Turnin 作業 9

- Turnin Code: cpp.hw9
- Due Date: 5/25 Monday 23:59:00 (midnight) **Hard Deadline**

繳交方式說明

本次作業繳交將以資料夾的形式繳交，需要為每一題建立一個資料夾（資料夾名稱為該題題目前方之代號，第一題為p1第二題為p2餘以此類推）。

繳交說明可參考Turnin 作業 4】中 p6 到 p10

任何未依照正確繳交格式的檔案將以 0 分計。



本文使用「 」及「 」代表「空白字元」與Enter 換行字元」，並且將使用者輸入的部份使用灰階方式顯示。另外，題目的執行結果中，如果出現「(」、「)」、「:」、「;」、「.」與「,」等符號，皆為英文半形！

---

p1 學生類別實作（外籍生繼承一般學生）

類別的繼承就像是孩子（子類別）直接擁有父親（父類別）的資產（成員變數）以及能力（成員函式），若要開發另一個相似功能的類別可以使用繼承的方式讓子類別繼承父類別後再（透過多型、多載等物件導向特性）開發新的功能，這可以減少過多相似的程式碼，並提升程式碼的可維護性。

接續 cpp.hw8本題的目的是計算一般學生與外籍生的學期成績。

Person 類別相關定義請參考下列的 person.h 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include <string>
using namespace std;

class Person
{
protected:
    string name;
    string student_id;
public:
```

```
    void set_name(string name);
    void set_student_id(string student_id);

    string get_name();
    string get_student_id();
};
```

請參考下列的 main.cpp 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include "foreign_student.h"
#include <iostream>
using namespace std;

int main()
{
    Student Kennedy;
    ForeignStudent Juila;

    Kennedy.set_name("Kennedy");
    Kennedy.set_student_id("cbb114901");
    Kennedy.set_score(87.42, 73.56, 95.30, 68.18);
    cout << "Student [" << Kennedy.get_name() << "]("
         << Kennedy.get_student_id() << ")" << endl;
    Kennedy.calculate_term_score();
    cout << "general performance: " << Kennedy.get_general_performance() <<
endl
         << "exam1: " << Kennedy.get_exam1() << endl
         << "exam2: " << Kennedy.get_exam2() << endl
         << "final exam: " << Kennedy.get_final_exam() << endl
         << "term score: " << Kennedy.get_term_score() << endl;

    cout << "--------------------------------------------" << endl;
    Juila.set_name("Juila");
    Juila.set_student_id("cbb113442");
    Juila.set_score(87.42, 73.56, 95.30, 68.18);
    Juila.set_nationality("Korea (ROK)");
    Juila.set_passport_id("M98238646");
    Juila.calculate_term_score();
    cout << "Foreign student [" << Juila.get_name() << "]("
         << Juila.get_student_id() << ", "
         << Juila.get_nationality() << " passport no. "
         << Juila.get_passport_id() << "): " << endl
         << endl;

    cout << "general performance: " << Juila.get_general_performance() <<
endl
         << "exam1: " << Juila.get_exam1() << endl
         << "exam2: " << Juila.get_exam2() << endl
         << "final exam: " << Juila.get_final_exam() << endl
```

```
        << "term score: " << Juila.get_term_score() << endl;

    return 0;
}
```

請參考 person.h 的 Person 類別，並完成 Student 類別與 ForeignStudent 類別。請注意，Student 類別需繼承自 Person 類別，ForeignStudent 需繼承自 Student 類別。

請參考 main.cpp 的 12 與 25 行 set_score 函式，其引數從左至右分別為「general performance（平時成績）」、「exam1（第一次期中考）」、「exam2（第二次期中考）」與「final exam（期末考）」，這四項成績分別佔學期成績的 25%，計算一般學生學期成績時按照此比重進行計算。

考慮到外籍生或多或少有語言以及文化的適應障礙，為了幫助與鼓勵這些同學們投入更多時間在課程內容的預習、複習以及理解上而設計了另一種計算分數的方式：平時成績 * 1.25 並以 100 分為上限，再依照上述的比重進行計算。

不論是一般學生（Student 類別）或是外籍生（ForeignerStudent 類別），計算學期成績時請四捨五入到小數點後第一位。

檔案名稱	**檔案用途與撰寫要求**	**-**
main.cpp	主程式	題目提供原始碼
person.h	Person 類別的定義	題目提供原始碼
person.cpp	Person 類別的實作	需繳交
student.h	Student 類別的定義，需繼承 Person 類別，且所有**成員變數請放置於「protected」區塊**、所有成員函式請放置於「public」區塊，否則不予計分	需繳交
student.cpp	Student 類別的實作	需繳交
foreign_student.h	ForeignStudent 類別的定義，需繼承 Student 類別，且所有成員變數請放置於「private」區塊、所有成員函式請放置於「public」區塊，否則不予計分	需繳交
foreign_student.cpp	ForeignStudent 類別的實作	需繳交

