UỶ BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG CAO ĐẲNG BÁCH KHOA NAM SÀI GÒN

GIÁO TRÌNH

MÔN HỌC/MÔ ĐUN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

NGÀNH/ NGHỀ: TIN HỌC ỨNG DỤNG

TRÌNH ĐỘ: CAO ĐẲNG

Ban hành kèm theo Quyết định số: 459/QĐ-NSG ngày 31 tháng 8 năm2022

của Hiệu trưởng Trường Cao Đẳng Bách Khoa Nam Sài Gòn

Tp.Hồ Chí Minh, năm 2022

TUYÊN BỐ BẢN QUYỀN

LỜI GIỚI THIỆU

Lập trình hướng đối tượng là mộn học thuộc khối kiến thức chuyên ngành dành cho đối tượng học sinh – sinh viên ngành/ nghề Tin học ứng dụng. Nhằm phục vụ cho việc giảng dạy và học tập môn lập trình hương đối tượng, nhóm tác giả chúng tôi biên soạn giáo trình này dựa trên khung chương trình chung môn tin học của Bộ Lao động Thương binh và Xã hội. Nội dung giáo trình gồm 9 bài:

- Bài 1: Mở đầu
- Bài 2: các khái niệm về lập trình hướng đối tượng
- Bài 3: Lớp và đối tượng trong Java
- Bài 4: Thừa kế
- Bài 5: Đa hình
- Bài 6: Lập trình tổng quát
- Bài 7: Cấu trúc dữ liệu
- Bài 8: Luồng dữ liệu
- Bài 9: Thiết kế hướng đối tượng

Trong quá trình biên soạn chúng tôi đã kết hợp kinh nghiệm giảng dạy trong nhiều năm của nhóm tác giả với mong muốn có thể giúp cho học sinh – sinh viên dễ dàng nắm bắt được nội dung của môn học. Mặc dù, rất cố gắng trong quá trình biên soạn, nhưng chắc chắn không thể tránh khỏi những thiếu sót, vì vậy, nhóm tác giả chúng tôi rất mong nhận được ý kiến đóng góp của bạn đọc để giáo trình ngày một hoàn thiện hơn.

Tp.Hồ Chí Minh, ngày 31 tháng 8 năm 2022

Tham gia biên soạn

1. Chủ biên: ThS Lê Thị Thùy Trang

2. ……

3. …….

MỤC LỤC

GIÁO TRÌNH MÔN HỌC

Tên môn học: Lập Trình Hướng Đối Tượng

Mã môn học: MH27

Thời gian thực hiện môn học: 45 giờ, (Lý thuyết: 15 giờ; Thực hành: 28, thí nghiệm, thảo luận, bài tập: 0 giờ; Kiểm tra: 2 giờ)

I. Vị trí, tính chất của môn học:

- Vị trí: môn học cơ sở cho ngành Công nghệ thông tin.
- Tính chất: là môn học bắt buộc.

II. Mục tiêu môn học:

1. Về kiến thức:

- Phát biểu được các nguyên lý cơ bản của thiết kế hướng đối tượng.
- Cài đặt được các lớp và phương thức bản chất của đối tượng, tham chiếu đến đối tượng.
- Trình bày và phân biệt được các quan niệm cây kế thừa, đa hình.
- Trình bày và phân biệt được các khái niệm căn bản về lập trình tổng quát, cấu trúc dữ liệu tổng quát.

2. Về kỹ năng:

- Giải quyết và đưa ra giải pháp lập trình hướng đối tượng cho các bài toán ở quy mô nhỏ.
- Trình bày được sơ đồ lớp bằng ngôn ngữ đặc tả UML với cú pháp cơ bản
- Cài đặt thiết kế hướng đối tượng cho trước bằng ngôn ngữ Java.

3. Về năng lực tự chủ và trách nhiệm:

- Học sinh có khả năng tự học các ngôn ngữ lập trình hướng đối tượng khác và tự tìm hiểu sử dụng các thư viện lập trình kèm theo.
- Có ý thức rèn luyện kỹ năng làm việc chính xác, cẩn thận và theo phong cách công nghiệp, hệ thống.

NỘI DUNG MÔN HỌC

Bài 1: Mở đầu

Mục tiêu:

Học xong chương này, người học có khả năng:

- Tìm hiểu lịch sử ngôn ngữ java
- Cài đặt môi trường java
- Biên dịch và chạy chương trình java

Nội dung:

1.1. Lịch sử ngôn ngữ Java

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak.

Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Java được tạo ra với tiêu chí "Viết (code) một lần, thực thi khắp nơi" (Write Once, Run Anywhere – WORA). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó

1.2. Môi trường lập trình Java, máy ảo, và JDK.

Để đảm bảo tính đa nền, Java sử dụng cơ chế Máy ảo của Java.

ByteCode là ngôn ngữ máy của Máy ảo Java tương tự như các lệnh nhị phân của các máy tính thực.

Một chương trình sau khi được viết bằng ngôn ngữ Java (có phần mở rộng là .java) phải được biên dịch thành tập tin thực thi được trên máy ảo Java (có phần mở rộng là .class). Tập tin thực thi này chứa các chỉ thị dưới dạng mã Bytecode mà máy ảo Java hiểu được phải làm gì.

