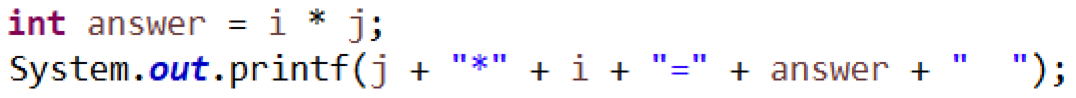
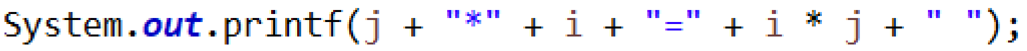
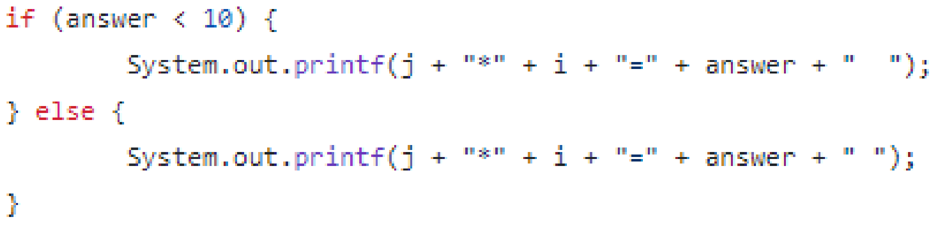
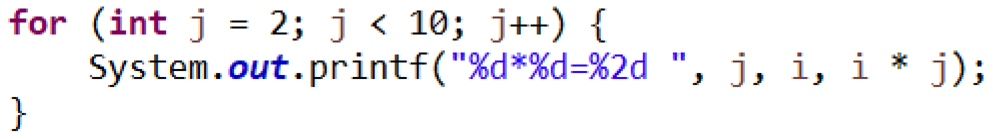
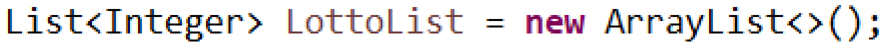
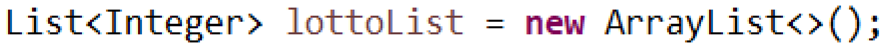
Code Review 紀錄內容			更新日期：2024/08/14
日期	CR 人員	題號	備註
2024/08/09	林家暉	1. 九九乘法表	

1. 減少變數的使用	只使用一次可不用宣告變數
	修改前：
	
	`int answer = i * j;`
	`System.out.printf(j + "*" + i + "=" + answer + "  ");`
	修改後：
	
	`System.out.printf(j + "*" + i + "=" + i * j + " ");`		

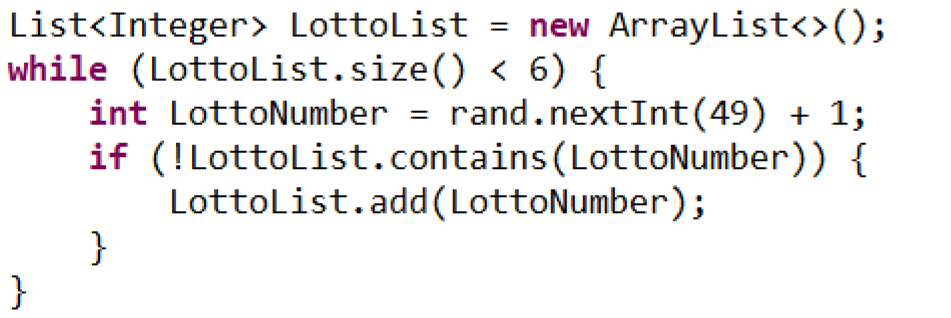
2. 利用 printf、%2d 格式化	修改前：
	
	`if (answer < 10) {`
	`    System.out.printf(j + "*" + i + "=" + answer + "  ");`
	`} else {`
	`    System.out.printf(j + "*" + i + "=" + answer + " ");`
	`}`
	修改後：
	
	`for (int j = 2; j < 10; j++) {`
	`    System.out.printf("%d*%d=%2d ", j, i, i * j);`
	`}`		

日期	CR 人員	題號	備註
2024/08/09	林家暉	2. 樂透程式	

3. 變數命名須符合規範	命名須有意義並符合駝峰或大寫底線
	修改前：
	
	`List<Integer> LottoList = new ArrayList<>();`
	修改後：
	
	`List<Integer> lottoList = new ArrayList<>();`		

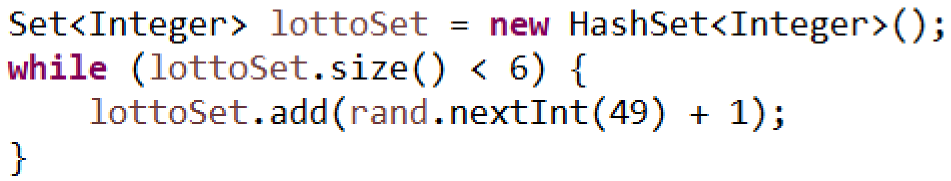
4. 利用 Set 取代 List

修改前：



```java
List<Integer> LottoList = new ArrayList<>();
while (LottoList.size() < 6) {
    int LottoNumber = rand.nextInt(49) + 1;
    if (!LottoList.contains(LottoNumber)) {
        LottoList.add(LottoNumber);
    }
}
```

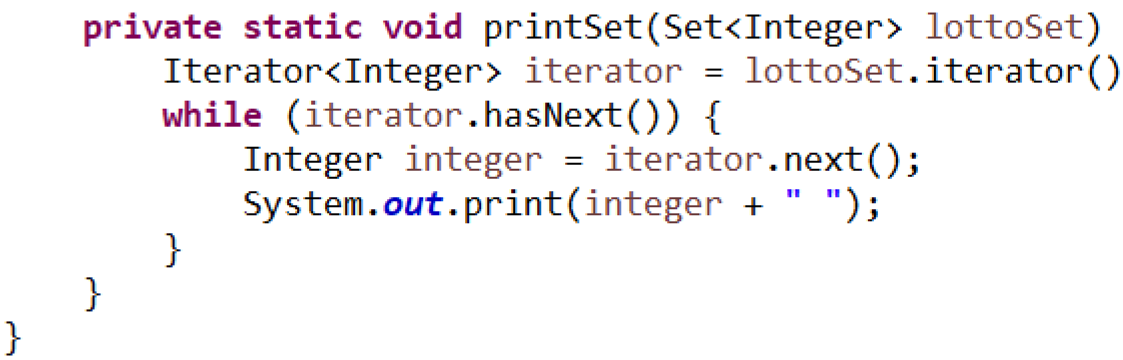
修改後：



```java
Set<Integer> lottoSet = new HashSet<Integer>();
while (lottoSet.size() < 6) {
    lottoSet.add(rand.nextInt(49) + 1);
}
```

