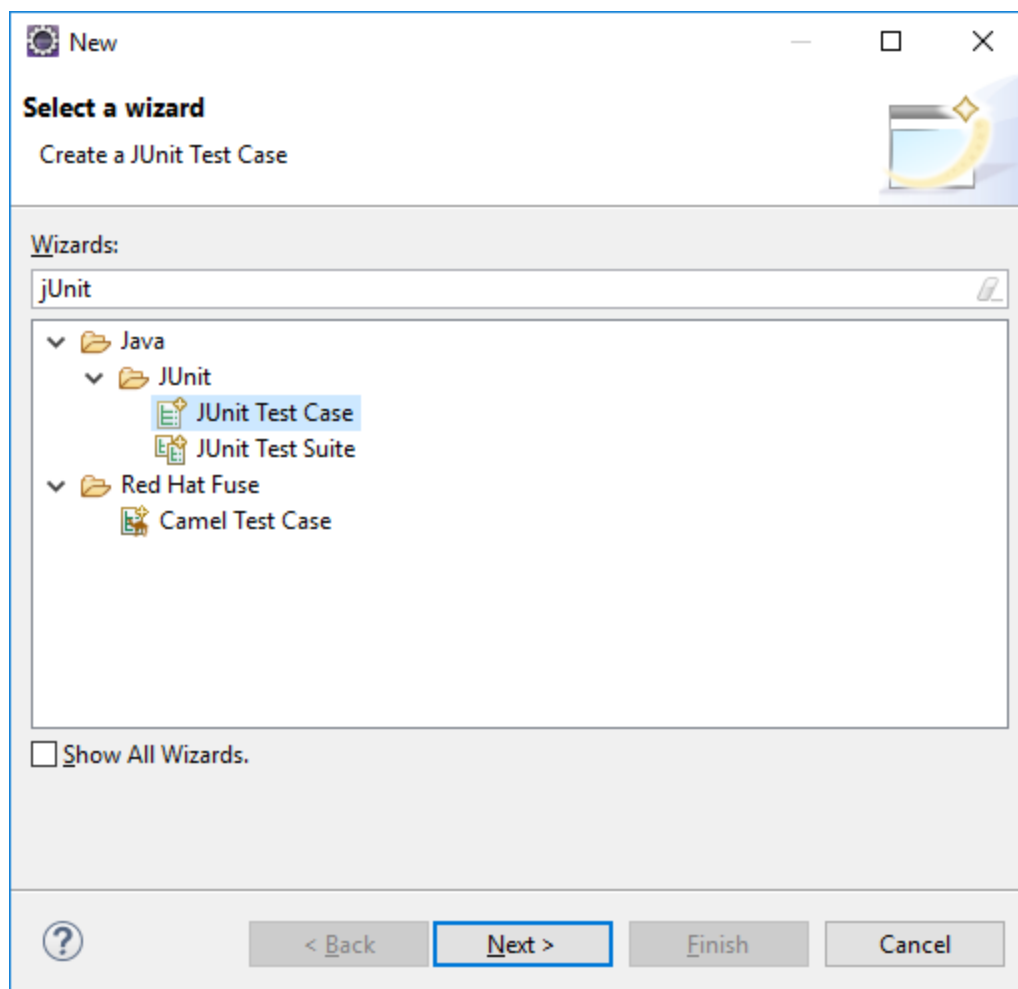
Lúc này bạn đã có source của của method sum(s1, s2). Để tạo ra code kiểm thử thì bạn nhấn phải chuột vào tên file

“MyBigNumber.java” ở khung Project Explorer bên trái, chọn menu New > Other...



Hoặc nhấn phím tắt “Ctrl + N”.

Trong hộp thoại “New”, chọn mục Java > JUnit > JUnit Test Case.



Nhấn nút “Next” để tiếp tục.

New JUnit Test Case

Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.

☐ New JUnit 3 test ☐ New JUnit 4 test ☒ New JUnit Jupiter test

Source folder:

Package:

Name:

Superclass:

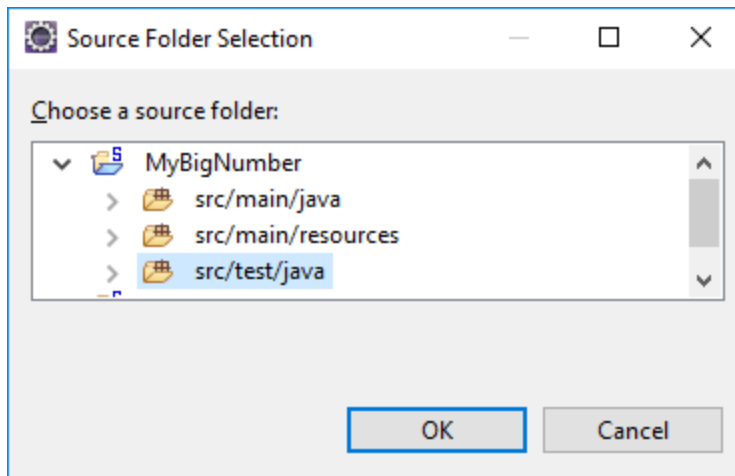
Which method stubs would you like to create?

☐ setUpBeforeClass() ☐ tearDownAfterClass()
☐ setUp() ☐ tearDown()
☐ constructor

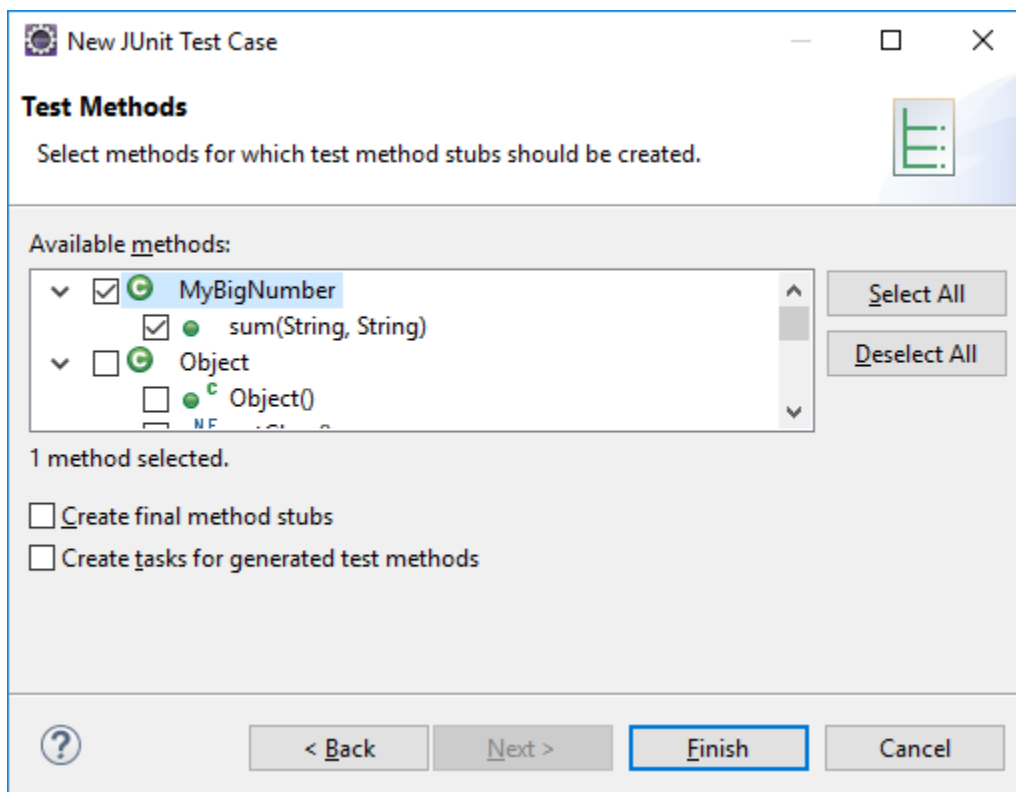
Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Class under test:

