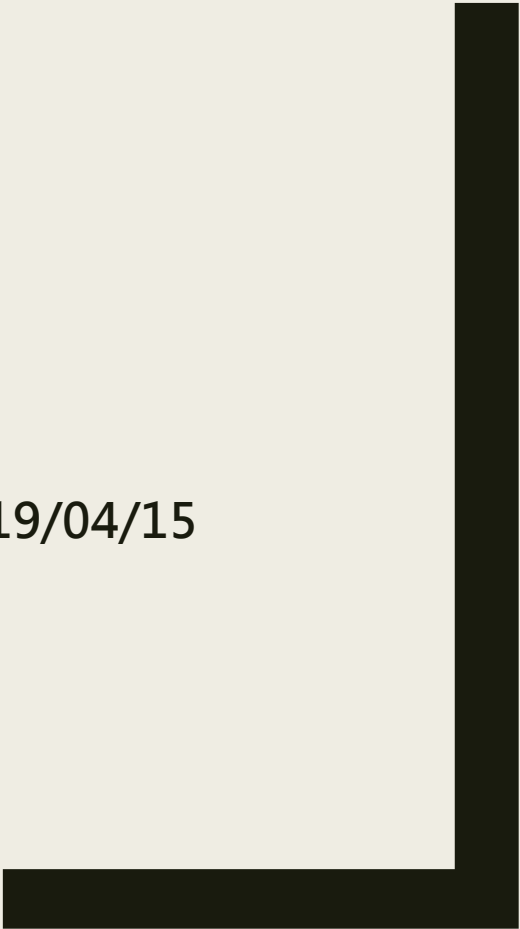




# Lab 5

2019/04/15



# 上機 (1)

## ■ New E3 課程網頁內

國立交通大學 數位教學平台

課程資訊

- 課程綱要
- 成員
- 公告列表
- 我的郵件

內容管理

- 大綱管理**
- 教材管理
- 作業管理
- 討論區管理
- 試卷管理
- 題庫維護
- 分組管理

評量管理

- 成績管理
- 配分設定

【107下】1190資料結構與物件導向程式設計 Programming

Slides 2019/2/18

Slides 2019/3/4

TA Courses

2019/3/4 Practice

考試時間到才會開啟，  
10分鐘內下載完畢

遲到超過10分鐘，  
該次以0分計

# 上機 (1)

三  國立交通大學 數位教學平台

課程資訊

- 課程綱要
- 成員
- 公告列表
- 我的郵件

內容管理

- 大綱管理
- 教材管理
- 作業管理
- 討論區管理
- 試卷管理
- 題庫維護
- 分組管理

評量管理

- 成績管理
- 配分設定

【107上】1189計算機概論與程式設計 Int

Quiz1

- Quiz1\_Q1\_sample.c
- Quiz1\_Q2\_sample.c
- Quiz1.pdf

下載資料夾 編修

點擊下載

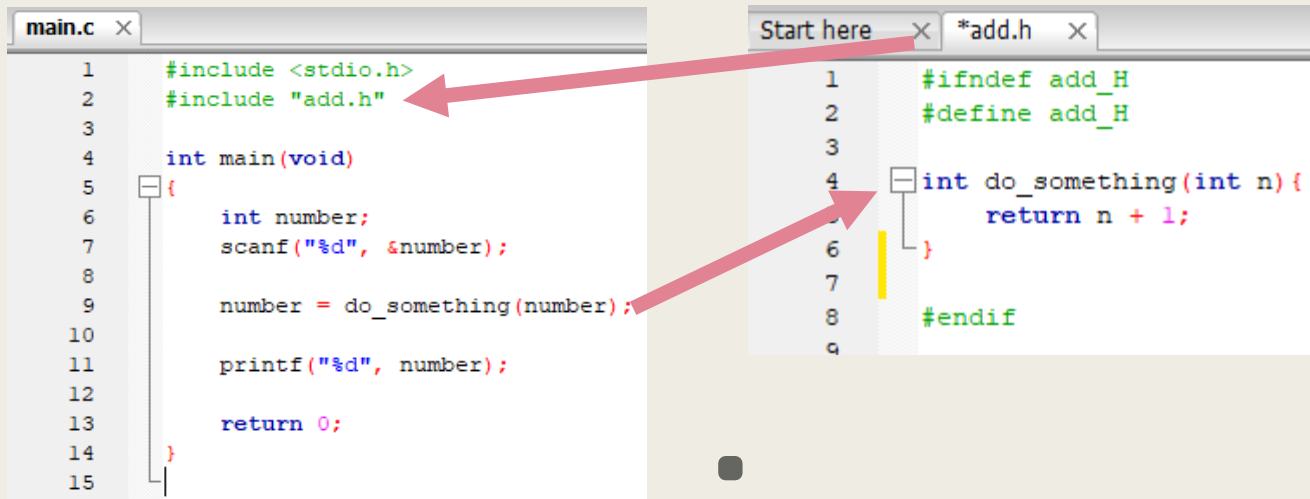
下載完記得先解壓縮，再開始編寫程式

# 考試規則

1. 可以翻閱你覺得有幫助的書、講義 (教室會斷網!!)
2. 不得作弊，違者依校規論處
3. 若有格式錯誤的情形，會將該題分數  $\times 0.8$  計算
4. 本次練習都只需繳交 **Header file**  
不得更該 `main_Q1.c` 中任何內容  
繳交時請自行將 Header file 檔名改為 學號 – 題號  
如：0756704-1.h  
註：不需變更 `ifndef`, `define`, `include` 的檔名
5. 總共只有一**次**繳交機會，請務必確認格式正確後，再舉手找助教繳交。
6. 行動電子產品 (手機、平板電腦等等)請收在包包內，不要放在桌面上或使用它。

# Header file

- Header file contains function declarations and macro definitions to be shared between several source files.
- For example



The image shows two code editors side-by-side. The left editor, titled 'main.c', contains the following code:

```
1  #include <stdio.h>
2  #include "add.h"
3
4  int main(void)
5  {
6      int number;
7      scanf("%d", &number);
8
9      number = do_something(number);
10
11     printf("%d", number);
12
13     return 0;
14 }
15
```

The right editor, titled '\*add.h', contains the following code:

```
1  #ifndef add_H
2  #define add_H
3
4  int do_something(int n){
5      return n + 1;
6  }
7
8  #endif
9
```

Two red arrows indicate the relationship: one arrow points from the `#include "add.h"` line in `main.c` to the header file, and another arrow points from the `do_something` function call in `main.c` to the `do_something` function definition in `*add.h`.

- In Header file (\*.h) , you can add any function or declaration **except** main function