5. 重複邏輯可建立方法

修改後：



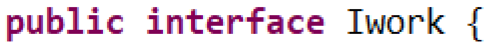
```java
    private static void printSet(Set<Integer> lottoSet)
        Iterator<Integer> iterator = lottoSet.iterator()
        while (iterator.hasNext()) {
            Integer integer = iterator.next();
            System.out.print(integer + " ");
        }
    }
}
```

日期	CR 人員	題號	備註
2024/08/09	林家暉	3. 員工資料存取	

6. 須注意題目要求

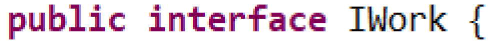
注意名稱

修改前：



```java
public interface Iwork {
```

修改後：

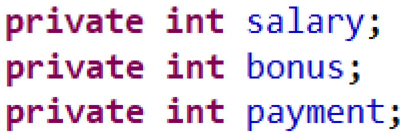


```java
public interface IWork {
```

7. 變數與錢有關，可使用 BigDecimal

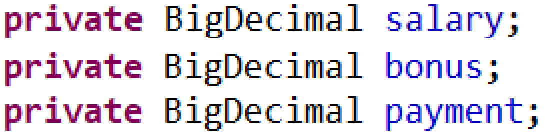
練習 BigDecimal 用法

修改前：

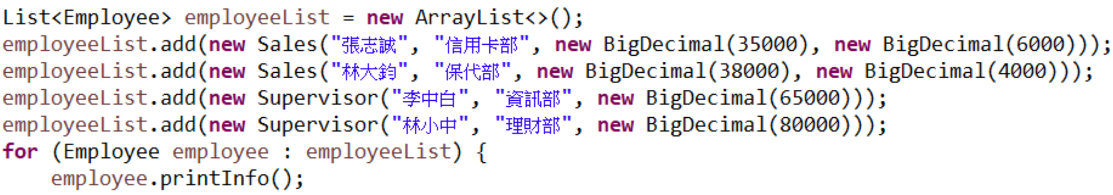


```java
private int salary;
private int bonus;
private int payment;
```

修改後：



```java
private BigDecimal salary;
private BigDecimal bonus;
private BigDecimal payment;
```

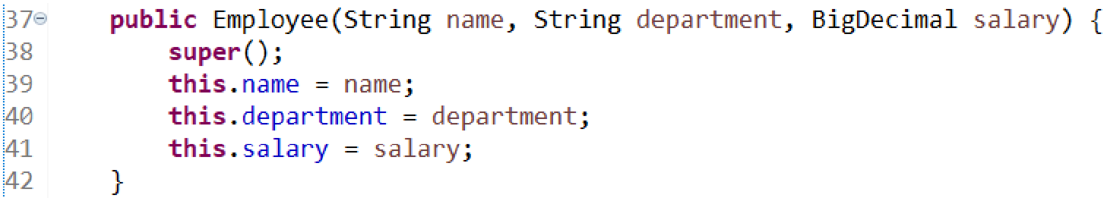


```java
List<Employee> employeeList = new ArrayList<>();
employeeList.add(new Sales("張志誠", "信用卡部", new BigDecimal(35000), new BigDecimal(6000)));
employeeList.add(new Sales("林大鈞", "保代部", new BigDecimal(38000), new BigDecimal(4000)));
employeeList.add(new Supervisor("李中白", "資訊部", new BigDecimal(65000)));
employeeList.add(new Supervisor("林小中", "理財部", new BigDecimal(80000)));
for (Employee employee : employeeList) {
    employee.printInfo();
```

<table>
<tr>
<td>8. 須符合規範</td>
<td>

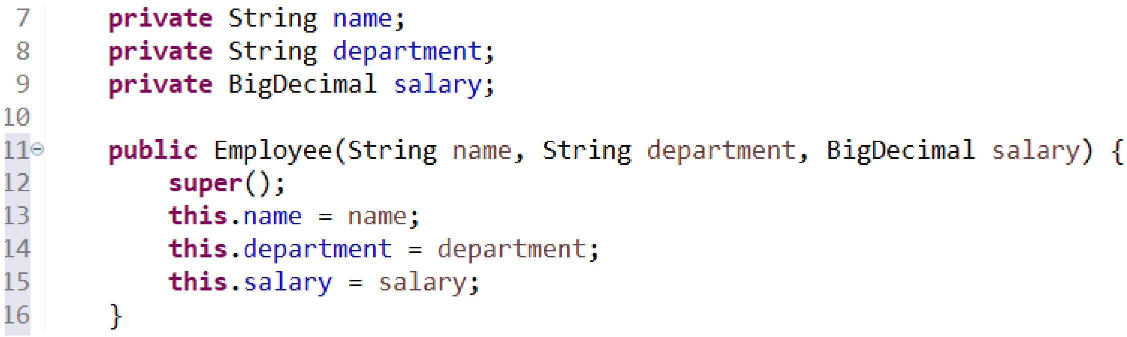
建構子須放在最前面，並在屬性變數的後面

修改前：



```
    public Employee(String name, String department, BigDecimal salary) {
        super();
        this.name = name;
        this.department = department;
        this.salary = salary;
    }
```