請完成名為 **person.cpp、student.h、student.cpp、foreign_student.h** 與 **foreign_student.cpp** 的 C++ 程式。

本題的相關程式將使用以下的 Makefile 進行編譯：

```
all: main.cpp student.o foreign_student.o person.o
    c++ main.cpp student.o foreign_student.o person.o
foreign_student.o: foreign_student.cpp foreign_student.h student.h person.h
    c++ foreign_student.cpp -c
student.o: student.cpp student.h person.h
    c++ student.cpp -c
person.o: person.cpp person.h
    c++ person.cpp -c
clean:
```

```
    rm -f *.o *~ *.*~ a.out*
```

此題的執行結果可參考如下：

```
[3:23▲user@ws▲p1]./a.out↵
Student▲[Kennedy](cbb114901)↵
general▲performance:▲87.42↵
exam1:▲73.56↵
exam2:▲95.3↵
final▲exam:▲68.18↵
term▲score:▲81.1↵
-------------------------------------------↵
Foreign▲student▲[Juila](cbb113442,▲Korea▲(ROK)▲passport▲no.▲M98238646):▲↵
↵
general▲performance:▲100↵
exam1:▲73.56↵
exam2:▲95.3↵
final▲exam:▲68.18↵
term▲score:▲84.3↵
[3:23▲user@ws▲p1]
```



- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。
- 類別撰寫要求請參考上方表格
- 本題若有使用浮點數的需求，請使用 double 型態。
- 本題應繳交檔案如下：
  - person.cpp
  - student.h
  - student.cpp
  - foreign_student.h
  - foreign_student.cpp

p2 旅客類別實作VIP 旅客繼承一般旅客）

延續 cpp.hw8 的 p2本題仍是幫助鳳梨航空寄出關於航班起飛前相關資訊的 email 給旅客，並增加了 VIP 旅客類別。

Flight 類別相關定義請參考 flight.h 檔案：

```
#include <string>
using namespace std;

class Flight
{
```

```
protected:
    string number;
    string departure;
    string destination;
    string date;
    string boarding_time;
    string gate;

public:
    // setter
    void set_info(string flight_no, string from, string to, string date,
string time, string gate);

    // getter
    string get_flight_no();
    string get_departure();
    string get_destination();
    string get_date();
    string get_boarding_time();
    string get_gate();
};
```

Tourist 類別相關定義請參考 tourist.h 檔案：

```
#include "flight.h"

class Tourist
{
protected:
    string cabin_class;
    int luggage_limitation;
    struct Name
    {
        string last_name;
        string first_name;
    } name;

public:
    Flight flight;

    // setter
    void set_info(string first_name, string last_name, string cabin_class);

    // getter
    string get_first_name();
    string get_last_name();
    string get_cabin_class();
    int get_luggage_limitation();
```

```
};
```

請參考下列的 main.cpp 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include "vip_tourist.h"
#include <iostream>
using namespace std;

void show_email_content(Tourist tourist)
{
    cout << "Dear " << tourist.get_first_name() << " " <<
tourist.get_last_name() << "," << endl;
    cout << "Thank you for choosing Pineapple Airlines. We are excited to
welcome you on board for your upcoming flight to "
         << tourist.flight.get_destination() << ". Please find your flight
details below to ensure a smooth journey." << endl
         << endl;

    cout << "Flight Information: " << endl;
    cout << "\tFlight Number: " << tourist.flight.get_flight_no() << endl;
    cout << "\tFrom: " << tourist.flight.get_departure() << endl;
    cout << "\tTo: " << tourist.flight.get_destination() << endl;
    cout << "\tLuggage Limit: " << tourist.get_luggage_limitation() << endl;
    cout << "\tCabin Class: " << tourist.get_cabin_class() << endl;
    cout << "\tDate: " << tourist.flight.get_date() << endl;
    cout << "\tBoarding Time: " << tourist.flight.get_boarding_time() <<
endl;
    cout << "\tGate: " << tourist.flight.get_gate() << endl;
    cout << "Safe travels,\nThe Pineapple Airlines Team" << endl;
}