# Q1 – Bank Account

## Description

Create an inheritance hierarchy that a bank might use to represent customers' bank accounts. All customers can debit(withdraw) and credit(deposit) money from their account. More specific types of accounts also exist. **Savings accounts** earn interest on the money they hold. **Checking accounts** charge a fee per transaction.

Create an inheritance hierarchy containing base class **Account** and derive classes **SavingsAccount** and **CheckingAccount** that inherit from class Account.

Base class Account should include one data member of type *double* to represent the account balance and uses it to initialize the data member. The constructor should validate the initial balance, if the balance is smaller than 0, then set the balance to 0.

The class should provide three member function. Member function **credit** should add an amount to the current balance. Member function **debit** should withdraw money from Account. If the debit amount exceed the Account's balance, then the balance should remain unchanged. Member function **getBalance** should return the current balance. The balance of account should always equal or greater than 0.

# Q1 – Bank Account

## Description

Derive class **SavingsAccount** should inherit the functionality of Account, but also include a data member of type *double* indicating the interest rate assigned to the Account. SavingsAccount's constructor should receive initial balance and initial interest rate, if the balance or interest rate is smaller than 0, then set it to 0. Provide a public member function **calculateInterest** that add the amount of interest to the balance earned by an account. The amount of interest is calculated by multiply the interest rate by the account balance.

Derive class **CheckingAccount** inherit from base class Account and include data member of type *double* that represent the fee charged per transaction. CheckingAccount's constructor should receive the initial balance and a parameter indicate a fee amount. Class CheckingAccount should redefine the member functions **credit** and **debit** so that they subtract the fee from the account when either transaction is performed successfully.