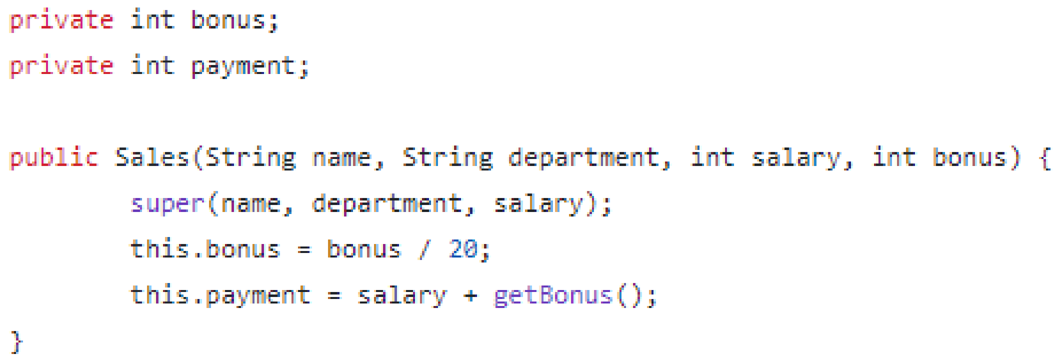
修改後：



```
    private String name;
    private String department;
    private BigDecimal salary;

    public Employee(String name, String department, BigDecimal salary) {
        super();
        this.name = name;
        this.department = department;
        this.salary = salary;
    }
```

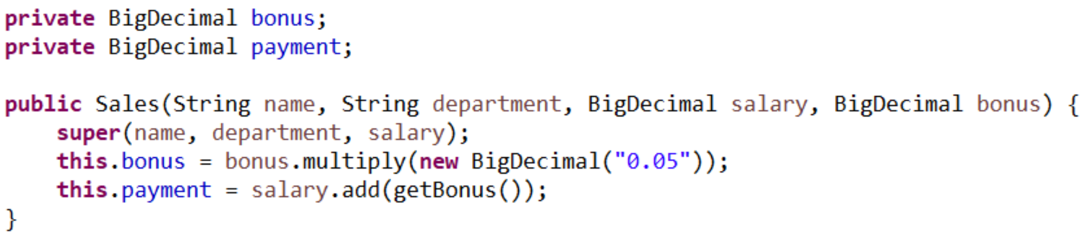
</td>
</tr>
<tr>
<td>9. 變數型別改為BigDecimal，並改變因應算法</td>
<td>

修改前：



```
private int bonus;
private int payment;

public Sales(String name, String department, int salary, int bonus) {
        super(name, department, salary);
        this.bonus = bonus / 20;
        this.payment = salary + getBonus();
}
```

修改後：

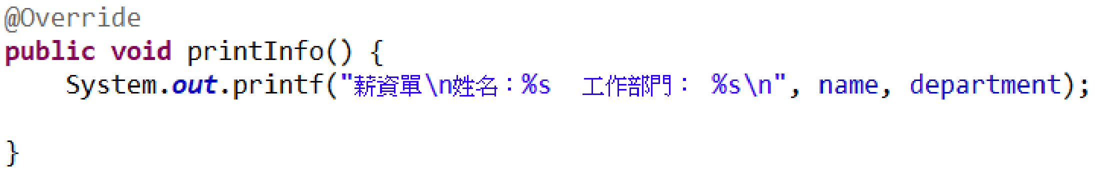


```
private BigDecimal bonus;
private BigDecimal payment;

public Sales(String name, String department, BigDecimal salary, BigDecimal bonus) {
    super(name, department, salary);
    this.bonus = bonus.multiply(new BigDecimal("0.05"));
    this.payment = salary.add(getBonus());
}
```

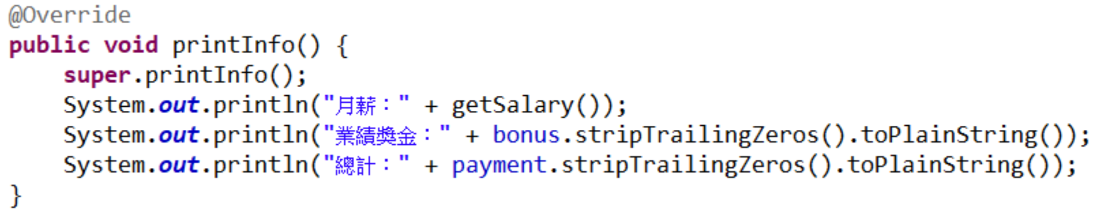
</td>
</tr>
</table>

<table>
<tr>
<td>10. 月薪為共同項目，可移至 Employee 一起打印</td>
<td colspan="3">

修改前：

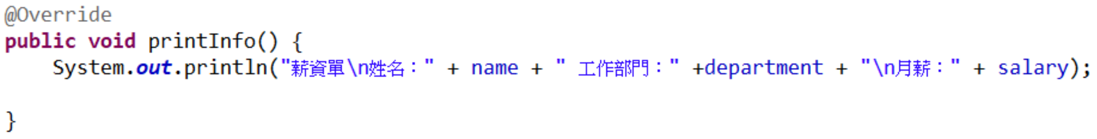


```
@Override
public void printInfo() {
    System.out.printf("薪資單\n姓名：%s　工作部門： %s\n", name, department);

}
```

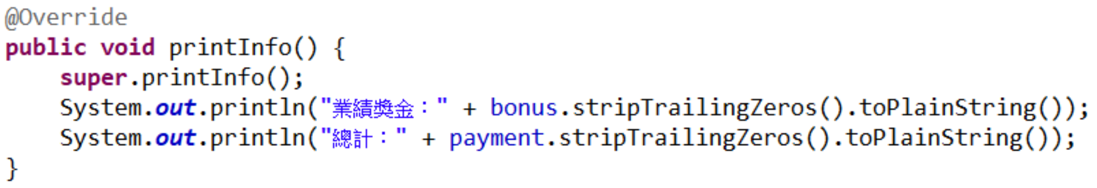


```
@Override
public void printInfo() {
    super.printInfo();
    System.out.println("月薪：" + getSalary());
    System.out.println("業績獎金：" + bonus.stripTrailingZeros().toPlainString());
    System.out.println("總計：" + payment.stripTrailingZeros().toPlainString());
}
```

修改後：



```
@Override
public void printInfo() {
    System.out.println("薪資單\n姓名：" + name + " 工作部門：" +department + "\n月薪：" + salary);

}
```