Khi thực hiện một chương trình, máy ảo Java lần lượt thông dịch các chỉ thị dưới dạng Bytecode thành các chỉ thị dạng nhị phân của máy tính thực và thực thi thực sự chúng trên máy tính thực (còn gọi là khả năng khả chuyển).

Máy ảo thực tế đó là một chương trình thông dịch. Vì thế các hệ điều hành khác nhau sẽ có các máy ảo khác nhau. Để thực thi một ứng dụng của Java trên một hệ điều hành cụ thể, cần phải cài đặt máy ảo tương ứng cho hệ điều hành đó.

JVM cung cấp môi trường thực thi cho chương trình Java (còn gọi là khả năng độc lập với nền).

Có nhiều JVM cho các nền tảng khác nhau chẳng hạn như: Windows, Liux, và Mac.

1.3. Giới thiệu về các loại chương trình Java: application và applet.

Các chương trình xây dựng bằng Java được chia làm 2 loại là Java Applet và Java Application

Java Application là gì?

Ngôn ngữ Java có cú pháp khá giống với ngôn ngữ C/C++, nhưng nó có thể đáp ứng khả năng độc lập với hệ điều hành. Nền tảng của ngôn ngữ lập trình Java là các class, nó đóng vai trò như những đối tượng. Người lập trình khi xây dựng các ứng dụng sẽ sử dụng 1 số class chuẩn của hệ thống, hoặc cũng có thể tự mình xây dựng class khác nhằm đáp ứng yêu cầu công việc.

Hiện nay Java được ứng dụng rất rộng rãi cho các ứng dụng được viết trên Internet, lấy mạng làm trung tâm và có thể được dùng như một nền tảng. Đây là 1 loại ngôn ngữ lập trình nhanh, bảo mật, đáng tin cậy, được dùng để viết mã cho các ứng dụng di động, phần mềm doanh nghiệp hay các ứng dụng dữ liệu lớn và công nghệ phía máy chủ.

Java Application (viết tắt Java App) là những ứng dụng độc lập, tương tự như các chương trình có đuôi .COM hay đuôi .EXE thông thường. Việc thực hiện Java Application đơn giản hơn Java Applet bởi chúng không cần phải thông qua trình duyệt Web.

Applet là gì?

Ngôn ngữ lập trình Java hiện nay được sử dụng khá phổ biến trên nhiều hệ điều hành khác nhau ở khắp nơi trên thế giới. Chính vì vậy, ngôn ngữ lập trình Java cũng được sử dụng để tạo ra nhiều loại ứng dụng đa dạng.

Trong đó, Applet là những loại ứng dụng nhỏ được xây dựng bởi ngôn ngữ lập trình Java hoặc ngôn ngữ lập trình khác được biên dịch sang mã bytecode của ngôn ngữ Java. Tuy nhiên, khi đưa đến người dùng, Applet sẽ xuất hiện hoàn toàn với dạng Java bytecode.

Những phiên bản Applet đầu tiên được ra mắt vào năm 1995. Tuy nhiên, sau nhiều hoạt động điều chỉnh, các applet của Java đã được loại bỏ hoàn toàn vào 2017.

Sở hữu khả năng linh hoạt trên đa nền tảng, Applet có thể sử dụng trên nhiều hệ điều hành khác nhau như Microsoft Windows, Linux, MacOS, FreeBSD, Unix, v.v.

1.4. Dịch và chạy chương trình

Chạy chương trình bằng các lệnh trong Command Prompt với JDK đã cài đặt. Đầu tiên, thay đổi folder đang thực thi trong Command Prompt đến thư mục chứa file code Java.

```
C:\Users\LHVINH>E: E:\>cd Demo
```

Sau đó, sử dụng chương trình javac trong JDK để biên dịch code Java thành bytecode. Một file tên là Welcome.class chứa bytecode sẽ được tự động tạo ra.

```
E:\Demo>javac Welcome.java
```

Cuối cùng, thông dịch bytecode thành mã máy để CPU thực thi với chương trình java trong JDK

```
E:\Demo>java Welcome Welcome to Java!
```

Các Java IDE thường dùng:

IDE (Integrated Development Environment) là một loại phần mềm máy tính có chức năng giúp đỡ các lập trình viên trong việc phát triển phần mềm. Một IDE gồm có các thành phần chính:

- Trình soạn thảo mã nguồn (source code editor): dùng để viết code
- Trình biên dịch (compiler), trình thông dịch (interpreter)
- Công cụ xây dựng tự động: khi sử dụng sẽ biên dịch (hoặc thông dịch) mã nguồn, thực hiện liên kết các thư viện và có thể chạy chương trình một cách tự động
- Trình gỡ lỗi (debugger): hỗ trợ tìm lỗi code