void show_email_content(VIPTourist tourist)
{
    string *special_treatment = tourist.get_special_privilege();
    cout << "Dear " << tourist.get_first_name() << " " <<
tourist.get_last_name() << "," << endl;
    cout << "Thank you for choosing Pineapple Airlines. We are excited to
welcome you on board for your upcoming flight to "
         << tourist.flight.get_destination() << ". Please find your flight
details below to ensure a smooth journey." << endl
         << endl;

    cout << "Flight Information: " << endl;
    cout << "\tFlight Number: " << tourist.flight.get_flight_no() << endl;
    cout << "\tFrom: " << tourist.flight.get_departure() << endl;
    cout << "\tTo: " << tourist.flight.get_destination() << endl;
    cout << "\tLuggage Limit: " << tourist.get_luggage_limitation() << endl;
    cout << "\tCabin Class: " << tourist.get_cabin_class() << endl;
    cout << "\tDate: " << tourist.flight.get_date() << endl;
```

```
    cout << "\tBoarding Time: " << tourist.flight.get_boarding_time() <<
endl;
    cout << "\tGate: " << tourist.flight.get_gate() << endl
         << endl;
    cout << "Special Privileges:" << endl;
    for (int i = 0; i < tourist.get_special_privilege_quantity(); i++)
    {
        cout << i + 1 << ". " << special_treatment[i] << endl;
    }
    cout << endl
         << "Safe travels,\nThe Pineapple Airlines Team" << endl;
}

int main()
{
    Tourist Alex, Jamie, Dennis;
    cout << "-----------------------------------------------------------------
---------" << endl;
    //          first name, last name, cabin class
    Alex.set_info("Alex", "Brookes", "Economy");
    //                  flight no.      from              to
mm/dd/yyyy   hr:mm   gate no.
    Alex.flight.set_info("PI 564", "Hong Kong(HKG)", "Taiwan Taoyuan(TPE)",
"9/27/2021", "18:45", "527");
    show_email_content(Alex);

    cout << "-----------------------------------------------------------------
---------" << endl;
    Jamie.set_info("Jamie", "Davis", "Business");
    Jamie.flight.set_info("PI 612", "Osaka (KIX)", "Seoul(ICN)",
"3/26/2026", "06:30", "7");
    show_email_content(Jamie);

    cout << "-----------------------------------------------------------------
---------" << endl;
    Dennis.set_info("Dennis", "Wilson", "First");
    Dennis.flight.set_info("PI 950", "Kaohsiung (KHH)", "Hong Kong(HKG)",
"1/30/2026", "10:00", "092");
    show_email_content(Dennis);

    cout << "-----------------------------------------------------------------
---------" << endl;
    VIPTourist Heinz;
    Heinz.set_info("Heinz", "Doofenshmirtz");
    Heinz.flight.set_info("PI 666", "Kaohsiung (KHH)", "Hong Kong(HKG)",
"1/30/2026", "10:00", "092");
    Heinz.add_special_privilege("We provide exclusive airport services, such
as priority check-in and lounge access.");
    Heinz.add_special_privilege("Seats are more spacious and can usually
```

```
recline into fully flat beds.");
    Heinz.add_special_privilege("We offer premium meals and beverages.");
    Heinz.add_special_privilege("Please enjoy more attentive and
personalized service.");
    show_email_content(Heinz);

    return 0;
}
```

鳳梨航空公司與其他的航空公司一樣有「economy class（經濟艙）」、「business class（商務艙）」以及「first class（頭等艙）」三個艙等。其中經濟艙的行李限制以 23 kg 為上限，其餘兩個艙等則以 32 kg 為上限。

請參考 69~75 行關於 VIPTouris 類別相關的操作，鳳梨航空為了提升飛行常客對於公司的品牌忠誠度設計了 VIP 制度。即旅客持有鳳梨航空 VIP 卡則可享受特殊禮遇，此外，其座艙艙等將無條件升級為頭等艙（行李亦額度亦調整為 32 kg 上限）。特殊禮遇則須視情況調整，例如：

- 鳳梨航空在機場設有貴賓室，因此 VIP 卡的旅客可以優先報到與免費使用貴賓室休息等侯登機（main.cpp 第 75 行）
- 該班次的機型頭等艙的座位能夠調整為平躺（main.cpp 第 76 行）
- 該班次提供高級餐食與飲料（main.cpp 第 77 行）

檔案名稱	**檔案用途與撰寫要求**	**-**
main.cpp	主程式	題目提供原始碼
flight.h	Flight 類別的定義	題目提供原始碼
flight.cpp	Flight 類別的實作	需繳交
tourist.h	Tourist 類別的定義	題目提供原始碼
tourist.cpp	Tourist 類別的實作	需繳交
vip_tourist.h	VIPTourist 類別的定義，需繼承 Tourist 類別，且所有成員變數請放置於「private」區塊、所有成員函式請放置於「public」區塊，否則不予計分	需繳交
vip_tourist.cpp	VIPTourist 類別的實作	需繳交

請完成名為 **flight.cpp**、**tourist.cpp**、**vip_tourist.h** 與 **vip_tourist.cpp** 的 C++ 程式，撰寫要求詳如上表。

本題的相關程式將使用以下的 Makefile 進行編譯：