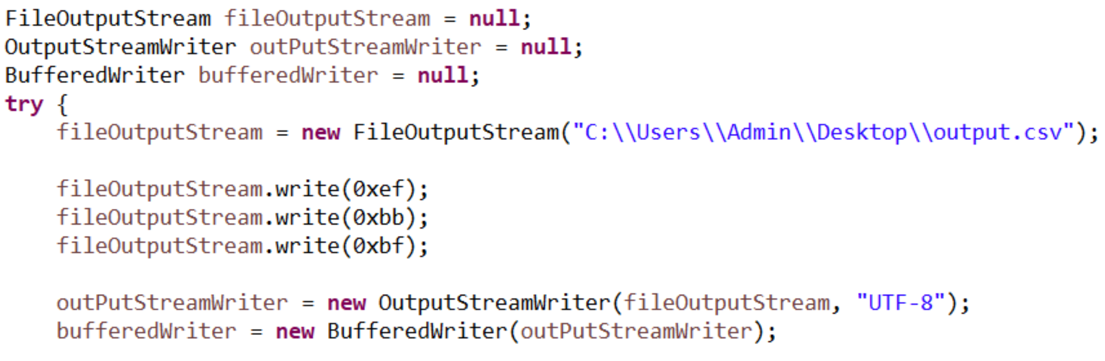


```
@Override
public void printInfo() {
    super.printInfo();
    System.out.println("業績獎金：" + bonus.stripTrailingZeros().toPlainString());
    System.out.println("總計：" + payment.stripTrailingZeros().toPlainString());
}
```

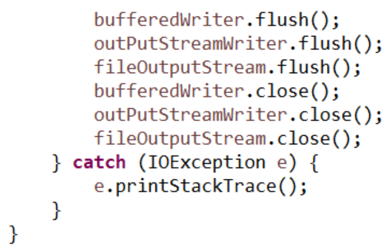
</td>
</tr>
<tr>
<td>日期</td>
<td>CR 人員</td>
<td>題號</td>
<td>備註</td>
</tr>
<tr>
<td>2024/08/09</td>
<td>林家暉</td>
<td>4. 輸出員工資料</td>
<td></td>
</tr>
<tr>
<td>11. Try catch 內資源須關閉</td>
<td colspan="3">

使用 try with resource 關閉資源

修改前：



```
FileOutputStream fileOutputStream = null;
OutputStreamWriter outPutStreamWriter = null;
BufferedWriter bufferedWriter = null;
try {
    fileOutputStream = new FileOutputStream("C:\\Users\\Admin\\Desktop\\output.csv");

    fileOutputStream.write(0xef);
    fileOutputStream.write(0xbb);
    fileOutputStream.write(0xbf);

    outPutStreamWriter = new OutputStreamWriter(fileOutputStream, "UTF-8");
    bufferedWriter = new BufferedWriter(outPutStreamWriter);
```



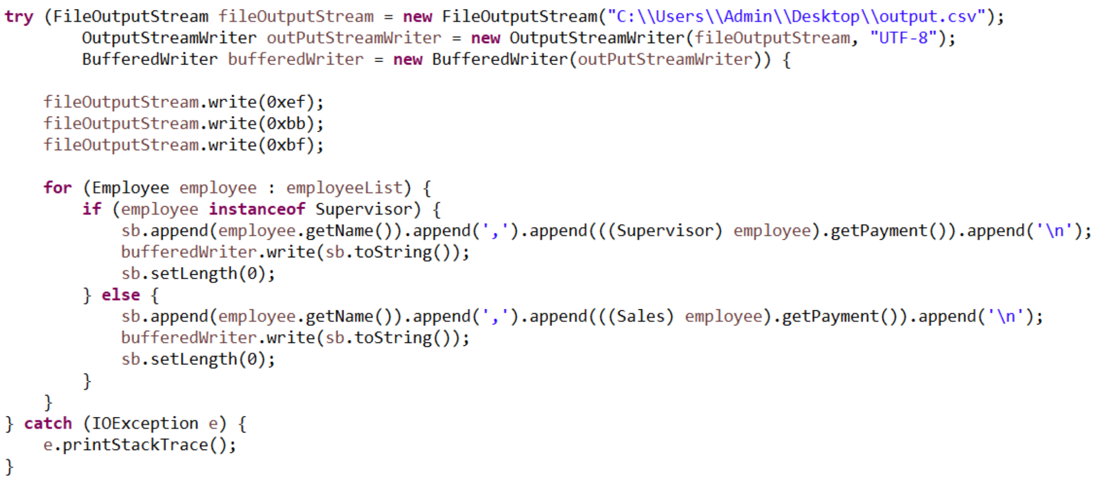
```
        bufferedWriter.flush();
        outPutStreamWriter.flush();
        fileOutputStream.flush();
        bufferedWriter.close();
        outPutStreamWriter.close();
        fileOutputStream.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

</td>
</tr>
</table>

11. Try catch 內資源須關閉

使用 try with resource 關閉資源

修改後：

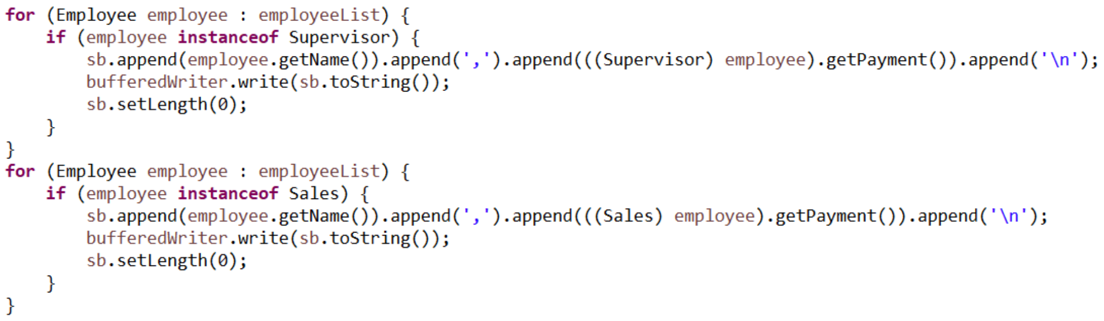