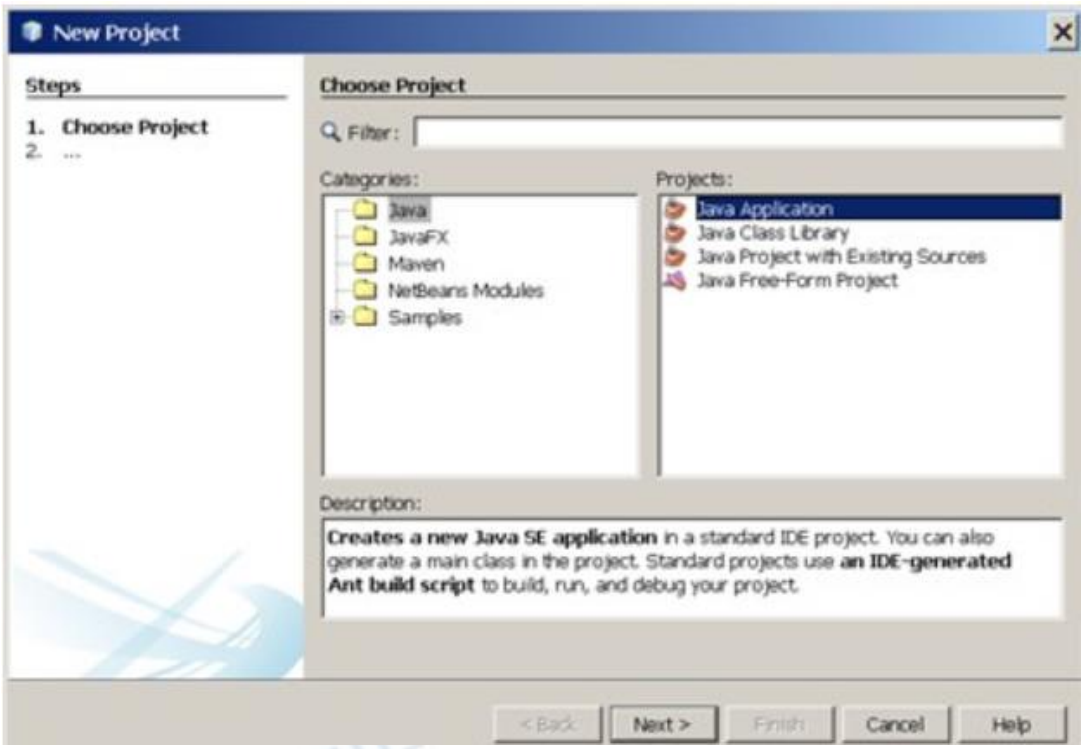
Các Java IDE thường dùng là NetBeans, Eclipse, IntelliJ IDEA,…

NetBeans IDE:

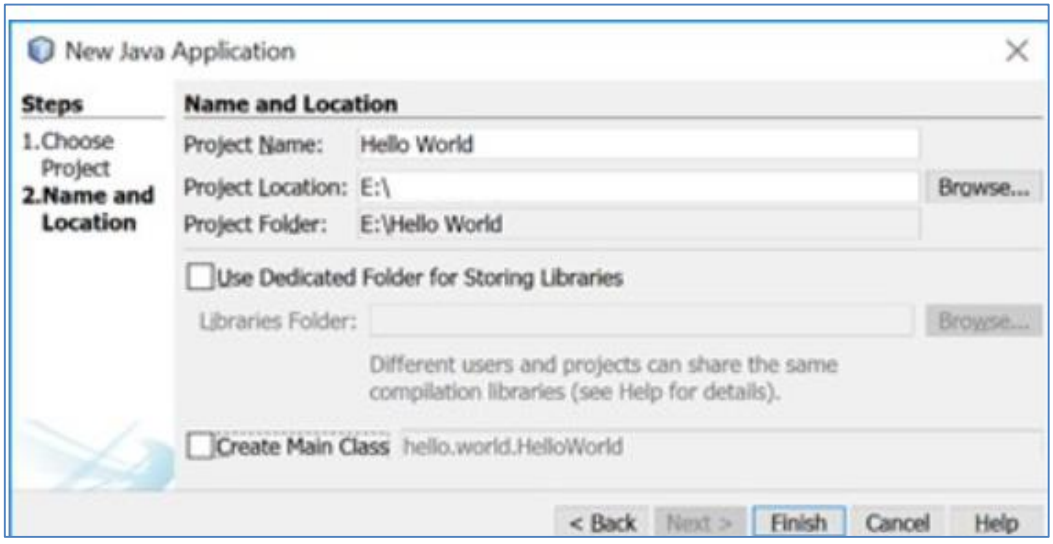
Tạo một Java Project

Trước khi tạo một chương trình Java trong NetBeans, cần phải tạo một project.

