รายงาน

แนวคิดและการเขียนโปรแกรมเชิงวัตถุ

โดย

593021504-0 นายเชาวรินทร์ ตู้จันโต

อาจารย์ผู้สอน : อ.ดร.นันท์นภัส เบญจมาศ

อ.ดร.สายยัญ สายยศ

รายงานนี้เป็นส่วนหนึ่งของวิชา 342118 แนวคิดและการเขียนโปรแกรมเชิงวัตถุ

ภาคเรียนที่ 2 ปีการศึกษา 2559

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น

(เดือน กุมภาพันธ์ พ.ศ. 2560)

คำนำ

รายงานนี้เป็นส่วนหนึ่งของวิชา 342118 แนวคิดและการเขียนโปรแกรมเชิงวัตถุ จัดหาขึ้นเพื่อศึกษา เกี่ยวกับความหมาย ความการเขียนโปรแกรมเชิงวัตถุ เปรียบเทียบแนวคิดระหว่างการเขียนโปรแกรมเชิงกระบวนการ และเชิงวัตถุ โดยใช้ภาษา จาวา(Java) โดยนำหลักการต่าง ๆเช่น Information Hiding , Encapsulation , Inheritance , Polymorphism และ Composition โดยนำหลักการดังกล่าวมาสร้างเป็นโปรแกรมหนึ่งโปรแกรมที่ได้จัดทำขึ้นมา ทำให้รู้จักและการนำหลักการดังกล่าวนำไปใช้ และเป็นการฝึกฝนเพื่อให้ทักษะการเขียนโปรแกรมได้ดียิ่งขึ้น

ได้หวังว่ารายงานฉบับนี้อาจจะเป็นที่สนใจกับผู้อ่านผู้ศึกษาได้และในที่นี่ขอขอบคุณผู้อ่านผู้ศึกษาหากมีขอสังเกตข้อสงสัยในรายงานฉบับนี้โปรดติดตามรายงานฉบับถัดไปซึ่งอาจจะเป็นรายงานวิชาอื่น ๆเป็นต้นไป

เชาวรินทร์ ตู้จันโต

ผู้จัดทำ

28 กุมภาพันธ์ พ.ศ. 2560

สารบัญ

หัวข้อ **หน้า**

คำนำ ก

1. การเขียนโปรแกรมเชิงวัตถุ 1

2.เปรียบเทียบแนวคิดระหว่างการเขียนโปรแกรมเชิงกระบวนการ และเชิงวัตถุ 1

3. 1.) คุณสมบัติของวัตถุ **3**

1.1) Information hiding 3

1.2) การห่อหุ้ม (Encapsulation) 3

1.3) การสืบทอด (Inheritance) 4

1.4) การมีได้หลายรูปแบบ (Polymorphism) 4

1.5) Composition 5

4. 2.) Source code ภาษา Java **6**

2.1) Class Diagram 6

2.2) Source Code 7

Demostic Main class 7

Details Sub class 8

PrintBoardingPass Class 11

Tested Demostic Class (Output) 14 14

5.เอกสารอ้างอิง 15

6.ประวัติผู้จัดทำ 16

การเขียนโปรแกรมเชิงวัตถุ

การเขียนโปรแกรมเชิงวัตถุ (Object-oriented programming, OOP) คือหนึ่งในรูปแบบการเขียนโปรแกรมคอมพิวเตอร์ ที่ให้ความสำคัญกับ วัตถุ ซึ่งสามารถนำมาประกอบกันและนำมาทำงานรวมกันได้ โดยการแลกเปลี่ยนข่าวสารเพื่อนำมาประมวลผลและส่งข่าวสารที่ได้ไปให้ วัตถุ อื่น ๆที่เกี่ยวข้องเพื่อให้ทำงานต่อไป

แนวคิดการเขียนโปรแกรมแบบดังเดิมมักนิยมใช้ การเขียนโปรแกรมเชิงกระบวนการ (Procedural Programming) ซึ่งให้ความสำคัญกับขั้นตอนกระบวนการที่ทำ โดยแบ่งโปรแกรมออกเป็นส่วนๆตามลำดับขั้นตอนการทำงาน แต่แนวคิดการเขียนโปรแกรมเชิงวัตถุนั้นให้ความสำคัญกับ ข้อมูล(data) และ พฤติกรรม(behavior) ของวัตถุ และความสัมพันธ์กันระหว่างวัตถุกันมากกว่า