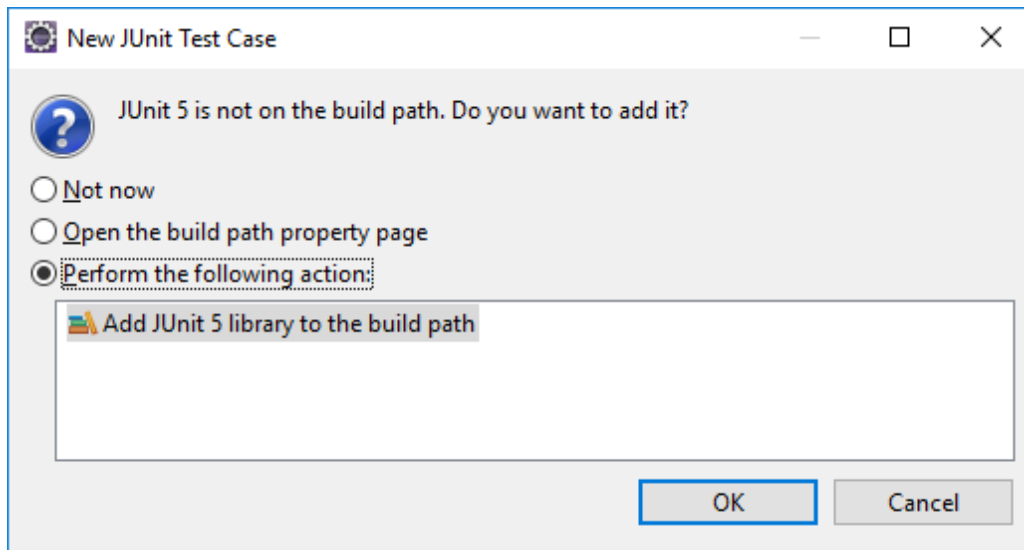
Trong hộp thoại “New JUnit Test Case” bạn chọn mục “New JUnit Jupiter test”. Mục Source folder, bạn bấm nút “Browse...” bên phải để chọn thư mục “src/test/java”.



Sau đó nhấn nút “Next”.



Đánh dấu vào mục `sum(String, String)` rồi nhấn nút Finish.



Nhấn nút “OK” để xác nhận Eclipse sẽ dùng thư viện JUNIT 5 cho project.

Source code JUNIT được tạo ra như sau:

```
package myjava.mybignumber;

import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

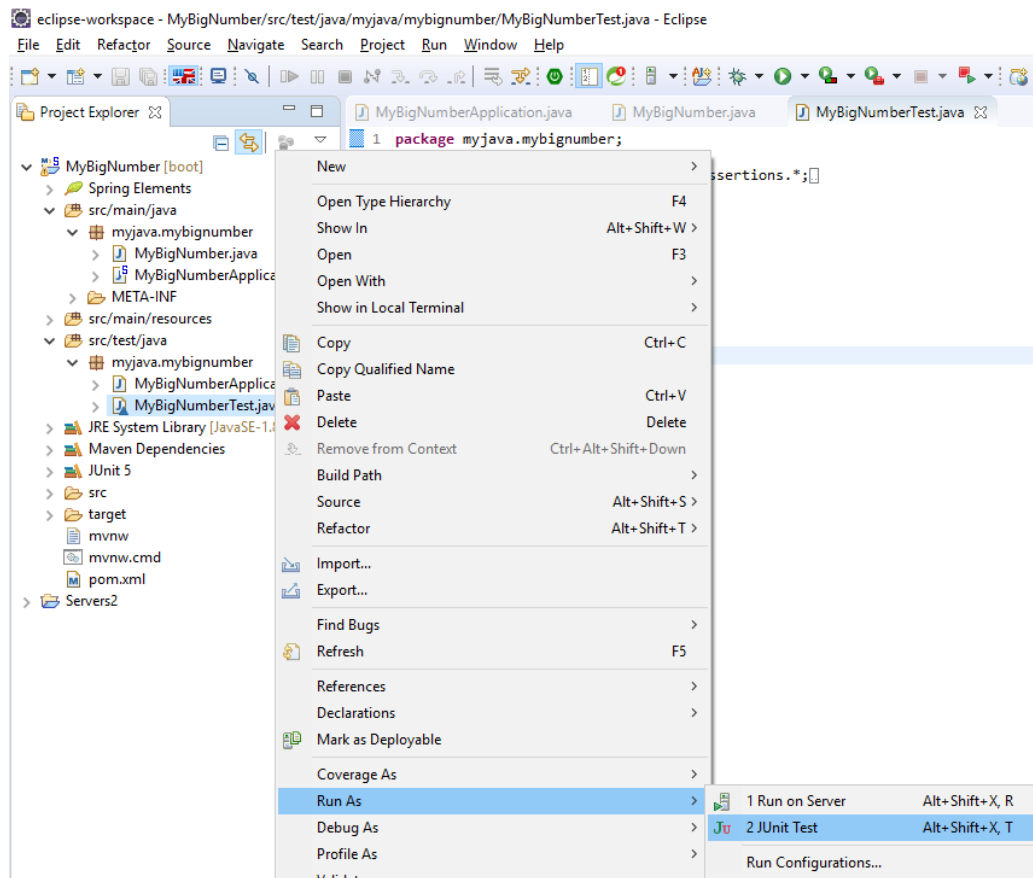
class MyBigNumberTest {

    @Test
    void testSum() {
        fail("Not yet implemented");
    }

}
```

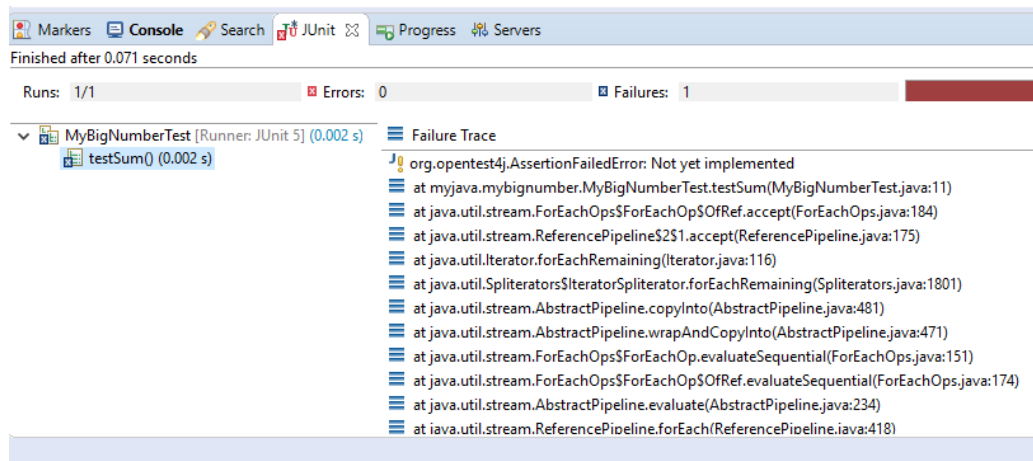
Cú pháp annotation `@Test` phía trước method `void testSum()` có nghĩa là method `testSum` là test case, sẽ được thư viện JUNIT gọi chạy.

Để chạy Test Case này, bạn nhấn phải chuột vào file `MyBigNumberTest.java`, chọn menu `Run As > JUnit Test`.



Hoặc bấm trái chuột vào file MyBigNumberTest.java rồi nhấn phím tắt “Alt + Shift + X”, sau đó nhấn tiếp phím T.

Kết quả thực hiện Test Case sẽ hiện ra trong tab JUnit ở phía đáy màn hình Eclipse.



Do code JUNIT do Eclipse sinh ra có một Test Case “testSum”.

```
@Test
void testSum() {
    fail("Not yet implemented");
}
```

Nội dung test case là lệnh fail(“Not yet implemented”);

Bạn cần xoá dòng này đi và viết lại Test case để gọi method sum(s1, s2) trong class MyBigNumber.