Chọn File –> New Project, hộp thoại New Project xuất hiện



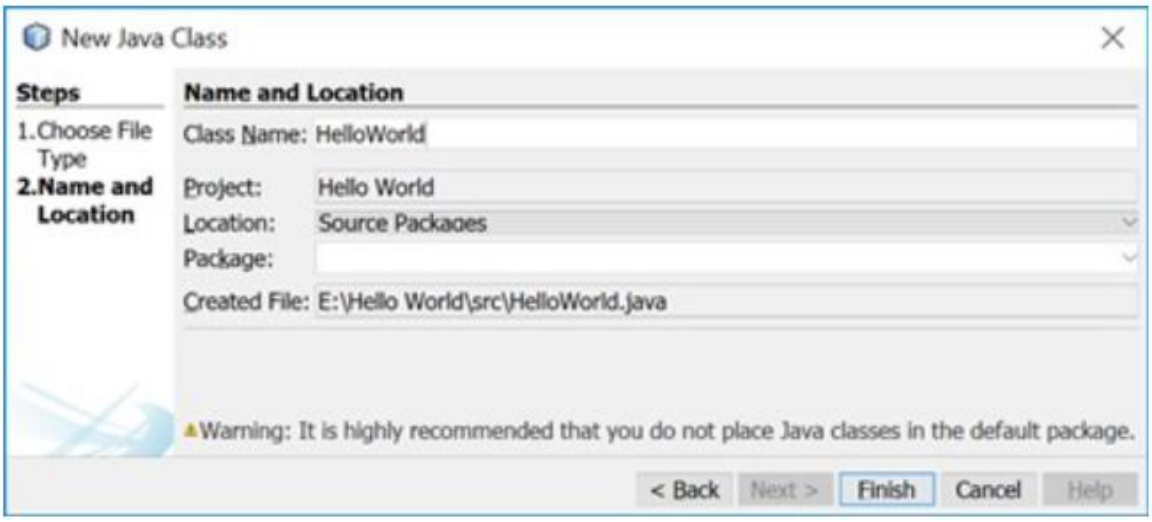
Chọn Java trong Categories và Java Application trong Projects và click Next để hiển thị hộp thoại New Java Application



Nhập tên (ví dụ: Hello World) trong mục Project Name và E:\ trong mục Project Location. Click Finish để tạo project

Tạo một Java Class

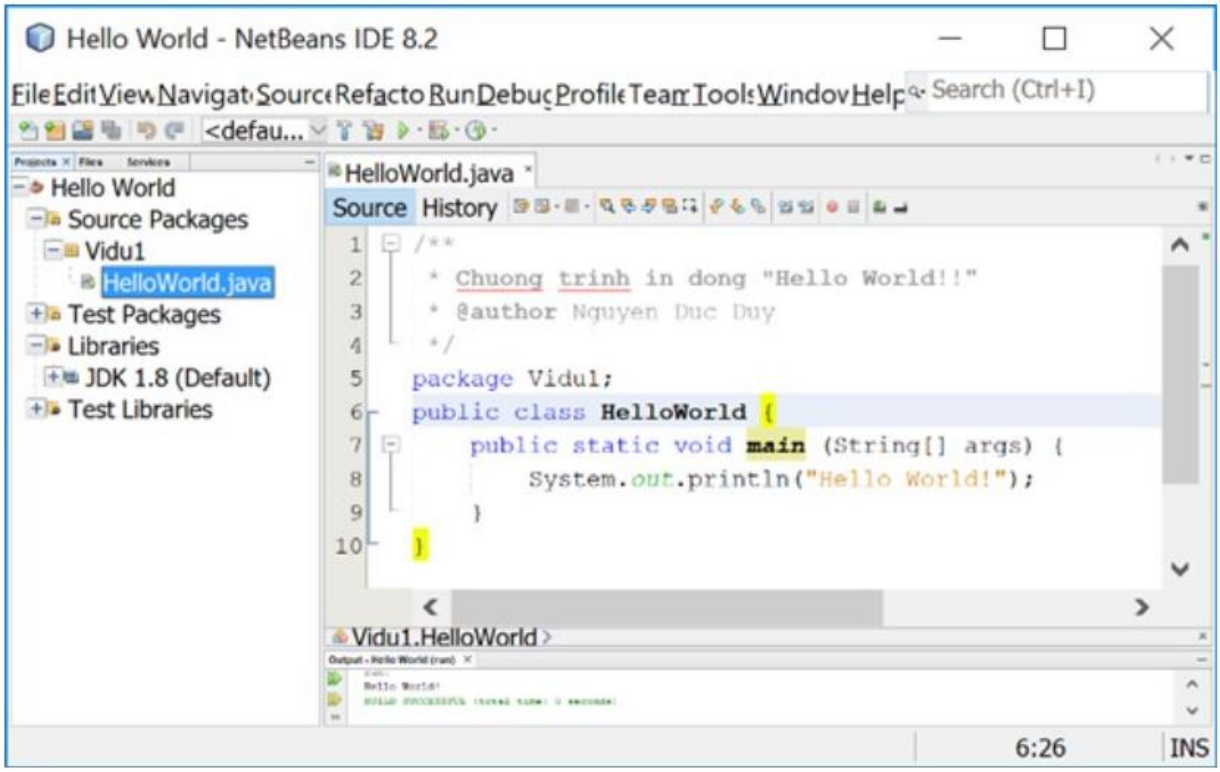
Right click vào Hello World trong Project Pane để hiển thị context menu. Chọn New –> Java Class để hiển thị hộp thoại New Java Class



Nhập HelloWorld trong mục Class Name và chọn Source Packages trong mục Location. Để trống mục Package, sẽ tạo một lớp trong default package. Click Finish để tạo HelloWorld class.

Bên dưới, tập tin HelloWorld.java được đặt trong <package Vidu1>. Soạn thảo code trong HelloWorld class.

Để chạy Welcome.java, right click Welcome.java để hiển thị context menu và chọn Run File hoặc nhấn Shift + F6.