```
all: main.cpp tourist.o flight.o vip_tourist.o
    c++ main.cpp tourist.o flight.o vip_tourist.o
vip_tourist.o: vip_tourist.cpp vip_tourist.h tourist.h flight.h
    c++ vip_tourist.cpp -c
tourist.o: tourist.cpp tourist.h flight.h
    c++ tourist.cpp -c
flight.o: flight.cpp flight.h
    c++ flight.cpp -c
clean:
    rm -f *.o *~ *.*~ a.out*
```

此題的執行結果可參考如下：

[3:23▲user@ws▲p2]./a.out↵
---↵
Dear▲Alex▲Brookes,↵
Thank▲you▲for▲choosing▲Pineapple▲Airlines.▲We▲are▲excited▲to▲welcome▲you▲on▲board▲for▲your▲upcoming▲flight▲to▲Taiwan▲Taoyuan(TPE).▲Please▲find▲your▲flight▲details▲below▲to▲ensure▲a▲smooth▲journey.↵
↵
Flight▲Information:▲↵
▲▲▲▲▲▲▲▲Flight▲Number:▲PI▲564↵
▲▲▲▲▲▲▲▲From:▲Hong▲Kong(HKG)↵
▲▲▲▲▲▲▲▲To:▲Taiwan▲Taoyuan(TPE)↵
▲▲▲▲▲▲▲▲Luggage▲Limit:▲23↵
▲▲▲▲▲▲▲▲Cabin▲Class:▲Economy↵
▲▲▲▲▲▲▲▲Date:▲9/27/2021↵
▲▲▲▲▲▲▲▲Boarding▲Time:▲18:45↵
▲▲▲▲▲▲▲▲Gate:▲527↵
Safe▲travels,↵
The▲Pineapple▲Airlines▲Team↵
---↵
Dear▲Jamie▲Davis,↵
Thank▲you▲for▲choosing▲Pineapple▲Airlines.▲We▲are▲excited▲to▲welcome▲you▲on▲board▲for▲your▲upcoming▲flight▲to▲Seoul(ICN).▲Please▲find▲your▲flight▲details▲below▲to▲ensure▲a▲smooth▲journey.↵
↵
Flight▲Information:▲↵
▲▲▲▲▲▲▲▲Flight▲Number:▲PI▲612↵
▲▲▲▲▲▲▲▲From:▲Osaka▲(KIX)↵
▲▲▲▲▲▲▲▲To:▲Seoul(ICN)↵
▲▲▲▲▲▲▲▲Luggage▲Limit:▲32↵
▲▲▲▲▲▲▲▲Cabin▲Class:▲Business↵
▲▲▲▲▲▲▲▲Date:▲3/26/2026↵
▲▲▲▲▲▲▲▲Boarding▲Time:▲06:30↵
▲▲▲▲▲▲▲▲Gate:▲7↵
Safe▲travels,↵
The▲Pineapple▲Airlines▲Team↵
---↵
Dear▲Dennis▲Wilson,↵
Thank▲you▲for▲choosing▲Pineapple▲Airlines.▲We▲are▲excited▲to▲welcome▲you▲on▲board▲for▲your▲upcoming▲flight▲to▲Hong▲Kong(HKG).▲Please▲find▲your▲flight▲details▲below▲to▲ensure▲a▲smooth▲journey.↵
↵
Flight▲Information:▲↵
▲▲▲▲▲▲▲▲Flight▲Number:▲PI▲950↵
▲▲▲▲▲▲▲▲From:▲Kaohsiung▲(KHH)↵
▲▲▲▲▲▲▲▲To:▲Hong▲Kong(HKG)↵
▲▲▲▲▲▲▲▲Luggage▲Limit:▲32↵
▲▲▲▲▲▲▲▲Cabin▲Class:▲First↵
▲▲▲▲▲▲▲▲Date:▲1/30/2026↵

▲▲▲▲▲▲▲▲Boarding▲Time:▲10:00↵
▲▲▲▲▲▲▲▲Gate:▲092↵
Safe▲travels,↵
The▲Pineapple▲Airlines▲Team↵
----------↵
Dear▲Heinz▲Doofenshmirtz,↵
Thank▲you▲for▲choosing▲Pineapple▲Airlines.▲We▲are▲excited▲to▲welcome▲you▲on▲board▲for▲your▲upcoming▲flight▲to▲Hong▲Kong(HKG).▲Please▲find▲your▲flight▲details▲below▲to▲ensure▲a▲smooth▲journey.↵
↵
Flight▲Information:▲↵
▲▲▲▲▲▲▲▲Flight▲Number:▲PI▲666↵
▲▲▲▲▲▲▲▲From:▲Kaohsiung▲(KHH)↵
▲▲▲▲▲▲▲▲To:▲Hong▲Kong(HKG)↵
▲▲▲▲▲▲▲▲Luggage▲Limit:▲32↵
▲▲▲▲▲▲▲▲Cabin▲Class:▲First↵
▲▲▲▲▲▲▲▲Date:▲1/30/2026↵
▲▲▲▲▲▲▲▲Boarding▲Time:▲10:00↵
▲▲▲▲▲▲▲▲Gate:▲092↵
↵
Special▲Privileges:↵
1.▲We▲provide▲exclusive▲airport▲services,▲such▲as▲priority▲check-in▲and▲lounge▲access.↵
2.▲Seats▲are▲more▲spacious▲and▲can▲usually▲recline▲into▲fully▲flat▲beds.↵
3.▲We▲offer▲premium▲meals▲and▲beverages.↵
4.▲Please▲enjoy▲more▲attentive▲and▲personalized▲service.↵
↵
Safe▲travels,↵
The▲Pineapple▲Airlines▲Team↵
[3:23▲user@ws▲p2]



- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。
- 類別撰寫要求請參考上方表格
- 本題若有使用浮點數的需求，請使用 double 型態。
- 本題應繳交檔案如下：
  - flight.cpp
  - tourist.cpp
  - vip_tourist.h
  - vip_tourist.cpp

p3 銀行帳戶類別實作（儲蓄帳戶繼承一般帳戶）

接續 cpp.hw8 的 p3本題新增了用來長期定存的儲蓄帳戶。

BankAccount 類別相關定義請參考下列的 bank.h 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include <iostream>
#include <string>
using namespace std;

class BankAccount
{
protected:
    string name;
    int balance;
    string transaction_record;
    int transaction_record_count = 1;

public:
    BankAccount(string name_str);
    string get_name();
    int get_balance();
    int get_transaction_record_count();
    void write_transaction_record(string record);
    void show_transaction_record();
    bool save(int amount);
    bool withdraw(int amount);
    bool transfer(int amount, BankAccount &recipient);
};
```