```java
try (FileOutputStream fileOutputStream = new FileOutputStream("C:\\Users\\Admin\\Desktop\\output.csv");
        OutputStreamWriter outPutStreamWriter = new OutputStreamWriter(fileOutputStream, "UTF-8");
        BufferedWriter bufferedWriter = new BufferedWriter(outPutStreamWriter)) {

    fileOutputStream.write(0xef);
    fileOutputStream.write(0xbb);
    fileOutputStream.write(0xbf);

    for (Employee employee : employeeList) {
        if (employee instanceof Supervisor) {
            sb.append(employee.getName()).append(',').append(((Supervisor) employee).getPayment()).append('\n');
            bufferedWriter.write(sb.toString());
            sb.setLength(0);
        } else {
            sb.append(employee.getName()).append(',').append(((Sales) employee).getPayment()).append('\n');
            bufferedWriter.write(sb.toString());
            sb.setLength(0);
        }
    }
} catch (IOException e) {
    e.printStackTrace();
}
```

12. 優化程式碼

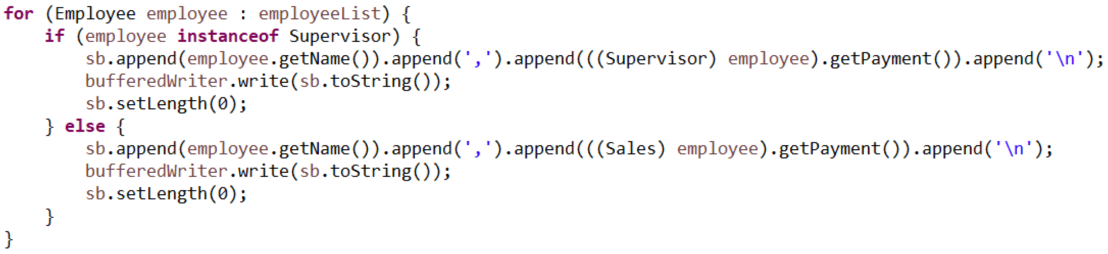
相同條件可用同一個迴圈

修改前：



```java
for (Employee employee : employeeList) {
    if (employee instanceof Supervisor) {
        sb.append(employee.getName()).append(',').append(((Supervisor) employee).getPayment()).append('\n');
        bufferedWriter.write(sb.toString());
        sb.setLength(0);
    }
}
for (Employee employee : employeeList) {
    if (employee instanceof Sales) {
        sb.append(employee.getName()).append(',').append(((Sales) employee).getPayment()).append('\n');
        bufferedWriter.write(sb.toString());
        sb.setLength(0);
    }
}
```

修改後：



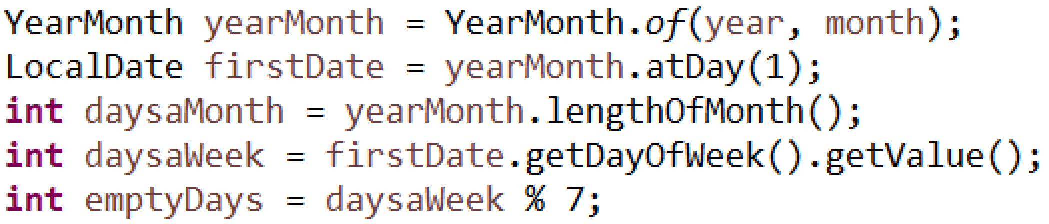
```java
for (Employee employee : employeeList) {
    if (employee instanceof Supervisor) {
        sb.append(employee.getName()).append(',').append(((Supervisor) employee).getPayment()).append('\n');
        bufferedWriter.write(sb.toString());
        sb.setLength(0);
    } else {
        sb.append(employee.getName()).append(',').append(((Sales) employee).getPayment()).append('\n');
        bufferedWriter.write(sb.toString());
        sb.setLength(0);
    }
}
```

日期	CR 人員	題號	備註
2024/08/14	林家暉	5. 熟悉日期相關類別操作	

13. 減少變數的使用

只使用一次可不用宣告變數

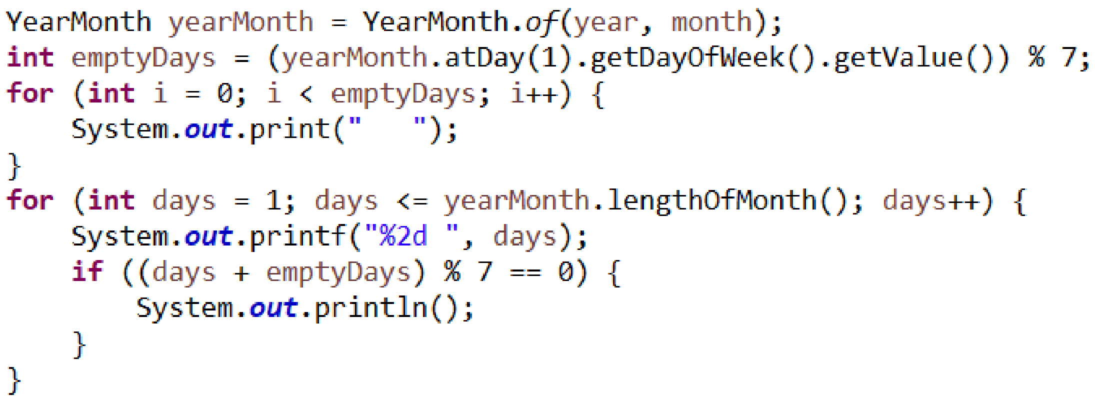
修改前：