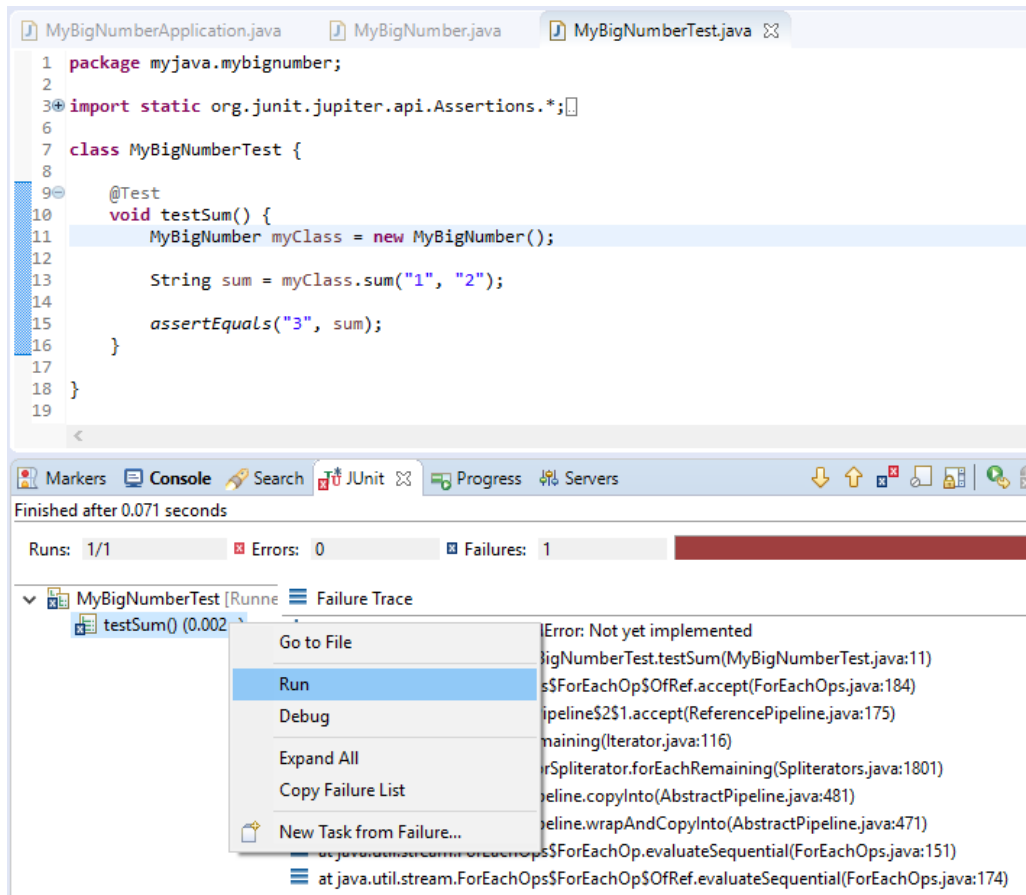
```
@Test
void testSum() {
    MyBigNumber myClass = new MyBigNumber();
    String sum = myClass.sum("1", "2");
    assertEquals("3", sum);
}
```

Nội dung test case “testSum” được viết lại gồm có 3 dòng. Hai dòng đầu là khởi tạo lớp cần test và gọi method để test. Cụ thể là ta cần test thử method sum với 2 chuỗi “1” và “2”.

Dòng thứ ba dùng method assertEquals để kiểm tra kết quả. assertEquals ở đây có 2 tham số. Tham số thứ nhất là giá trị mong đợi (1 cộng với 2 sẽ mong đợi kết quả là 3. Vì là kiểu chuỗi nên ta để trong cặp nháy đôi “3”). Tham số thứ hai là biến cần so sánh với giá trị mong đợi.

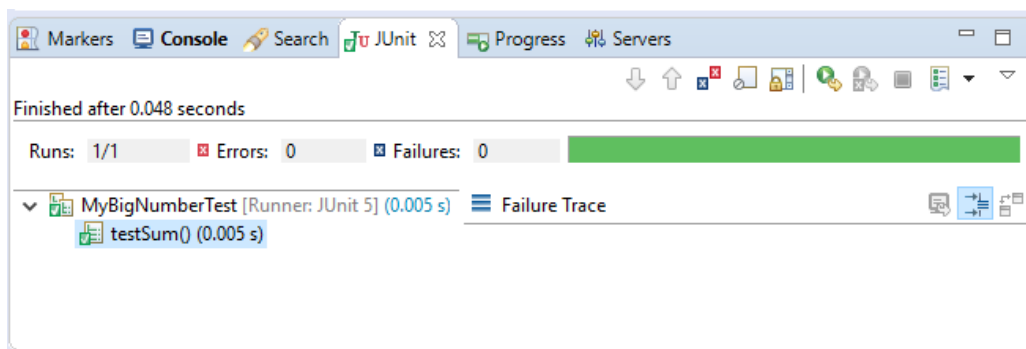
Chạy lại test case sau khi đã chỉnh bằng phím tắt Alt + Shift + X, T.

Một cách khác để chạy lại test case là trong cửa sổ JUnit, bấm phải chuột vào test case (tên method), chọn menu Run.



Ghi nhớ
Code xong một
method thì nên
viết Test Case để
kiểm tra.

Kết quả ra như sau:



Dấu stick màu xanh bên trái test case “testSum()” cho biết là đoạn code test đã chạy không có lỗi. Tức là kết quả sum(“1”, “2”) trả lại đúng bằng “3”.

Bạn có thể thử thêm vài phép cộng khác với độ dài 2 chuỗi bằng nhau. Ví dụ tôi thêm hai dòng bôi vàng bên dưới để thử cộng 2 số lớn có nhớ.

```
@Test
void testSum_N_1() {
    MyBigNumber myClass = new MyBigNumber();
    String sum = myClass.sum("1", "2");

    assertEquals("3", sum);

    sum = myClass.sum("8", "9");

    assertEquals("17", sum);
}
```

Trong đoạn code test ở trên tôi có sửa lại tên method là `testSum_N_1()`. Tên này giúp cho chúng ta gợi nhớ: N là Normal (tức là test trường hợp bình thường, tức là dữ liệu đúng), số 1 là trường hợp thứ nhất. Với tên test case như vậy bạn có thể copy thành nhiều test case khác như sau. Chú ý copy cả anotation `@Test`.

```
class MyBigNumberTest {

    @Test
    void testSum_N_1() {
        MyBigNumber myClass = new MyBigNumber();
        String sum = myClass.sum("1", "2");

        assertEquals("3", sum);

        sum = myClass.sum("8", "9");

        assertEquals("17", sum);
    }

    @Test
    void testSum_N_2() {
        MyBigNumber myClass = new MyBigNumber();

        String sum = myClass.sum("123", "9");

        assertEquals("132", sum);
    }
}
```

Bạn hãy chạy lại 2 test case này bằng phím tắt `Alt + Shift + X, R`. Kết quả như thế nào?

The screenshot shows an IDE with three tabs: `MyBigNumberApplication.java`, `MyBigNumber.java`, and `MyBigNumberTest.java`. The `MyBigNumber.java` tab is active, showing the following code:

```

16  int index2;    // chỉ số của ký tự đang xét của chuỗi 2
17  char c1;      // ký tự tại vị trí index1 của chuỗi s1
18  char c2;      // ký tự tại vị trí index2 của chuỗi s2
19  int d1;       // ký số của c1
20  int d2;       // ký số của c2
21  int t;        // tổng tạm của d1 và d2;
22  int mem = 0;  // nhớ nếu t lớn hơn hoặc bằng 10
23
24  /// Lặp maxlen lần
25  for (int i = 0; i < maxlen; i++) {
26      index1 = len1 - i - 1;
27      index2 = len2 - i - 1;
28
29      c1 = s1.charAt(index1);
30      c2 = s2.charAt(index2);
31
32      d1 = c1 - '0';
33      d2 = c2 - '0';
34
35      t = d1 + d2;
36
37      // lấy hàng đơn vị của t ghép vào phía trước kết quả
38      finalResult = (t % 10) + finalResult;
39      mem = t / 10;

```

Below the code editor, the IDE shows the results of a unit test run. The test suite is `MyBigNumberTest` with 2/2 runs, 1 error, and 0 failures. The failure trace for the test `testSum_N_20` is as follows:

```

Failure Trace
java.lang.StringIndexOutOfBoundsException: String index out of range: -1
    at java.lang.String.charAt(String.java:658)
    at myjava.mybignumber.MyBigNumber.sum(MyBigNumber.java:30)
    at myjava.mybignumber.MyBigNumberTest.testSum_N_2(MyBigNumberTest.java:26)
    at java.util.stream.ForEachOps$ForEachOp$OfRef.accept(ForEachOps.java:184)
    at java.util.stream.ReferencePipeline$2$1.accept(ReferencePipeline.java:175)
    at java.util.Iterator.forEachRemaining(Iterator.java:116)
    at java.util.Spliterators$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
    at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:481)
    at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)
    at java.util.stream.ForEachOps$ForEachOp.evaluateSequential(ForEachOps.java:151)
    at java.util.stream.ForEachOps$ForEachOp$OfRef.evaluateSequential(ForEachOps.java:171)
    at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
    at java.util.stream.ReferencePipeline.forEach(ReferencePipeline.java:419)

```

Ghi nhớ
Phải đọc kỹ từng dòng, từ chữ của thông báo lỗi.