Bài 2: Các khái niệm về lập trình hướng đối tượng

Mục tiêu:

Học xong chương này, người học có khả năng:

- Phân biệt lập trình hướng thủ tục và hướng đối tượng.

Nội dung:

2.1. Lịch sử ngôn ngữ lập trình.

Chương trình máy tính đầu tiên (1942-1949): Ada Lovelace đã dịch hồi ký của nhà toán học Ý Luigi Menabrea về cỗ máy mới nhất được đề xuất của Charles Babbage: Máy phân tích; bà đã bổ sung vào cuốn hồi ký những ghi chú chỉ ra chi tiết phương pháp tính toán số Bernoulli bằng cỗ máy đó, được hầu hết các nhà sử học công nhận là chương trình máy tính đầu tiên trên thế giới.

Máy tính điện tử đầu tiên (1940s): Lần đầu tiên ghi nhận Máy tính điện tử hiện đại được tạo. Do giới hạn về học tốc độ và bộ nhớ nên các lập trình viên phải viết các mã tay các chương trình hợp ngữ

Đầu những năm 1950 - Ngôn ngữ lập trình bậc cao đâu tiên: John Mauchly tạo ra Short Code, ngôn ngữ lập trình bậc cao đầu tiên, mỗi lần chạy cần được dịch thành mã máy nên khá là chậm

Cuối những năm 1960 – OOP: Simula, được phát minh bởi Nygaard và Dahl, là ngôn ngữ đầu tiên được thiết kế để hỗ trợ lập trình hướng đối tượng

1970 – Pascal: Niklaus Wirth tạo ra Pascal cùng với một số ngôn ngữ khác, ông thích tạo ra các ngôn ngữ. Ông cũng phát minh ra luật của Wirth cái làm cho luật của Moore thành lỗi thời bởi vì các lập trình viên phần mềm sẽ viết các phần mềm cồng kềnh đến nỗi ngay cả các máy tính lớn (mainframe) cũng không thể theo kịp. Điều này sau đó sẽ được chứng minh là đúng với phát minh của Electron.js.

1972 – C: Dennis Ritchie và Ken Thompson quyết định tạo ra C ngôn ngữ có những dấu ngoặc nhọn {} và nó là một thành công lớn. Sau đó ông bổ sung thêm secmentation faults và các tính năng thân thiện với các lập trình viên để tăng hiệu xuất.

Ông và bạn bè của mình tại Bell Labs cũng làm một chương trình demo cho C, kết quả họ đã tạo ra một hệ điều hành gọi là Unix.

1972 - Prolog, ngôn ngữ logic đầu tiên: Prolog, ngôn ngữ lập trình logic đầu tiên, được thiết kế bởi Colmerauer, Roussel, và Kowalsk

1978 – SQL: SQL, ngôn ngữ truy vấn có cấu trúc, được tạo bởi Donald D. Chamberlin và Raymond F. Boyce ở IBM

1983 - C++: Bjarne Stroustrup bổ sung thêm nhiều tính năng nâng cao mà ông có thể nghĩ ra cho C và đặt cho ngôn ngữ lập trình này là C++.

1986 - Objective-C: Brac Box và Tol Move quyết định tạo thêm một phiên bản C dựa trên Smalltalk, cái được gọi là Objective-C.

1987 – Perl: Larry Wall với kinh nghiệm tôn giáo, trở thành một nhà thuyết giảng và làm cho Perl trở thành một học thuyết.

1991 – Python: Guido van Rossum phát minh ra Python, cú pháp được chọn lấy cảm hứng từ Monty Python và Flying Circus.

1991 – Visual Basic: Được tạo ra bởi Microsoft, bản release cuối cùng là version 6 vào năm 1998, window 10 vẫn duy trì khả năng tương thích với các ứng dụng Visual Basic 6

1995 – JavaScipt: Brendan Eich sử dụng những ngày cuối tuần để thiết kế một ngôn ngữ sẽ được sử dụng rộng rãi trên mọi trình duyệt và cuối cùng là Skynet. Ban đầu khi làm việc tại Netscape ông gọi nó là LiveScript nhưng Java trở thành một một ngôn ngữ lập trình phổ biến vì thế họ đã quyết định đổi tên nó thành JavaScipt.

Java trở thành một thương hiệu và điều này đem đến rắc rối cho họ, JavaScript sau đó được đổi tên thành ECMAScript nhưng mọi người vẫn gọi nó là JavaScript.

1996 – Java: James Gosling phát minh Java, ngôn ngữ đầu tiên thực sự hướng đối tượng nơi các quy tắc design patterns được chú ý hơn chủ nghĩa thực dụng.

2001 - C#: [Anders Hejlsberg] viết lại Java và gọi nó là C# bởi vì lập trình trong C tuyệt hơn Java. Mọi người thích phiên bản Java mới này vì nó hoàn toàn không giống Java.

2.2. So sánh lập trình thủ tục và lập trình hướng đối tượng.

Ưu điểm của các kiểu lập trình.

POP (Lập trình hướng thủ tục)

- Cung cấp khả năng tái sử dụng cùng một mã tại nhiều nơi khác nhau.
- Tạo điều kiện trong việc theo dõi dòng chương trình.
- Có khả năng xây dựng các mô-đun.