```java
YearMonth yearMonth = YearMonth.of(year, month);
LocalDate firstDate = yearMonth.atDay(1);
int daysaMonth = yearMonth.lengthOfMonth();
int daysaWeek = firstDate.getDayOfWeek().getValue();
int emptyDays = daysaWeek % 7;
```

13. 減少變數的使用

修改後：



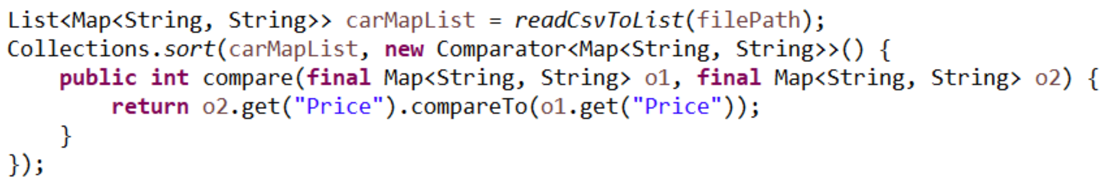
```java
YearMonth yearMonth = YearMonth.of(year, month);
int emptyDays = (yearMonth.atDay(1).getDayOfWeek().getValue()) % 7;
for (int i = 0; i < emptyDays; i++) {
    System.out.print("   ");
}
for (int days = 1; days <= yearMonth.lengthOfMonth(); days++) {
    System.out.printf("%2d ", days);
    if ((days + emptyDays) % 7 == 0) {
        System.out.println();
    }
}
```

日期	CR 人員	題號	備註
2024/08/14	林家暉	6. 熟悉資料分群累計	

14. 價格排序用 BigDecimal

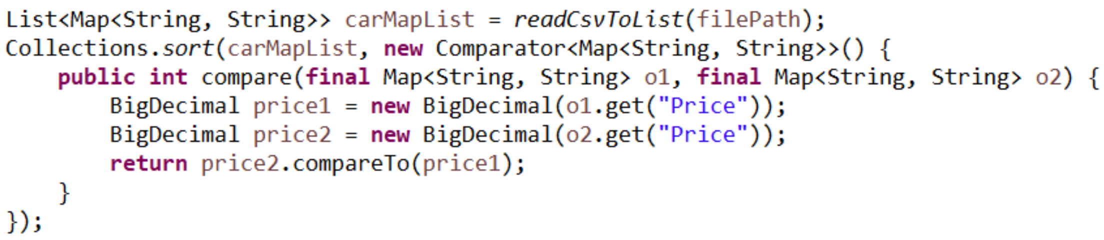
使用 String 只會比較第一位數字

修改前：



```java
List<Map<String, String>> carMapList = readCsvToList(filePath);
Collections.sort(carMapList, new Comparator<Map<String, String>>() {
    public int compare(final Map<String, String> o1, final Map<String, String> o2) {
        return o2.get("Price").compareTo(o1.get("Price"));
    }
});
```

修改後：

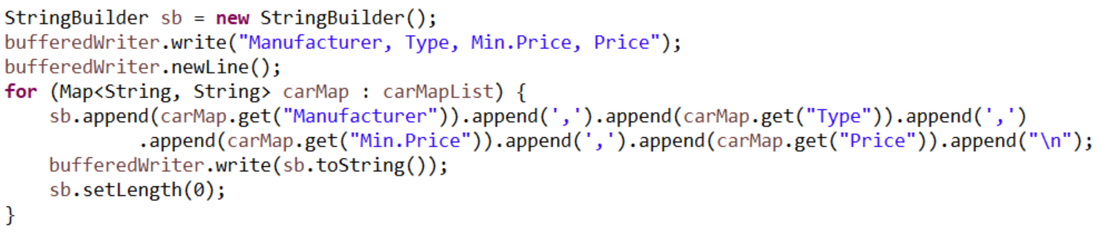


```java
List<Map<String, String>> carMapList = readCsvToList(filePath);
Collections.sort(carMapList, new Comparator<Map<String, String>>() {
    public int compare(final Map<String, String> o1, final Map<String, String> o2) {
        BigDecimal price1 = new BigDecimal(o1.get("Price"));
        BigDecimal price2 = new BigDecimal(o2.get("Price"));
        return price2.compareTo(price1);
    }
});
```

15. String Builder 類別要使用時再建立

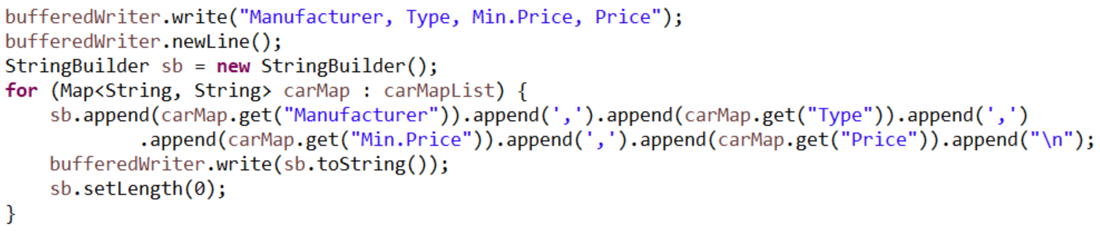
放在所需迴圈前，以免下方程式碼有誤無法進入迴圈

修改前：



```java
StringBuilder sb = new StringBuilder();
bufferedWriter.write("Manufacturer, Type, Min.Price, Price");
bufferedWriter.newLine();
for (Map<String, String> carMap : carMapList) {
    sb.append(carMap.get("Manufacturer")).append(',').append(carMap.get("Type")).append(',')
            .append(carMap.get("Min.Price")).append(',').append(carMap.get("Price")).append("\n");
    bufferedWriter.write(sb.toString());
    sb.setLength(0);
}
```

修改後：

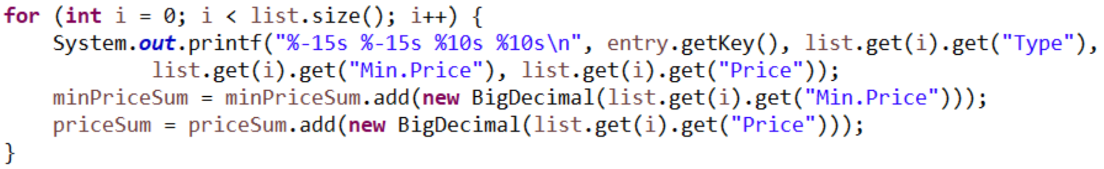


```java
bufferedWriter.write("Manufacturer, Type, Min.Price, Price");
bufferedWriter.newLine();
StringBuilder sb = new StringBuilder();
for (Map<String, String> carMap : carMapList) {
    sb.append(carMap.get("Manufacturer")).append(',').append(carMap.get("Type")).append(',')
            .append(carMap.get("Min.Price")).append(',').append(carMap.get("Price")).append("\n");
    bufferedWriter.write(sb.toString());
    sb.setLength(0);
}
```

16. 重複用方法取值應宣告為變數

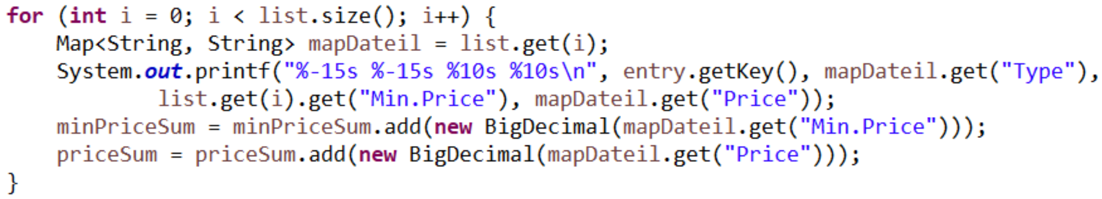
重複使用 list.get(i)，將其宣告為 Map<String, String> mapDateil

修改前：