Phân tích lỗi Unit Testing

Kết quả test case “testSum_N1()” là OK nhưng test case “testSum_N_2()” bị báo đỏ. Khung “Failure Trace” bên phải cung cấp thông tin chi tiết hơn về lỗi. Cụ thể:

- Dòng đầu tiên:
`java.lang.StringIndexOutOfBoundsException: String index out of range: -1`
Có nghĩa là lỗi index vượt ngoài giới hạn. index = -1.
- Dòng thứ 2:
`at java.lang.String.charAt(String.java:658)`

Đây là lỗi tại dòng 658 trong lớp String của Java, tại method `charAt`.

Chúng ta sẽ bỏ qua, không phân tích lỗi xuất hiện trong source code của Java.

- Dòng thứ 3: đây là lỗi của chúng ta

`at`

```
myjava.mybignumber.MyBigNumber.sum(MyBig  
Number.java:30)
```

Trong source code `MyBigNumber.java` tại dòng 30 gây ra lỗi index vượt ngoài giới hạn. Bạn double-click vào dòng thứ ba này để Eclipse mở source code cho chúng ta xem.

Vì chúng ta đang test thử `sum("123", "9")` thì rõ ràng khi chỉ số `index1` chạy từ phải sang tới ký tự '2' thì đối với chuỗi `s2` đã hết chuỗi (tức là `index2` đã lùi về tới -1, bị âm). Điều này gây ra lỗi trong test case số 2.

Tóm tắt:

Như vậy đến thời điểm này bạn đã code được luồng logic chính để cộng hai số dạng chuỗi. Bạn cũng đã biết cách viết Test Case dùng JUNIT để test chương trình thay vì viết method `main` như thói quen thông thường.

Bạn cũng đã đưa vào một test case để test method `sum` với 2 tham số có độ dài khác nhau. Kết quả đương nhiên là **FAILED**.

Việc tiếp theo của chúng ta là tiếp tục nâng cấp code để giải quyết tiếp các tình huống phụ.

Giai đoạn hai – cài đặt các logic phụ

Tôi có gắng trình bày thành hai phần logic chính và logic phụ là để bạn có thói quen khi lập trình nên tập trung vào chức năng chính trước, hình thành được khung code giải quyết được các vấn đề quan trọng; trình bày code trong sáng, dễ hiểu.

Sau đó mới cải tiến tiếp để giải quyết các luồng xử lý phụ khác.

Cụ thể trong source code ở trên, chúng ta sẽ cần nâng cấp tiếp khi chỉ số `index1` và `index2` đã lùi về bên trái đến hết chuỗi.

Đoạn code tính `c1`, `c2` được tinh chỉnh lại như sau:

```
c1 = (index1 >= 0) ? s1.charAt(index1) : '0';  
c2 = (index2 >= 0) ? s2.charAt(index2) : '0';
```

Dùng toán tử 3 ngôi cho gọn thay vì dùng `if...else`.

Một số điểm bạn có thể cải tiến tiếp:

- `d1`, `d2` là các biến trung gian để cho code dễ hiểu. Việc gán `c1`, `c2` bằng ký tự '0' có thể hơi thừa. Nếu muốn tối ưu code thêm thì có thể gán `d1`, `d2` bằng zero luôn khi `index1`, `index2` bị âm.

Sau khi sửa được lỗi khi `index1`, `index` âm thì bạn chạy lại Test Case sẽ phát hiện test case số 2 vẫn bị FAILED. Bạn tập phân tích kết quả test case và phát hiện dòng lệnh `t = d1 + d2` quên cộng thêm biến nhớ "mem". Source code của vòng `for` sẽ sửa thêm một chút:

```
for (int i = 0; i < maxlen; i++) {  
    index1 = len1 - i - 1;  
    index2 = len2 - i - 1;  
  
    c1 = (index1 >= 0) ? s1.charAt(index1) : '0';  
    c2 = (index2 >= 0) ? s2.charAt(index2) : '0';  
  
    d1 = c1 - '0';  
    d2 = c2 - '0';  
  
    t = d1 + d2 + mem;  
  
    // Lấy hàng đơn vị của t ghép vào phía trước kết quả  
    finalResult = (t % 10) + finalResult;  
    mem = t / 10;  
}
```

Tự trải nghiệm

- Thêm các test case cộng số có nhiều chữ số hơn.

Nâng cấp version 2

Để method `sum(String s1, String s2)` có thể cung cấp được cho người khác sử dụng thì chúng ta cần phải xử lý nhiều tình huống hơn. Cụ thể là cần bổ sung thêm 3 tính năng:

- 1) Nếu các tham số `s1`, `s2` truyền vào là null hoặc emty (rỗng) thì xem như là zero. Ví dụ `sum("", "2")` hoặc `sum(null, "2")` sẽ cho kết quả giống như `sum("0", "2")`.
- 2) Nếu các tham số `s1`, `s2` truyền vào có chứa các ký tự không phải là ký số '0'..'9' thì method sẽ throw (văng ra / ném ra) Exception với các thông chi tiết như sau:
 - Exception class:
`java.lang.NumberFormatException (String msg)`
 - msg: là nội dung của Exception có dạng:

Lỗi ở tham số <param> tại vị trí <index>: <invalid char>.

- <param>: là `s1` hoặc `s2`
- <index>: là vị trí chứa ký tự lỗi tính theo index của String trong java.
- <invalid char>: là ký tự gây lỗi.

Ví dụ: `sum("123", "1a2")` sẽ throw ra `NumberFormatException("Lỗi ở tham số s2 tại vị trí 1: a")`.

Nâng cấp version 3

Bổ sung thêm tính năng cho version 2.

- 3) Nếu tham số `s1`, `s2` truyền vào là các ký số nhưng có dấu trừ ở đầu, tức là số âm thì method `sum(s1, s2)` sẽ throw ra `java.lang.NumberFormatException (String msg)` với msg có dạng:

Chưa hỗ trợ số âm <param>: <value>

- <param>: là s1 hoặc s2
- <value>: là giá trị tương ứng của param

Ví dụ: `sum("-123", "45")` sẽ throw ra `NumberFormatException("Chưa hỗ trợ số âm s1: -123")`.

Nâng cấp version 4

Lớp `NumberFormatException` sử dụng message làm tham số trong Constructor không được tốt lắm về mặt thiết kế.

Chúng ta sẽ nâng cấp một chút bằng cách tự viết một lớp `ExNumberFormatException(Integer errorPos)` với tham số là `Integer` để chỉ vị trí lỗi của kí tự không hợp lệ.

Code test cũng sẽ nâng cấp một chút bằng cách kiểm tra 2 nội dung:

- Kiểm tra đúng loại `Exception`
- Kiểm tra thông tin trong `Exception` chính xác. Ở đây là `errorPos`

Bước 3.1: Kiểm thử mã nguồn

Trong bước 3 ở trên bạn đã làm quen với cách dùng JUNIT để tạo ra test code gọi method cần test để thực thi. Bạn cũng biết cách dùng method `assertEquals` trong JUNIT để kiểm tra kết quả có đúng mong đợi hay không.

Câu hỏi?

Các test case bạn viết như vậy đã kiểm tra đủ tất cả các tình huống chưa? Bạn viết bao nhiêu test case là đủ?