OOP (Lập trình hướng đối tượng)

- Các đối tượng giúp phân vùng nhiệm vụ trong dự án.
- Chương trình an toàn có thể được xây dựng bằng cách ẩn dữ liệu.
- Nó có khả năng lập biểu đồ cho các đối tượng.

- Cho phép phân loại các đối tượng thành các lớp khác nhau.
- Hệ thống hướng đối tượng có thể được nâng cấp dễ dàng.
- Mã dự phòng có thể được loại bỏ bằng cách sử dụng kế thừa.
- Mã có thể được mở rộng bằng cách sử dụng lại.
- Mô-đun lớn hơn có thể đạt được.
- Trừu tượng dữ liệu làm tăng độ tin cậy.
- Linh hoạt do các khái niệm ràng buộc năng động.
- Tách riêng các đặc điểm kỹ thuật cần thiết từ việc thực hiện bằng cách sử dụng ẩn thông tin.

Sự khác biệt chính giữa OOP và POP:

- POP là lập trình hướng thủ tục trong khi OOP là lập trình hướng đối tượng.
- Trọng tâm chính của POP là về cách thức thực hiện nhiệm vụ của hệ thống, nó tuân theo biểu đồ dòng chảy để hoàn thành nhiệm vụ. Trọng tâm chính của OOP là bảo mật dữ liệu vì chỉ các đối tượng của một lớp mới được phép truy cập các thuộc tính hoặc chức năng của một lớp.
- Các chức năng là các đơn vị nhỏ của các chương trình lớn hoặc một chương trình con thực thi để hoàn thành nhiệm vụ chính. Ngược lại, các thuộc tính và hàm OOP của lớp được chia cho các đối tượng .
- Trong POP, không có chế độ truy cập cụ thể để truy cập các thuộc tính hoặc chức năng trong chương trình. Ngược lại, trong OOP có ba chế độ truy cập, "public", "private", "protected" được sử dụng như một phương thức truy cập để truy cập các thuộc tính hoặc chức năng.
- POP không hỗ trợ khái niệm Overloading/polymorphism. Ngược lại, OOP hỗ trợ Quá tải / Đa hình, nghĩa là sử dụng cùng tên hàm để thực hiện các chức năng khác nhau. Chúng ta có thể Overload các hàm, hàm tạo và toán tử trong OOP.
- Không có khái niệm thừa kế trong POP trong khi đó, OOP hỗ trợ kế thừa cho phép sử dụng thuộc tính và chức năng của lớp khác bằng cách kế thừa nó.
- POP kém an toàn hơn so với OOP vì trong OOP, bộ chỉ định truy cập giới hạn quyền truy cập vào các thuộc tính hoặc chức năng làm tăng tính bảo mật.
- Trong POP nếu một số dữ liệu được chia sẻ giữa tất cả các chức năng trong chương trình, nó được khai báo trên toàn hệ thống bên ngoài tất cả các chức năng. Trong khi

trong OOP, data-member của lớp có thể được truy cập thông qua các member-function của lớp.

- Không có khái niệm về các lớp ảo trong POP trong khi trong OOP, các hàm ảo hỗ trợ đa hình.

2.3. Khái niệm hướng đối tượng và mục tiêu của thiết kế hướng đối tượng.

Lập trình hướng đối tượng (OOP) là một kỹ thuật lập trình cho phép lập trình viên tạo ra các đối tượng trong code trừu tượng hóa các đối tượng.

Đối tượng:

Một đối tượng bao gồm 2 thông tin: thuộc tính và phương thức.

- Thuộc tính chính là những thông tin, đặc điểm của đối tượng. Ví dụ: con người có các đặc tính như mắt, mũi, tay, chân…
- Phương thức là những thao tác, hành động mà đối tượng đó có thể thực hiện. Ví dụ: một người sẽ có thể thực hiện hành động nói, đi, ăn, uống, . . .

Lớp:

Một lớp là một kiểu dữ liệu bao gồm các thuộc tính và các phương thức được định nghĩa từ trước. Đây là sự trừu tượng hóa của đối tượng. Khác với kiểu dữ liệu thông thường, một lớp là một đơn vị (trừu tượng) bao gồm sự kết hợp giữa các phương thức và các thuộc tính. Hiểu nôm na hơn là các đối tượng có các đặc tính tương tự nhau được gom lại thành một lớp đối tượng.

Các nguyên lý cơ bản của OOP:

- Tính đóng gói (Encapsulation)
- Tính kế thừa (Inheritance)
- Tính đa hình (Polymorphism)
- Tính trừu tượng (Abstraction)

Các ưu điểm của lập trình hướng đối tượng:

- Dựa trên nguyên lý kế thừa, trong quá trình mô tả các lớp có thể loại bỏ những chương trình bị lặp, dư. Và có thể mở rộng khả năng sử dụng các lớp mà không cần thực hiện lại. Tối ưu và tái sử dụng code hiệu quả.
- Đảm bảo rút ngắn thời gian xây dựng hệ thống và tăng năng suất thực hiện.
- Sự xuất hiện của 2 khái niệm mới là lớp và đối tượng chính là đặc trưng của phương pháp lập trình hướng đối tượng. Nó đã giải quyết được các khuyết điểm của phương

pháp lập trình hướng cấu trúc để lại. Ngoài ra 2 khái niệm này đã giúp biểu diễn tốt hơn thế giới thực trên máy tính.