เปรียบเทียบแนวคิดระหว่างการเขียนโปรแกรมเชิงกระบวนการ และเชิงวัตถุ

วิธีการคิดแบบการเขียนโปรแกรมเชิงกระบวนการ

เมื่อมีการหยอดเหรียญเข้าตู้

ตรวจสอบจำนวนเหรียญและชนิดของเหรียญ

แสดงผลชนิดของน้ำที่สามารถเลือกซื้อได้

ตรวจสอบจำนวนน้ำกระป๋องที่มีอยู่ในตู้

รับผลการเลือกชนิดน้ำ

ส่งน้ำที่เลือกออกมาจากช่อง

จัดเก็บเงินเข้าระบบ

หากมีเงินทอน ให้ทอนเงินที่เหลือ ที่ช่องรับเงินทอน

วิธีการคิดแบบการเขียนโปรแกรมเชิงวัตถุ

ตู้ขายเครื่องดื่มอัตโนมัติ ประกอบด้วยส่วนประกอบต่าง ๆได้แก่ หน่วยตรวจสอบและจัดการเรื่องเงิน หน่วยจัดการเครื่องดื่ม หน่วยแสดงผลและรอรับคำสั่ง

- หน่วยตรวจสอบและจัดการเรื่องเงิน มีข้อมูลเกี่ยวกับเงินที่ได้รับ และเงินที่มีอยู่ในระบบ สามารถรับและตรวจสอบเงินที่หยอดเข้ามาได้ และทอนเงินได้

- หน่วยจัดการเครื่องดื่ม มีข้อมูลชนิดของเครื่องดื่ม จำนวนเครื่องดื่ม สามารถจัดเตรียมชนิดเครื่องดื่มที่พอกับเงินที่หยอด และสามารถจ่ายเครื่องดื่มออกมาจากตู้ได้

- หน่วยแสดงผลและรอรับคำสั่ง มีหน้าที่รอรับคำสั่ง และแสดงผลเงินที่หยอดเข้ามา

หมายเหตุ ตัวอย่างนี้เป็นเพียงตัวอย่างโดยสังเขป

1.คุณสมบัติของวัตถุ

1.1) Information hiding

เป็นกระบวนการซ่อนรายละเอียดการทำงานและข้อมูลไว้ภายใน ไม่ให้ภายนอกสามารถมองเห็นได้ หรือที่เรียกว่า "Information hiding" และเมื่อภายนอกมองไม่เห็นสิ่งที่ถูกซ่อนไว้ภายในแล้ว ก็จะไม่สามารถทำการเปลี่ยนแปลง แก้ไข สิ่ง ต่าง ๆที่อยู่ภายในได้

Example :
```
private String arrival;  //Information hiding
private String departures;  //Information hiding
private String airlines;  //Information hiding
private String flight;  //Information hiding
```

1.2) การห่อหุ้ม (Encapsulation)

หมายถึงการจะเรียกใช้คุณลักษณะของออปเจ็ค จะทำได้โดยการเรียกผ่านเมธอดเท่านั้น หลักการของการห่อหุ้ม คือการกำหนดให้คุณลักษณะของออปเจ็คมีคุณสมบัติเป็น private และกำหนดให้เมธอดมีคุณสมบัติเป็น public โดยมีเมธอด get / set ไว้เพื่อเข้าถึง data นั้น ๆ จะเรียกว่า class นั้น ๆ เป็น Full Encapsulation class

ข้อดีของการห่อหุ้ม (Encapsulation)

– Flexibility

มีความยืดหยุ่นสูง การใช้งาน method ใน OOP นั้นมีความยืดหยุ่นสูงกว่าการใช้งาน Data Fields ใน Object โดยตรงๆ

– Maintainability

การดูแลรักษาง่าย เพราะหากมีการเปลี่ยนแปลงภายใน Object ใด ๆ หากว่าส่วนติดต่อกับ Object อื่นยังคงเดิมแล้ว ก็ย่อมไม่ส่งผลกระทบต่องานทั้งระบบ

– Information Hiding

การซ่อนเร้นข้อมูลทำให้ออปเจ็คสามารถติดต่อกับออปเจ็คภายนอกผ่านเมธอดที่เป็นส่วนของ interface เท่านั้นRE

Example :
```
public String getFlight(){return flight;}
public void setFlight(String flight){this.flight=flight;}
```

1.3) การสืบทอด (Inheritance)

หมายถึงการนิยามคลาสใหม่จากคลาสที่มีอยู่แล้วโดยคลาสใหม่สามารถที่จะนำคุณลักษณะและเมธอดของคลาสเดิมมาใช้ได้

คือ Class หนึ่งๆสามารถสืบทอดคุณสมบัติบางประการจาก Class อื่น แล้วเพิ่มคุณสมบัติเฉพาะของ Class นั้นเข้าไป