# Q1 - Bank Account

## Description

After the classes have initialized, we can initialize the three different classes and input their balance, interest rate and transaction fee if needed.

Then we can manipulate the accounts by input the account code, instruction code and value sequentially if needed with the following code:

input_account	description
1	account1
2	account2
3	account3

input_instruction	description	Input content
1	debit	Debit amount
2	credit	Credit amount
3	Calculate interest rate	

The program will terminate when user prompt “0 0 0” and return the balance of the each account.



# Example

## ■ Sample Input

100  
200 0.1  
300 10  
1 1 150  
2 3 0  
3 2 700  
0 0 0

## ■ Sample Output

Account1 : \$100  
Account2 : \$220  
Account3 : \$990

# Q2 – Package Inheritance

## Description

Create an inheritance hierarchy to represent various types of packages. Use class **Package** as the base class of the hierarchy, then includes classes **TwoDayPackage** that derives from **Package**. Base class **Package** should include data members representing the **name**, **address**, **city**, **zipcode**, **weight**(in ounces) and **cost per ounce**. **Package**'s constructor should initialize these data members.

Also, create the class **Human** that is independent of Base class **Package**. Class **Human** should provide a public member function **writeName**, **writeAddress**, **writeCity**, **writeZipCode** for giving the package info and a member function **getZipCode** for checking whether the sender and the receiver are in the same city or not. The above functions return **string**, **string**, **string**, **int**, **int** respectively.

# Q2 – Package Inheritance

## Description(cont.)

**Package** should provide a public member function **calculateCost** that returns a **double** indicating the cost associated with shipping the package and a public member function **setTransferFee** that will initialize the private **double** variable **transferFee** within the **Package** class. **Package**'s **calculateCost** function should determine the cost by multiplying the weight by the cost per ounce and it should consider **adding the transferFee** when it was shipped in different city.

# Q2 – Package Inheritance

## Description(cont.)

Derived class **TwoDayPackage** should inherit the functionality of base class **Package**, but also include a data member that represents a **flat fee** that the shipping company charges for two-day-delivery service. **TwoDayPackage**'s constructor should receive a value to initialize this data member.

**TwoDayPackage** should **redefine** member function **calculateCost** so that it computes the shipping cost by adding the flat fee to the weight-based cost calculated by base class **Package**'s **calculateCost** function.

[Remind]

In the main.cpp file, the Sender object and the Receiver object are in the Human class. The Sender object writes the info for the Package class. And the receiver object prints out its own info after the Sender has paid the cost. Only the Sender pays the cost and the cost is calculated from **calculateCost** function.

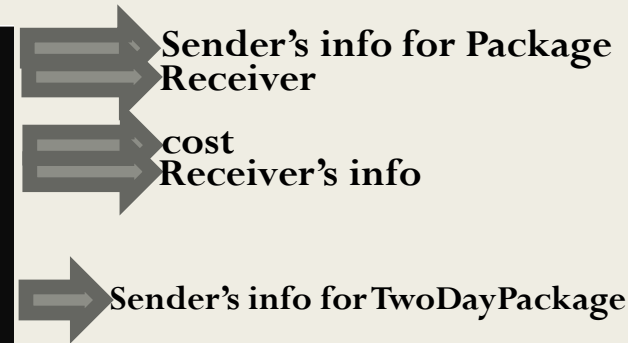
# Example

```
john west_street Taipei 100 25.6 3.7
james south_street Taipei 100

*****
Total cost: 94.72
james
south_street
Taipei
100
*****
Eric east_street Tainan 700 30.8 40.9 150.0
Kenneth north_street Taipei 100 30.8 40.9

*****
Total cost: 1509.72
Kenneth
north_street
Taipei
100
*****

-----
Process exited after 217.8 seconds with return value 0
請按任意鍵繼續 . . .
```



[Remind]

In the example, each city has its own zipcode( Taipei:100,Hsinchu:300, Taichung:400,Tainan:700,Kaoshiung:800). The value of cost is in two decimal places and the value of input for weight(double), cost\_per\_ounce(double), flatfee(double) are in one decimal places. Also, cost\_per\_ounce will not change even though in different city. [Hint: only consider the transferfee].

Q&A