2.4. Sơ lược về các khái niệm cơ bản: trừu tượng hóa, đối tượng, lớp, thông điệp, đóng gói, che dấu thông tin, thừa kế.

Tính đóng gói (Encapsulation):

Các dữ liệu và phương thức có liên quan với nhau được đóng gói thành các lớp để tiện cho việc quản lý và sử dụng. Tức là mỗi lớp được xây dựng để thực hiện một nhóm chức năng đặc trưng của riêng lớp đó.

Ngoài ra, đóng gói còn để che giấu một số thông tin và chi tiết cài đặt nội bộ để bên ngoài không thể nhìn thấy.

Tính kế thừa (Inheritance):

Nó cho phép xây dựng một lớp mới dựa trên các định nghĩa của lớp đã có. Có nghĩa là lớp cha có thể chia sẽ dữ liệu và phương thức cho các lớp con. Các lớp con khỏi phải định nghĩa lại, ngoài ra có thể mở rộng các thành phần kế thừa và bổ sung thêm các thành phần mới. Tái sử dụng mã nguồn 1 cách tối ưu, tận dụng được mã nguồn. Một số loại kế loại kế thừa thường gặp: đơn kế thừa, đa kế thừa, kế thừa đa cấp, kế thừa thứ bậc.

Khi bắt đầu xây dựng ứng dụng chúng ta sẽ bắt đầu thiết kế định nghĩa các lớp trước. Thông thường một số lớp có quan hệ với những lớp khác, chúng có những đặc tính giống nhau.

VD: 2 lớp Android, iPhone

Mỗi lớp đều đại diện cho một loại smartphone khác nhau nhưng lại có những thuộc tính giống nhau như gọi điện, nhắn tin, chụp hình. Thay vì sao chép những thuộc tính này, ta nên đặt chúng vào một lớp chung gọi là lớp cha. Chúng ta có thể định nghĩa lớp cha – trong trường hợp này l

Tính đa hình (Polymorphism):

Tính đa hình là một hành động có thể được thực hiện bằng nhiều cách khác nhau. Đây lại là một tính chất có thể nói là chứa đựng hầu hết sức mạnh của lập trình hướng đối tượng.

Hiểu một cách đơn giản hơn: Đa hình là khái niệm mà hai hoặc nhiều lớp có những phương thức giống nhau nhưng có thể thực thi theo những cách thức khác nhau.

Ví dụ như ở phần trên, mỗi một smartphone kế thừa từ lớp cha Smartphone nhưng có thể lưu trữ dữ liệu trên cloud theo những cách khác nhau. Android lưu trữ bằng Google Drive iPhone lưu trên iCloud.

Một ví dụ về đa hình trong thực tế. Ta có 2 con vật: chó, mèo. Cả 2 con vật này đều là lớp động vật. Nhưng khi ta bảo cả 2 động vật kêu thì con chó sẽ kêu gâu gâu, con mèo sẽ kêu meo meo.

Vậy trong ví dụ chó, mèo xem như là các đối tượng. 2 con vật có thể hiểu cùng kêu nhưng theo các cách khác nhau. à Smartphone và có những lớp con kế thừa từ nó, tạo ra một mối quan hệ cha/con.

Tính trừu tượng (Abstraction):

Trừu tượng có nghĩ là tổng quát hóa một cái gì đó lên, không cần chú ý chi tiết bên trong. Nó không màng đến chi tiết bên trong là gì và người ta vẫn hiểu nó mỗi khi nghe về nó.

Ví dụ: Bạn chạy xe tay ga thì có hành động là tăng ga để tăng tốc, thì chức năng tăng ga là đại diện cho trừu tượng (abstraction). Người dùng chỉ cần biết là tăng ga thì xe tăng tốc, không cần biết bên trong nó làm thế nào.

Ở đây trong lập trình OOP, tính trừu tượng nghĩa là chọn ra các thuộc tính, phương thức của đối tượng cần cho việc giải quyết bài toán đang lập trình. Vì một đối tượng có rất nhiều thuộc tính phương thức, nhưng với bài toán cụ thể không nhất thiết phải chọn tất cả.

BÀI TẬP

Quản lý thông tin về sách trong một thư viện. Sử dụng lập trình hướng đối tượng trong Java, thiết kế các lớp để thực hiện các chức năng sau:

1. Định nghĩa lớp Sach (Book):

- Các thuộc tính bao gồm: mã sách, tựa đề, tác giả, năm xuất bản và số lượng trong kho.
- Xác định các phương thức để truy cập và cập nhật thông tin của sách.

2. Định nghĩa lớp ThuVien (Library):

- Sử dụng một ArrayList để lưu trữ danh sách các sách trong thư viện.
- Các phương thức bao gồm: thêm sách mới, xóa sách, tìm kiếm sách theo mã sách hoặc tựa đề, và hiển thị danh sách sách có trong thư viện.

Hướng dẫn:

1. Tạo lớp Sach (Book) với các thuộc tính và phương thức phù hợp như đã mô tả trong bài tập.
2. Tạo lớp ThuVien (Library) với một ArrayList để lưu trữ danh sách các sách.