Trả lời của bạn:

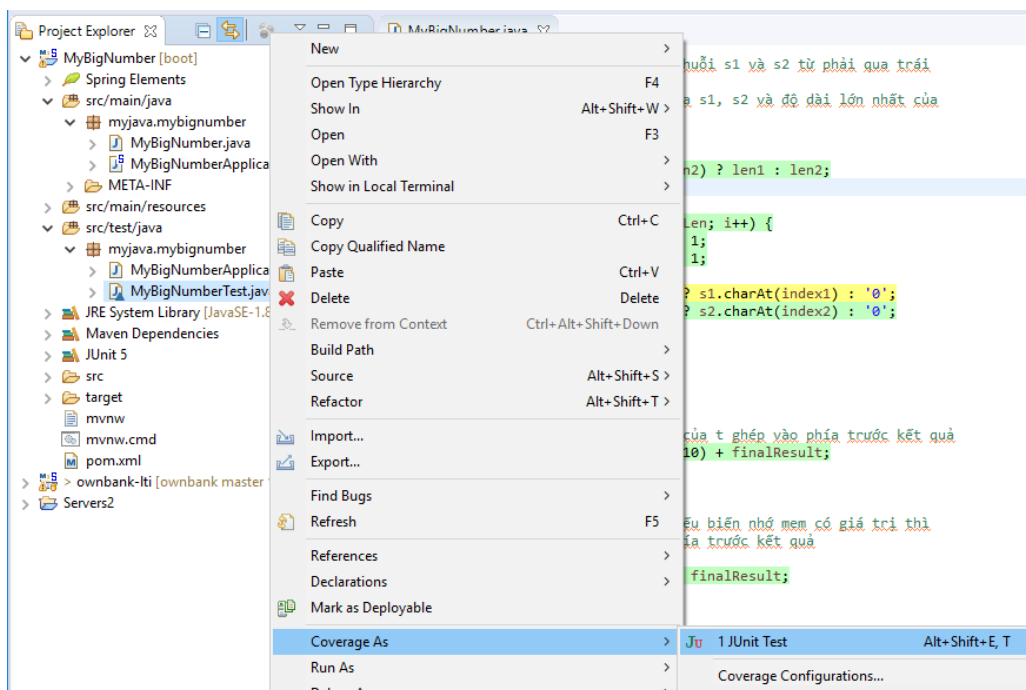
.....

.....

.....

Để trả lời câu hỏi trên, bạn hãy thử dùng chức năng “Coverage” trong Eclipse để kiểm tra mức độ bao phủ mã nguồn của các test case. Cách làm như sau:

Bấm phải chuột lên file Test Case “MyBigNumberTest.java” ở khung Project Explorer bên trái. Chọn menu Coverage As > JUnit Test (phím tắt là Alt + Shift + E, T)



Kết quả sẽ được trình bày trong tab “Coverage” như hình bên dưới:

The screenshot shows an IDE with a Java project named 'MyBigNumber'. The main editor displays the source code of 'MyBigNumber.java', which implements a recursive algorithm to calculate the sum of digits of a number. The code is annotated with comments in Vietnamese. Below the code editor, the 'Coverage' tab is active, showing a table of coverage data for the project.

Element	Coverage	Covered Instruction	Missed Instructions	Total Instructions
MyBigNumber	91.8 %	146	13	159
src/main/java	91.4 %	96	9	105
myjava.mybignumber	91.4 %	96	9	105
MyBigNumberApplication.java	0.0 %	0	8	8
MyBigNumberTest.java	99.0 %	96	1	97
src/test/java	92.6 %	50	4	54

Trong tab “Coverage” cho bạn biết các thông tin sau: Với bộ test case MyBigNumberTest.java thì class MyBigNumber.java đã được thực thi (xem cột Element), độ bao phủ (cột Coverage) là 99.0%.

Coverage được tính bằng: lấy số dòng lệnh đã được thực thi chia cho tổng số dòng lệnh có trong class.

$$\text{Coverage} = \frac{\text{Số dòng lệnh đã thực thi}}{\text{Tổng số dòng lệnh đã viết}} \times 100$$

Ý nghĩa 3 cột tiếp theo có nghĩa là:

- Covered Instructions: số dòng lệnh đã thực thi (đã chạy).
- Missed Instructions: số dòng lệnh chưa chạy (xem các dòng bôi đỏ, bôi vàng trong khung source code).
- Total Instructions: tổng số dòng lệnh.

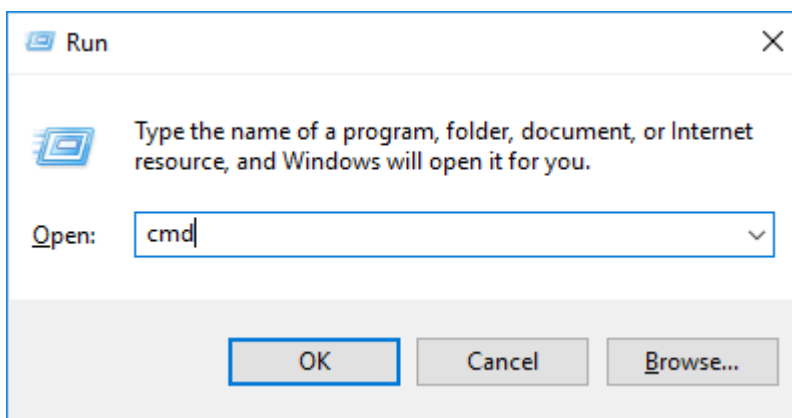
Ghi nhớ
Cố gắng đảm bảo Coverage là 100%.

Như vậy một định hướng cho chúng ta là cần phải bổ sung test case sao cho khi thực thi thì Coverage là 100%.

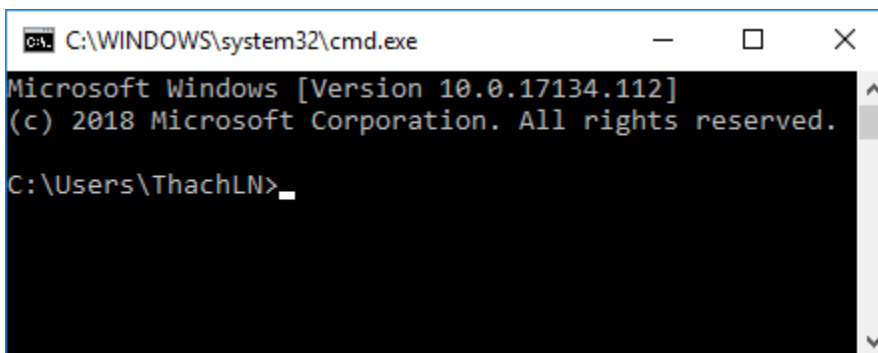
Bước 3.2: Hoàn thiện giao diện

Trong bước 2 tôi có thiết kế giao diện của ứng dụng đơn giản là console. Tức là phần mềm “sum” cho phép người dùng gõ lệnh và chạy trong cửa sổ lệnh của hệ điều hành. Tùy theo hệ điều hành thì cửa sổ lệnh có tên gọi khác nhau.

Trong Windows, để mở cửa sổ lệnh thì nhấn phím tắt Windows + R (Hầu hết các bàn phím trên máy PC hoặc laptop mới thì có phím có cửa sổ - biểu tượng của Microsoft Windows, gần phím Alt bên trái) để mở hộp thoại Run:



Trong mục Open, bạn gõ chữ “cmd” (không có dấu nháy) rồi nhấn Enter. Cửa sổ lệnh sẽ hiện ra như bên dưới:



Quay lại phần mềm Cộng hai số, bạn cần tạo một class có method main để thực thi chương trình. Nếu bạn dùng Springboot thì sẽ có sẵn class này.

Nhiệm vụ của bạn

Viết code cho method main để lấy các tham số, khởi tạo đối tượng MyBigNumber và gọi method sum để tính toán.

Bước 4: Kiểm thử hệ thống

Sau khi bạn đã lập trình và kiểm thử xong trong bước 3. Bây giờ là lúc bạn đóng gói và thử triển khai phần mềm của mình lên máy của một người khác để kiểm tra lại. Trong công nghệ phần mềm, bước này gọi là Integration Test hoặc System Testing. Hai chủ đề này không nằm trong phạm vi của cuốn sách này. Trong một phần mềm rất nhỏ như Cộng hai số thì hai thuật ngữ này có thể xem là một.

Nhiệm vụ của bạn trong bước này là đóng gói phần mềm và cài đặt phần mềm vào một máy khác để thực hiện System Testing. Để đơn giản thì chúng ta có thể gọi là chạy thử.

Đóng gói

Trong bước 3, bạn đã tạo Project dùng Springboot. Chúng ta cần phần mềm maven để biên dịch và đóng gói chương trình.