```
for (int i = 0; i < list.size(); i++) {
    System.out.printf("%-15s %-15s %10s %10s\n", entry.getKey(), list.get(i).get("Type"),
            list.get(i).get("Min.Price"), list.get(i).get("Price"));
    minPriceSum = minPriceSum.add(new BigDecimal(list.get(i).get("Min.Price")));
    priceSum = priceSum.add(new BigDecimal(list.get(i).get("Price")));
}
```

修改後：



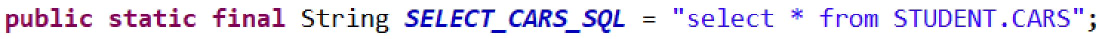
```
for (int i = 0; i < list.size(); i++) {
    Map<String, String> mapDateil = list.get(i);
    System.out.printf("%-15s %-15s %10s %10s\n", entry.getKey(), mapDateil.get("Type"),
            list.get(i).get("Min.Price"), mapDateil.get("Price"));
    minPriceSum = minPriceSum.add(new BigDecimal(mapDateil.get("Min.Price")));
    priceSum = priceSum.add(new BigDecimal(mapDateil.get("Price")));
}
```

日期	CR 人員	題號	備註
2024/08/14	林家暉	7. 資料庫	

17. SQL 查詢不要使用*

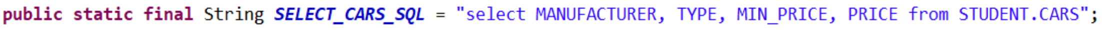
須將所有欄位列出，否則效能差

修改前：



```
public static final String SELECT_CARS_SQL = "select * from STUDENT.CARS";
```

修改後：：

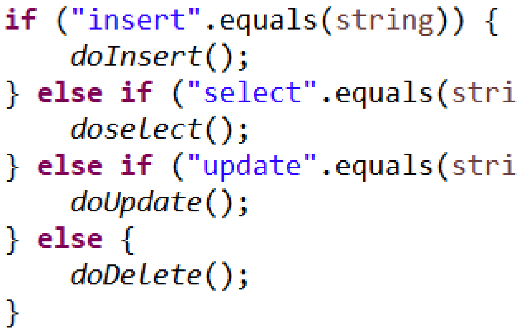


```
public static final String SELECT_CARS_SQL = "select MANUFACTURER, TYPE, MIN_PRICE, PRICE from STUDENT.CARS";
```

18. 注意判斷式、處理邏輯細節

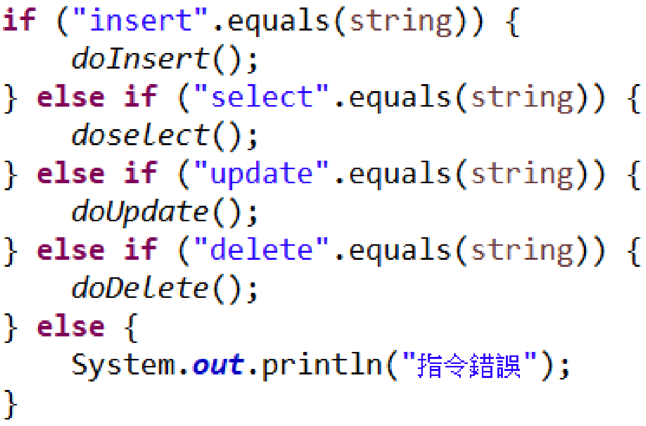
若將 doDelete 放在 else，輸入亂碼也會執行

修改前：



```
if ("insert".equals(string)) {
    doInsert();
} else if ("select".equals(string)) {
    doselect();
} else if ("update".equals(string)) {
    doUpdate();
} else {
    doDelete();
}
```

修改後：

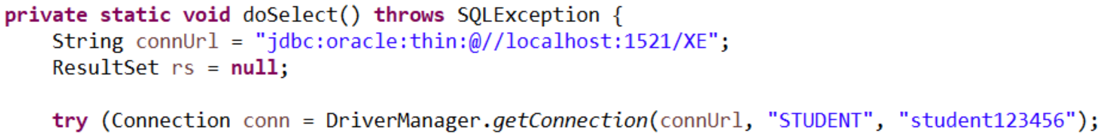


```
if ("insert".equals(string)) {
    doInsert();
} else if ("select".equals(string)) {
    doselect();
} else if ("update".equals(string)) {
    doUpdate();
} else if ("delete".equals(string)) {
    doDelete();
} else {
    System.out.println("指令錯誤");
}
```

<table>
<tr>
<td>19. 移除無用變數，若成員變數已宣告可重複使用</td>
<td>

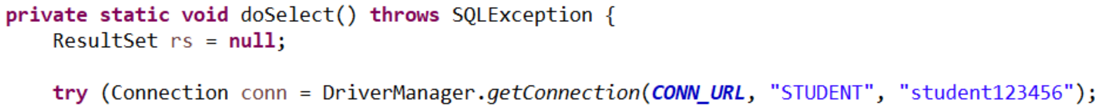
前面已宣告 CONN_URL 成員變數可直接使用

修改前：



```
private static void doSelect() throws SQLException {
    String connUrl = "jdbc:oracle:thin:@//localhost:1521/XE";
    ResultSet rs = null;

    try (Connection conn = DriverManager.getConnection(connUrl, "STUDENT", "student123456");
```

修改後：

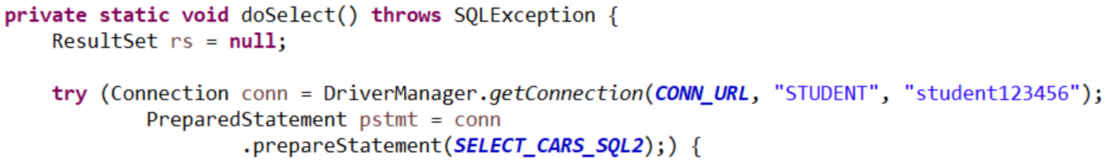