- Viết phương thức để thêm sách mới vào danh sách.

- Viết phương thức để xóa sách khỏi danh sách.
- Viết phương thức để tìm kiếm sách theo mã sách hoặc tựa đề.
- Viết phương thức để hiển thị danh sách sách có trong thư viện.

3. Tạo lớp chương trình chính để thử nghiệm các chức năng của lớp ThuVien (Library).

Tham khảo:

```
import java.util.ArrayList;
class Sach {
    private String maSach;
    private String tuaDe;
    private String tacGia;
    private int namXuatBan;
    private int soLuong;
    public Sach(String maSach, String tuaDe, String tacGia, int
namXuatBan, int soLuong) {
        this.maSach = maSach;
        this.tuaDe = tuaDe;
        this.tacGia = tacGia;
        this.namXuatBan = namXuatBan;
        this.soLuong = soLuong;
    }
    // Getters và setters
    // ...
    @Override
    public String toString() {
        return "Mã sách: " + maSach + ", Tựa đề: " + tuaDe + ",
Tác giả: " + tacGia + ", Năm xuất bản: " + namXuatBan + ", Số
lượng: " + soLuong;
    }
}
class ThuVien {
    private ArrayList<Sach> danhSachSach = new ArrayList<>();

    public void themSach(Sach sach) {
        danhSachSach.add(sach);
    }

    public void xoaSach(String maSach) {
        for (Sach sach : danhSachSach) {
            if (sach.getMaSach().equals(maSach)) {
                danhSachSach.remove(sach);
                break;
            }
        }
    }
    public Sach timKiemSachTheoMa(String maSach) {
        for (Sach sach : danhSachSach) {
            if (sach.getMaSach().equals(maSach)) {
                return sach;
            }
```

```
        }
        return null;
    }
    public Sach timKiemSachTheoTuaDe(String tuaDe) {
        for (Sach sach : danhSachSach) {
            if (sach.getTuaDe().equals(tuaDe)) {
                return sach;
            }
        }
        return null;
    }

    public void hienThiDanhSachSach() {
        for (Sach sach : danhSachSach) {
            System.out.println(sach);
        }
    }
}
public class Main {
    public static void main(String[] args) {
        ThuVien thuVien = new ThuVien();

        // Thêm sách mới
        thuVien.themSach(new Sach("MS001", "Java Programming",
"John Doe", 2020, 5));
        thuVien.themSach(new Sach("MS002", "Python Programming",
"Jane Smith", 2019, 3));

        // Hiển thị danh sách sách trong thư viện
        System.out.println("Danh sách sách trong thư viện:");
        thuVien.hienThiDanhSachSach();

        // Tìm kiếm sách theo mã sách
        String maSach = "MS001";
        System.out.println("\nTìm kiếm sách có mã " + maSach +
":");
        Sach sach1 = thuVien.timKiemSachTheoMa(maSach);
        if (sach1 != null) {
            System.out.println(sach1);
        } else {
            System.out.println("Không tìm thấy sách có mã " +
maSach);
        }

        // Xóa sách khỏi thư viện
        thuVien.xoaSach("MS002");
        System.out.println("\nDanh sách sách sau khi xóa:");
        thuVien.hienThiDanhSachSach();
    }
}
```

Bài 3: Lớp và đối tượng trong Java

Mục tiêu:

Học xong chương này, người học có khả năng:

- Định nghĩa được lớp
- Tạo được đối tượng
- Phân biệt các kiểu cơ sở và tham chiếu
- Sử dụng thành thạo các phép gán, phép so sánh.

Nội dung:

3.1. Vị trí của đối tượng trong chương trình

Đối tượng (Object) có thể là con người, điện thoại, máy tính, yasuo, leesin ... và điểm chung là đều gồm 2 thành phần chính là:

- Thuộc tính (Attribute): là những thông tin, đặc điểm của đối tượng.
- Phương thức (Method): là những hành động mà đối tượng có thể thực hiện.

Ví dụ: máy tính sẽ có các thuộc tính như: màu sắc, kích thước, bộ nhớ, ...

Và các phương thức như: quét virus, tắt máy, khởi động máy, mở LoL ...

Khi một đối tượng có những đặc tính như nhau sẽ được gom lại thành một lớp đối tượng (class) và cũng sẽ có 2 thành phần là thuộc tính và phương thức. Lớp (class) cũng có thể được dùng để định nghĩa một kiểu dữ liệu mới.

Với ví dụ máy vi tính trên ta hiểu lớp (class) máy vi tính có:

Các thuộc tính như: màu sắc, kích thước, bộ nhớ, ...

Và các phương thức như: quét virus, tắt máy, khởi động máy, mở LoL ...

Và đối tượng (object) có thể là các dòng máy như Asus, Acer, Lenovo, Thinkpad, ... đều mang đặc tính của lớp máy vi tính.

3.2. Cài đặt và sử dụng lớp và đối tượng.

Lớp:

Một lớp là một nhóm đối tượng có các thuộc tính chung. Nó là một mẫu hoặc thiết kế từ đó các đối tượng được tạo ra.

Một lớp trong java có thể chứa:

- Thành viên dữ liệu
- Constructor
- Phương thức