– Class ที่ได้รับการสืบทอดคุณสมบัติเรียกว่า Subclasses

– Class ที่เป็นต้นแบบเรียกว่า Superclass

เป็นการช่วยให้ไม่ต้องพัฒนา ส่วนที่ซ้ำหลายๆรอบ (Reusable)

Class หนึ่งๆจะมี Superclass ได้เพียง Class เดียวเท่านั้น (Single Inheritance)

ในภาษาจาวา จะใช้คีย์เวิร์ด extends เพื่อระบุการสืบทอด

Example :

```
class Details extends Demostic { //Sub class

    //Constructors and Inheritance
    public Details(String arrival, String departures, String airlines, String flight, String name, String seat, String gates) {
    super(arrival, departures, airlines, flight);
    this.name=name;
    this.seat=seat;
    this.gates=gates;
    }
```

1.4) การมีได้หลายรูปแบบ (Polymorphism)

หมายถึง การที่สามารถตอบสนองต่อข่าวสาร (เมธอด) เดียวกันด้วยวิธีการที่ต่างกัน และสามารถกำหนดออปเจ็คได้หลายรูปแบบ

Example :

```
public class PrintBoardingPass {

    static void printde(Demostic a) {//Polymorphism Method
    System.out.println(a.BoardingPass());
    }
    static void printda(Dateandmin b)
    {System.out.println(b.BoardingPass());}
    static void printdet(Details c)
    {System.out.println(c.BoardingPass());}
```

1.5) Composition

เป็นการนำ class ที่มีอยู่เดิมมาเป็น attribute ของ class ใหม่ เช่น คลาสDemostic ประกอบด้วยคลาส Details Dateandmin

Example :

```
class Dateandmin extends Demostic { //Sub class
    public Dateandmin(String arrival, String departures, String airlines, String flight, int aday, int amonth, int ayear,int ahour, int amin, int dday, int dmonth, int dyear,  int dhour, int dmin) {
    super(arrival, departures, airlines, flight);}
```

2) Source code ภาษา Java

2.1) Class Diagram

2.2) Source Code

Demostic Main class

```
class Demostic {// Main class
        private String arrival;  //Information hiding
        private String departures;  //Information hiding
        private String airlines;  //Information hiding
        private String flight;  //Information hiding
        //Constructors
        public Demostic (String arrival ,String departures,String airlines,String flight ){
        this.arrival=arrival;//Composition
        this.departures=departures;//Composition
        this.airlines=airlines;//Composition
        this.flight=flight;//Composition

        }
        //Encapsulation
        public String getArrival() {return arrival;}
        public String getDepartures(){return departures;}
        public String getAirlines(){return airlines;}
        public String getFlight(){return flight;}

        public void setArrival(String arrival) {this.arrival=arrival;}
        public void setDepartures(String depeatures, String
departures){this.departures=departures;}

        public void setAirlines(String airlines){this.airlines=airlines;}
        public void setFlight(String flight){this.flight=flight;}

        public String BoardingPass()
        {return  airlines+"\t DEMOSTIC BOARDING PASS"+"\n"+"ARRIVAL
:"+"\t"+arrival+"\n"+"DEPARTURES :"+"\t"+departures+"\n";}}
```

Details Sub class

```
class Details extends Demostic { //Sub class

        private String name;
        private String seat;
        private String gates;
       //Constructors and Inheritance
        public Details(String arrival, String departures, String airlines, String flight, String name,
String seat, String gates) {
                super(arrival, departures, airlines, flight);
                this.name=name;
                this.seat=seat;
                this.gates=gates;


        }
        public String getName(){return name;}
        public String getSeat(){return seat;}
        public String getGates(){return gates;}
        public void setName(String name){this.name=name;}
        public void setSeat(String seat){this.seat=seat;}
        public void setGates(String gates){this.gates=gates;}
        public String BoardingPass()
        {return "DETAILS "+"\n"+"Name :"+"\t"+name+"\n"+"SEAT :"+"\t"+seat+"\n"+"GATE
:"+"\t"+gates+"\n";}
}
```