```
private static void doSelect() throws SQLException {
    ResultSet rs = null;

    try (Connection conn = DriverManager.getConnection(CONN_URL, "STUDENT", "student123456");
```

</td>
</tr>
<tr>
<td>20. 程式碼優化</td>
<td>

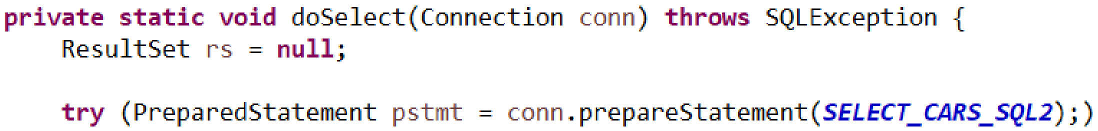
將已建立的 conn 連線作為參數，傳進方法使用減少建立額外的連線

修改前：



```
private static void doSelect() throws SQLException {
    ResultSet rs = null;

    try (Connection conn = DriverManager.getConnection(CONN_URL, "STUDENT", "student123456");
            PreparedStatement pstmt = conn
                    .prepareStatement(SELECT_CARS_SQL2);) {
```

修改後：



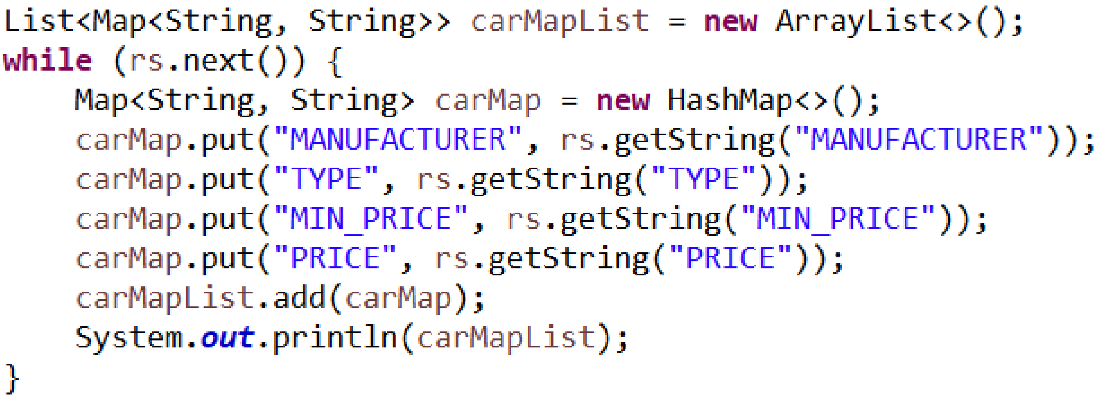
```
private static void doSelect(Connection conn) throws SQLException {
    ResultSet rs = null;

    try (PreparedStatement pstmt = conn.prepareStatement(SELECT_CARS_SQL2);)
```

</td>
</tr>
<tr>
<td>21. 程式碼優化</td>
<td>

doSelect()內查詢結果邏輯優化，減少建立額外變數，可直接 print 結果

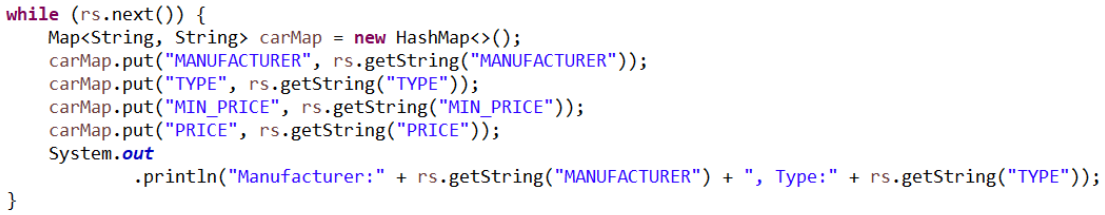
※補充：若要印出 carMapList 應放在迴圈外否則會重複印出

修改前：



```
List<Map<String, String>> carMapList = new ArrayList<>();
while (rs.next()) {
    Map<String, String> carMap = new HashMap<>();
    carMap.put("MANUFACTURER", rs.getString("MANUFACTURER"));
    carMap.put("TYPE", rs.getString("TYPE"));
    carMap.put("MIN_PRICE", rs.getString("MIN_PRICE"));
    carMap.put("PRICE", rs.getString("PRICE"));
    carMapList.add(carMap);
    System.out.println(carMapList);
}
```

修改後：



```
while (rs.next()) {
    Map<String, String> carMap = new HashMap<>();
    carMap.put("MANUFACTURER", rs.getString("MANUFACTURER"));
    carMap.put("TYPE", rs.getString("TYPE"));
    carMap.put("MIN_PRICE", rs.getString("MIN_PRICE"));
    carMap.put("PRICE", rs.getString("PRICE"));
    System.out
            .println("Manufacturer:" + rs.getString("MANUFACTURER") + ", Type:" + rs.getString("TYPE"));
}
```

</td>
</tr>
</table>

<table>
<tr>
<td>22. 關閉 scanner 資源</td>
<td>

利用 try with resource 關閉資源

修改後：

```java
try (Scanner scanner = new Scanner(System.in)) {
    System.out.print("請輸入製造商:");
    pstmt.setString(1, scanner.next());

    System.out.print("請輸入類型:");
    pstmt.setString(2, scanner.next());

    System.out.print("請輸入底價:");
    pstmt.setBigDecimal(3, scanner.nextBigDecimal());

    System.out.print("請輸入售價:");
    pstmt.setBigDecimal(4, scanner.nextBigDecimal());

    pstmt.executeUpdate();
    conn.commit();
    System.out.println("新增成功");

} catch (Exception e) {
    try {
        conn.rollback();
        System.out.println("新增失敗");
    } catch (SQLException sqle) {
        sqle.printStackTrace();
    }
    e.printStackTrace();
}
```

</td>
</tr>
<tr>
<td>23. Catch 至少印出異常訊息</td>
<td>

修改前：

```java
} catch (Exception e) {
}
```

修改後：

```java
} catch (Exception e) {
    e.printStackTrace();
}
```

</td>
</tr>
</table>