Cài đặt maven

Để cài đặt maven (<http://maven.apache.org>) bạn thực hiện các bước sau:

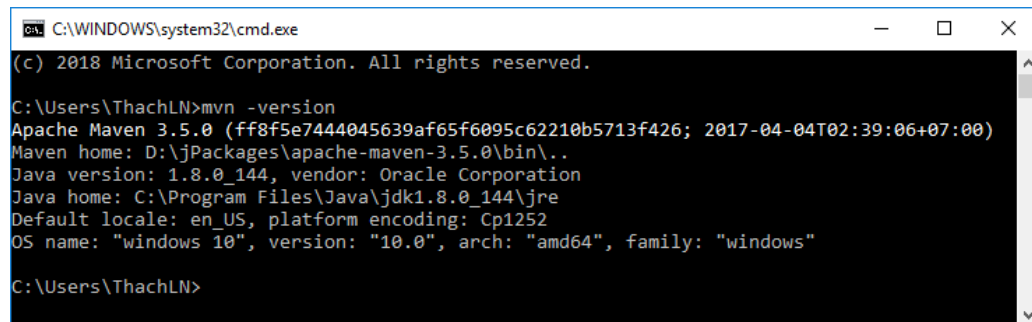
- Tải gói binary. Vd: tải gói “<http://mirrors.viethosting.com/apache/maven/maven-3/3.5.4/binaries/apache-maven-3.5.4-bin.tar.gz>” về thư mục “D:\jPackages”.
- Extract maven. Vd: dùng phần mềm 7-zip (<https://7zip.org>) để extract ra thư mục “D:\jPackages\apache-

maven-3.5.0” sao cho thư mục “bin” có đường dẫn là “D:\jPackages\apache-maven-3.5.0\bin”.

- Thêm biến môi trường “M2_HOME” vào trong hệ điều hành. M2_HOME có giá trị là:
D:\jPackages\apache-maven-3.5.0
- Cập nhật biến môi trường PATH của hệ điều hành bằng cách thêm đường dẫn: %M2_HOME%\bin.
- Kiểm tra lại maven đã cài đặt đúng chưa bằng cách mở cửa sổ lệnh gõ:

```
mvn -version
```

Kết quả tương tự như sau:



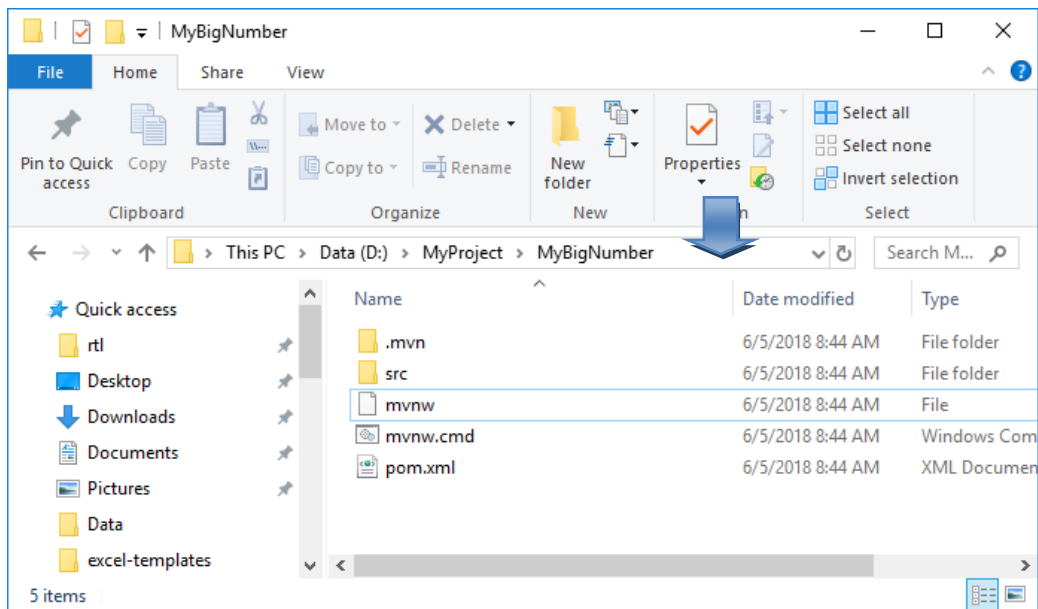
```
C:\WINDOWS\system32\cmd.exe
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ThachLN>mvn -version
Apache Maven 3.5.0 (ff8f5e7444045639af65f6095c62210b5713f426; 2017-04-04T02:39:06+07:00)
Maven home: D:\jPackages\apache-maven-3.5.0\bin\..
Java version: 1.8.0_144, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.8.0_144\jre
Default locale: en_US, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

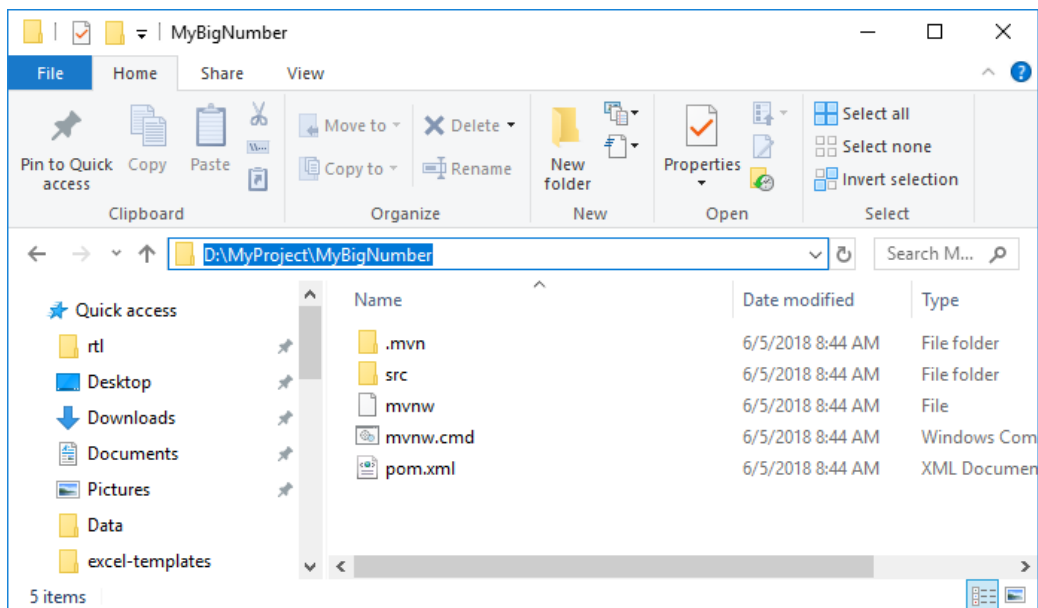
C:\Users\ThachLN>
```

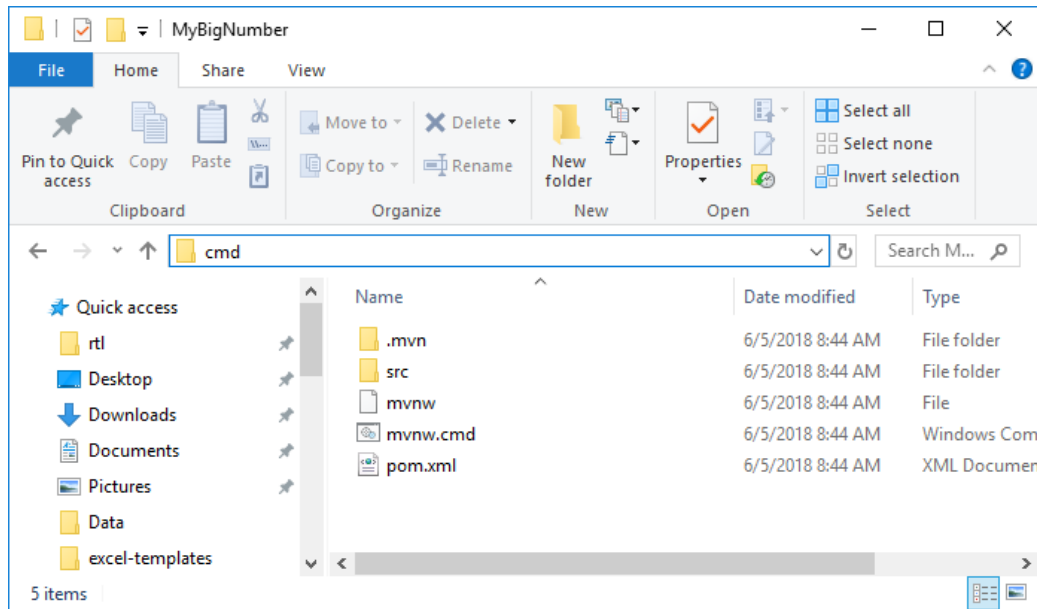
Tạo file .jar

Bạn cần mở cửa sổ lệnh với thư mục làm việc (working directory) là thư mục mã nguồn của dự án. Trong Windows Explorer đang mở thư mục dự án, bạn có thể mở nhanh cửa sổ lệnh bằng cách bấm chuột vào thanh Address – nhớ bấm vào chỗ trống bên phải tên thư mục.