Dateandmin Sub class

```
class Dateandmin extends Demostic { //Sub class
        private int aday;
        private  int amonth;
        private int ayear;
```

```
    private int dday;
    private int dmonth;
    private  int dyear;
    private  int ahour;
    private int amin;
    private int dhour;
    private int dmin;


    //Constructors and Inheritance
    public Dateandmin(String arrival, String departures, String airlines, String flight, int aday,
int amonth, int ayear,int ahour, int amin, int dday, int dmonth, int dyear,  int dhour, int dmin) {
            super(arrival, departures, airlines, flight);
              this.aday=aday;
              this.amonth=amonth;
              this.ayear=ayear;
              this.dday=dday;
              this.dmonth=dmonth;
              this.dyear=dyear;
              this.ahour=ahour;
              this.amin=amin;
              this.dhour=dhour;
              this.dmin=dmin;
    }
    public int getAday (){return aday;}
    public int getAmonth(){return amonth;}
    public int getAyear(){return ayear;}
    public int getDday(){return dday;}
    public int getDmonth(){return dmonth;}
    public int getDyear(){return dyear;}
    public int getAhour(){return ahour;}
    public int getAmin(){return amin;}
    public int getDhour(){return dhour;}
    public int getDmin(){return dmin;}
```

```
public void setAday (int aday){this.aday=aday;}
public void setAmonth(int amonth){this.amonth=amonth;}
public void setAyear(int ayear){this.ayear=ayear;}
public void setDday(int dday){this.dday= dday;}
public void setDmonth(int dmonth){this.dmonth=dmonth;}
public void setDyear(int dyear){this.dyear=dyear;}
public void setAhour(int ahour){this.ahour=ahour;}
public void setAmin(int amin){this.amin=amin;}
public void setDhour(int dhour){this.dhour= dhour;}
public void setDmin(int dmin){this.dmin= dmin;}

public String BoardingPass()
{return "DATE"+"\n"+"ARRIVAL  :"+"\t"+String.format("%02d", aday)+"/"+String.format("%02d",
amonth)+"/"+ayear+"\t"+String.format("%02d", ahour)+":"+String.format("%02d",amin)+"\n"
+"DEPARTURES :"+"\t"+String.format("%02d", dday)+"/"+String.format("%02d",
dmonth)+"/"+dyear+"\t"+String.format("%02d", dhour)+":"+String.format("%02d",dmin)+"\n";}
}}
}
```

PrintBoardingPass Class

```
import java.util.Scanner;//input import
import java.sql.Timestamp;

public class PrintBoardingPass {

	static void printde(Demostic a) {//Polymorphism Method
		System.out.println(a.BoardingPass());
	}
	static void printda(Dateandmin b)
	{System.out.println(b.BoardingPass());}
	static void printdet(Details c)
	{System.out.println(c.BoardingPass());}
	public static void main(String args[]){
		Scanner cin=new Scanner (System.in);
		System.out.println("MODE PRINT BOARDING PASS FOR DOMOSITC AIRPORT ");
		System.out.print("ARRIVAL : " );
		String ar= cin.nextLine();//input
		System.out.print("DEPARTURE : " );
		String de= cin.nextLine();//input
		System.out.print("AIRLINES : " );
		String ai= cin.nextLine();//input
		System.out.print("FLIGHT EG(SHORT AIRLINESNAME-SERIAL) : " );
		String fi= cin.nextLine();//input
		System.out.println("\n---------------------------- \n " );
		System.out.println("TIMES : " );
		System.out.println("ARRIVAL TIME EX:16/4/2017 13:45 IS 16420171345 : " );
		System.out.print("DAY : " );
		int ad= cin.nextInt();//input
		System.out.print("MONTH :" );
		int am= cin.nextInt();//input
		System.out.print("YEAR : " );
		int ay= cin.nextInt();//input
		System.out.println("TIMES  " );
```

```
System.out.print("HOUR : " );
int ah= cin.nextInt();//input
System.out.print("MIN: " );
int ami= cin.nextInt();//input
System.out.println("\n------------------------------ " );
System.out.println("DEPARTURE TIME EX:16/4/2017 14:55 IS 16420171455 : " );
System.out.print("DAY : " );
int dd= cin.nextInt();//input
System.out.print("MONTH :" );
int dm= cin.nextInt();//input
System.out.print("YEAR : " );
int dy= cin.nextInt();//input
System.out.println("TIMES  " );
System.out.print("HOUR : " );
int dh= cin.nextInt();//input
System.out.print("MIN: " );
int dmi= cin.nextInt();//input
System.out.println("\n------------------------------  " );
System.out.println("DETAILS  " );
String dd1=cin.nextLine();//input
System.out.print("NAME FIRST-FAMILY :  " );
String na=cin.nextLine();//input
System.out.print("SEAT EX  A2 :  " );
String se= cin.nextLine();//input

System.out.print("GATE EX: GATE-A41 :  " );
String ga= cin.nextLine();//input
System.out.println("\n------------------------------ \n " );
System.out.println("PROCESS..... " );
System.out.println("\n------------------------------ \n " );
System.out.println("****************************** \n " );
System.out.println("DONE \n" );
System.out.println("---------------------------------------------------- \n " );
//Polymorphism Method
printde(new Demostic(ar, de, ai, fi));
```

```
                printda(new Dateandmin(ar, de, ai, fi, ad, am, ay, ah, ami, dd, dm, dy, dh, dmi));
                printdet(new Details(ar, de, ai, fi, na, se, ga));
                //Polymorphism Method Print out time currents.
                Timestamp timestamp = new Timestamp(System.currentTimeMillis());
                System.out.print("Print in :");
        System.out.print(timestamp);
        System.out.print("\n");

                System.out.println("---------------------------------------------------------- \n " );



                /*printde(new Demostic("KHONKAEN", "DONMEUGN", "THAIAIRWAYS", "TG53-0"));
                printda(new Dateandmin("KHONKAEN", "DONMEUGN", "THAIAIRWAYS", "TG53-0" ,3,
5, 1995, 15, 5, 21, 5, 1996, 10, 00));
                printdet(new Details("KHONKAEN", "DONMEUGN", "THAIAIRWAYS", "TG53-0",
"Perzelita Kon", "T35", "D39"));*/
            }
}
```