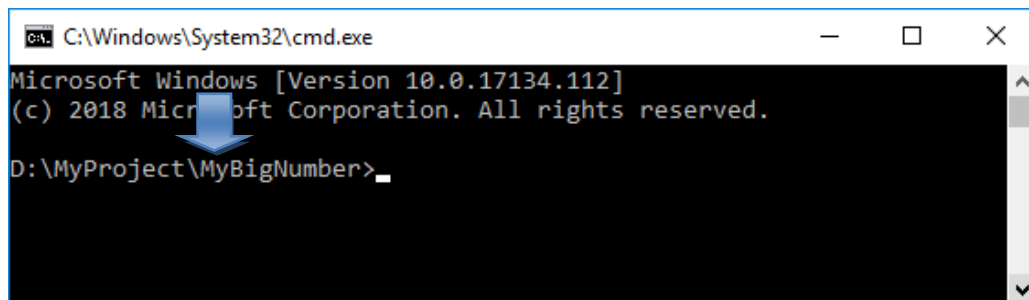


Lúc đó đường dẫn thư mục của dự án được bôi xanh, bạn gõ luôn chữ rồi gõ “cmd” (không có dấu nháy). Sau đó nhấn Enter.





Cửa sổ lệnh hiện ra với đường dẫn của dự án như bên dưới:



Tiếp theo, bạn gõ lệnh “mvn clean package” rồi nhấn Enter để biên dịch và đóng gói dự án. Kết quả của lệnh này là

Trong trường hợp quá trình biên dịch bị lỗi thì bạn thử lệnh:

```
mvn clean package -Dmaven.test.skip
```

Tham số “-Dmaven.test.skip” có nghĩa là bỏ qua quá trình testing.

Sau khi có kết quả **"BUILD SUCCESS"** thì bạn lấy kết quả file jar tại thư mục `".\target\MyBigNumber-0.0.1-SNAPSHOT.jar"` tính tại thư mục bạn đang làm việc.

Chạy thử chương trình trong file .jar

Thực thi chương trình bằng đánh lệnh:

```
java -jar ./target/MyBigNumber-0.0.1-SNAPSHOT.jar  
"1" "2"
```

Như vậy bạn đã biên dịch, tạo file .jar cho chương trình và có thể thực thi chương trình thông qua gói .jar. Tuy nhiên việc gõ lệnh để chạy chương trình như trên sẽ gây khó khăn cho người dùng.

Bạn hãy suy nghĩ cách để cải tiến cách đóng gói sản phẩm cho người dùng tiện lợi hơn xem!

Triển khai

Bạn tạo một thư mục “Release\sum” để chuẩn bị đóng gói sản phẩm cho khách hàng theo các bước như sau:

- Copy file MyBigNumber-0.0.1-SNAPSHOT.jar vào thư mục “Release\sum”.
- Trong thư mục “Release\sum”, tạo file sum.cmd có nội dung sau:

```
java -jar D:/sum/MyBigNumber-0.0.1-SNAPSHOT.jar %1 %2
```

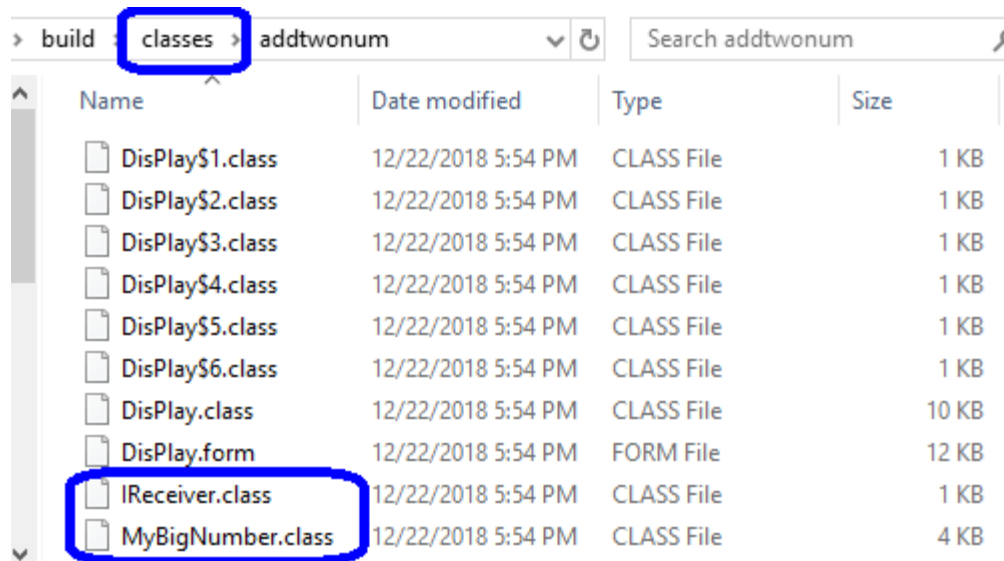
Để triển khai chương trình Cộng hai số bạn thực hiện 2 bước sau:

- Copy thư mục “sum” ở trên vào ổ đĩa D: của máy khách hàng.
- Thêm đường dẫn “D:/sum” vào biến môi trường PATH trên máy khách hàng.
- Mở cửa sổ lệnh gõ thử lệnh: sum “1” “2”

Nâng cao

Khi muốn chủ động đóng gói vài file .class vào thành thư viện .jar thì bạn có thể dùng lệnh “jar”.

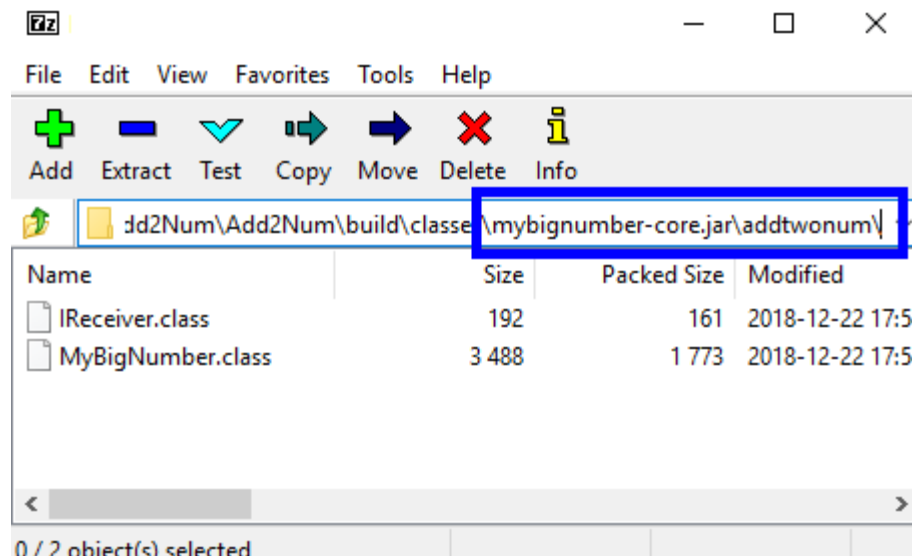
Ví dụ trong hình bên dưới bạn muốn đóng gói 2 file IReciver.clas và MyBigNumer.class thành thư viện mybignumber-core.jar?



Mở cmd trong thư mục “classes”, thực thi lệnh sau (viết trên 1 dòng):

```
jar cvf mybignumber-core.jar .\addtwonum\MyBigNumber.class
.\addtwonum\IReceiver.class
```

Sau đó dùng 7-zip ở open file “mybignumber-core.jar” trong thư mục “classes” để kiểm tra lại nội dung file:





Thử thách cho bạn: #1

Chủ đầu tư có nhu cầu nâng cấp phần mềm này cho phép cộng hai số lớn được lưu trong file text.

Cú pháp sử dụng như sau:

```
sum <file path 1> <file path 2> -t file
```

<file path 1>: là đường dẫn của file văn bản chứa các kí số đầu tiên.

<file path 2>: là đường dẫn của file văn bản chứa các kí số thứ hai.

-t file: cho biết là kiểu file (t: viết của chữ type)

Gợi ý

- Áp dụng tính Overload trong lập trình Java để bổ sung method cho source code có sẵn.



Thử thách cho bạn: #2

Áp dụng các kiến thức, kỹ năng đã học hãy viết một ứng dụng loại bỏ dấu tiếng Việt (Remove Tone Marks) trong một văn bản có sẵn. Hỗ trợ tiếng Việt với bảng mã UTF-8.

Phiên bản 1: Hỗ trợ truyền văn bản trên dòng lệnh

```
rtm "Chúng tôi là Lập trình viên Java chuyên nghiệp."
```

Vd: lệnh

```
rtm "Chúng tôi là Lập trình viên Java chuyên nghiệp."
```

sẽ in ra màn hình câu bên dưới (không có kẻ ô chữ nhật):

Chúng tôi là Lập trình viên Java chuyên nghiệp.

Phiên bản 2: Hỗ trợ truyền đường dẫn file đầu vào và file đầu ra trên dòng lệnh

`rtm <input file> <output file> -t file`

<input file>: là đường dẫn của file văn bản chứa chứa tiếng Việt UTF-8.

<output file>: là đường dẫn của file kết quả sau khi đã loại bỏ tiếng Việt.

-t file: cho biết là kiểu file (t: viết của chữ type)

Gợi ý

- Luyện tập kỹ năng viết yêu cầu ra file High Level Requirement.
- Luyện tập kỹ năng viết thuật toán ra giấy hoặc ra file trên máy tính. Suy nghĩ kỹ cách làm thật đơn giản, dễ hiểu như cách suy nghĩ thông thường – chưa cần phải tối ưu.
- Luyện tập cách trang Javadoc API JDK để tìm method xử lý chuỗi (String) phù hợp.

Tham khảo thêm source code bằng CSharp:

```
static string sum(string number1, string number2)
{
    int number1Len = number1.Length, number2Len = number2.Length;
    int lenMax = number1Len > number2Len ? number1Len : number2Len;
    string result = String.Empty;
    int temp;
    int idx1, idx2;
    char c1, c2;
    int d1, d2;
    int mem = 0;
```

```
for (int i = 0; i < lenMax; i++)
{
    idx1 = number1Len - i - 1;
    idx2 = number2Len - i - 1;
    c1 = (idx1 >= 0) ? number1[idx1] : '0';
    c2 = (idx2 >= 0) ? number2[idx2] : '0';

    d1 = c1 - '0';
    d2 = c2 - '0';
    temp = d1 + d2 + mem;
    mem = temp / 10;
    temp = temp % 10;
    result = temp + result;
}

if (mem > 0) { result = mem + result; }

return result;
}
```

Tham khảo source code bằng C

```
char* addBigNumber(char* number1, char* number2)
{
    size_t number1Len = strlen(number1);
    size_t number2Len = strlen(number2);
    size_t temp;
    int idx1, idx2;
    char c1, c2;
    size_t d1, d2;
    size_t mem = 0;

    size_t lenMax = number1Len > number2Len ? number1Len : number2Len;

    char* result = new char[lenMax + 2];

    memset(result, ' ', lenMax);
    bool remember = false;

    for (int i = 0; i < lenMax; i++)
    {
        idx1 = number1Len - i - 1;
        idx2 = number2Len - i - 1;
```

```

        c1 = (idx1 >= 0) ? number1[idx1] : '0';
        c2 = (idx2 >= 0) ? number2[idx2] : '0';

        d1 = c1 - '0';
        d2 = c2 - '0';

        temp = d1 + d2 + mem;

        mem = temp / 10;
        temp = temp % 10;

        result[lenMax - i] = temp + '0'
    }

    result[lenMax + 1] = '\0';

    if (mem > 0) {
        result[0] = '1';
    }

    return result;
}

```

Tham khảo thêm source code bằng Python

```

def addBigNumber(number1, number2):
    number1Len = len(number1)
    number2Len = len(number2)
    lenMax = max(number1Len, number2Len)
    result = ""
    mem = 0

    for i in range(lenMax):
        idx1 = number1Len - i - 1
        idx2 = number2Len - i - 1

        c1 = number1[idx1] if idx1 >= 0 else '0'
        c2 = number2[idx2] if idx2 >= 0 else '0'

        d1 = int(c1)
        d2 = int(c2)
        temp = d1 + d2 + mem
        mem = int(temp / 10)
        temp1 = int(temp % 10)

```



```
result = str(temp1) + result

if mem > 0:
    result = str(mem) + result

return result
```

Bài 4 – Quen dần với phong cách lập trình chuyên nghiệp

Tầm quan trọng của Quy ước lập trình

Bên cạnh việc viết chương trình sao cho chạy được, chạy đúng như ý tưởng thiết kế thì lập trình viên phải chú ý đến các quy ước lập trình (Coding convention). Các quy ước này rất quan trọng trong việc phát triển phần mềm vì một số lý do chính như sau:

- 80% chi phí để tạo ra và duy trì phần mềm trong thực tế là chi phí để bảo trì (maintenance). Tức là phần mềm viết ra là để sửa chữa, để thay đổi.
- Hiếm có phần mềm nào được bảo trì lâu dài bởi chính tác giả của nó. Tức là hầu hết các lập trình viên là phải đi sửa, thay đổi mã nguồn của người khác viết ra.

Vì vậy việc tuân thủ Quy ước lập trình sẽ giúp cho việc đọc hiểu code từ một lập trình viên khác dễ dàng hơn. Từ đó cho phép người mới hiểu mã nguồn nhanh chóng, đầy đủ và dễ dàng nâng cấp phần mềm.

Các quy ước chung (General rules)

Đơn giản

Có gắng viết mã nguồn đơn giản và dễ hiểu. Đơn giản và dễ hiểu được thể hiện như sau:

- Mỗi dòng lệnh nên thực hiện một việc.

Code đơn giản
✓ Mỗi dòng lệnh làm một việc.
✓ Tách mã nguồn thành method riêng khi có thể

- Không khai báo nhiều biến trên cùng một dòng.
- Hạn chế gọi hàm trong các tham số của một hàm khác
- Chia để trị: Khi một đoạn mã nguồn trong một method quá dài (cỡ trên 100 dòng) và có thể tách thành hàm con thì nên tách method.

Rõ ràng

- Sử dụng khoảng trắng (space), dòng trắng (empty line) hợp lý.
 - Dùng khoảng trắng tại khi:
 - Sau từ khoá for, while, if, do
 - Trước và sau toán tử: +, -, *, /, ==, !=
 - Sau các dấu phẩy của tham số trong method
 - Sau dấu ; trong vòng lặp for
 - Trước và sau else trên cùng dòng đóng mở khối lệnh: `} else {`
 - Các khối lệnh được canh lề (indentation) 4 khoảng trắng so với khối lệnh cha: Bạn cần cấu hình IDE để khi nhấn phím tab thì IDE sẽ thay bằng 4 khoảng trắng.
 - Dùng dòng trắng tại các vị trí:
 - Giữa phần khai báo biến và lệnh (statement) đầu tiên
 - Giữa các phần lệnh có mục đích các nhau
 - Trước phần chú thích của method hoặc trước method khi chưa chú thích

Format code trong Eclipse

✓ Bôi chọn đoạn code rồi nhấn tổ hợp phím Ctrl + Shift + F để định dạng mã nguồn.

Chú thích Eclipse Java

✓ Chọn tên lớp, phương thức, biến của lớp. Nhấn phím Alt + Shift + J để

- Chú thích tài liệu đầy đủ theo quy ước của ngôn ngữ lập trình.
 - Chú thích đầy đủ cho lớp (Class)
 - Chú thích đầy đủ cho phương thức (Method)
 - Đối với Java, sử dụng IDE Eclipse có thể bấm chuột vào tên class, method, properties của lớp. Sau đó nhấn tổ hợp phím Alt + Shift + J để Eclipse sinh ra khung chú thích cho hàm.

sinh ra khung tài liệu.

Thống nhất

- Đặt tên gói (package) lớp, tên biến, thêm phương thức thống nhất theo một qui tắc tùy theo :
 - Tên package chỉ chứa kí tự thường
 - Tên lớp bắt đầu bằng ký tự Hoa; bắt đầu bằng danh từ (Noun)
 - Tên biến bắt đầu bằng ký tự thường
 - Tên phương thức bắt đầu bằng động từ.

[Còn tiếp]