Tested Demostic Class (Output)

```
MODE PRINT BOARDING PASS FOR DOMOSITC AIRPORT
ARRIVAL : KHONKAEN
DEPARTURE : DONMEUNG
AIRLINES : NOKAIR
FLIGHT EG(SHORT AIRLINESNAME-SERIAL) : DD9669

--------------------------------

TIMES :
ARRIVAL TIME EX:16/4/2017 13:45 IS 16420171345 :
DAY : 7
MONTH :3
YEAR : 2017
TIMES
HOUR : 10
MIN: 30

--------------------------------
DEPARTURE TIME EX:16/4/2017 14:55 IS 16420171455 :
DAY : 8
MONTH :3
YEAR : 2017
TIMES
HOUR : 10
MIN: 30

--------------------------------
DETAILS
NAME FIRST-FAMILY :  CHAOWARIN TUCHANTO
SEAT EX  A2 :  F4
GATE EX: GATE-A41 :  A03

--------------------------------

PROCESS.....................

--------------------------------

******************************

DONE

------------------------------------------------------------

NOKAIR   DEMOSTIC BOARDING PASS
ARRIVAL :       KHONKAEN
DEPARTURES :    DONMEUNG

DATE
ARRIVAL  :      07/03/2017      10:30
DEPARTURES :    08/03/2017      10:30

DETAILS
Name :  CHAOWARIN TUCHANTO
SEAT :  F4
GATE :  A03

Print in :2017-03-06 18:07:31.401
------------------------------------------------------------
```

เอกสารอ้างอิง

1. พนิดา พานิชกุล. (ธันวาคม 2552). “เทคโนโลยีเชิงวัตถุ (Object Oriented Technology)” ห้างหุ้นส่วนจำกัด วี.ซี.พี ซัคเซสกรุ๊ป 3/59 ซ.ลาดพร้าว แขวงจันทร์เกษม เขตจตุจักร กรุงเทพฯ 10900, (94-95).
2. วิกิพีเดีย สารานุกรมเสรี. การเขียนโปรแกรมเชิงวัตถุ, https://th.wikipedia.org/wiki/การเขียนโปรแกรมเชิงวัตถุ, สืบค้นเมื่อ : 28/02/2560.
3. Kunchit Surachon. “จาวากับคำว่า "Object Oriented Language"”, http://programming-to-learn.blogspot.com/2014/10/object-oriented-languate.html, สืบค้นเมื่อ : 28/02/2560.
4. nongtha57. “Java oop หรือ การเขียน Program เชิงวัตถุ”,https://nongtha57.wordpress.com/ , สืบค้นเมื่อ : 28/02/2560.

ประวัติผู้จัดทำ

ชื่อ : นายเชาวรินทร์ ตู้จันโต

(Chaowarin Tuchanto)

อายุ : 20ปี (2559)

รหัสฯ : 593021504-0

กำลังศึกษา : คณะวิทยาศาสตร์ ภาควิชาการคอมพิวเตอร์ สาขาเทคโนโลยีสารสนเทศ ชั้นปีที่ 1(2559) โครงการพิเศษ มหาวิทยาลัยขอนแก่น

การติดต่อ

Facebook : Chaowarin Tuchanto

Line : Perzelita

E-mail : Neung_secket@windowslive.com