請參考下列的 main.cpp 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include "saving_account.h"
using namespace std;
#define SUCCESS "\e[0;32m[ SUCCESS ]\e[0m " // show "[ SUCCESS ] " with
green color
#define FAIL "\e[0;31m[   FAIL  ]\e[0m "    // show "[   FAIL  ] " with red
color

class Banking
{
public:
    void show_status(BankAccount a)
    {
        cout << a.get_name() << "\'s current balance: " << a.get_balance()
<< endl;
    }

    void save(BankAccount &a, int money)
    {
        string output;
        if (a.save(money))
        {
```

```
            output = SUCCESS + a.get_name() + " saved $" + to_string(money);
        }
        else
        {
            output = FAIL + a.get_name() + " failed to save $" +
to_string(money);
        }
        cout << output << endl;
a.write_transaction_record(to_string(a.get_transaction_record_count()) + " "
+ output + "\n");
    }

    void transfer(BankAccount &a, BankAccount &b, int money)
    {
        string output;
        if (a.transfer(money, b))
        {
            output = SUCCESS + a.get_name() + " transfered $" +
to_string(money) + " to " + b.get_name();
        }
        else
        {
            output = FAIL + a.get_name() + " failed to transfer $" +
to_string(money) + " to " + b.get_name();
        }
        cout << output << endl;
a.write_transaction_record(to_string(a.get_transaction_record_count()) + " "
+ output + "\n");
    }

    void withdraw(BankAccount &a, int money)
    {
        string output;
        if (a.withdraw(money))
        {
            output = SUCCESS + a.get_name() + " withdrew $" +
to_string(money);
        }
        else
        {
            output = FAIL + a.get_name() + " failed to withdraw $" +
to_string(money);
        }
        cout << output << endl;
a.write_transaction_record(to_string(a.get_transaction_record_count()) + " "
+ output + "\n");
    }
};

int main()
{
```

```
    Banking bank;
    BankAccount Justin("Justin");
    BankAccount Amy("Amy");
    SavingAccount Phineas("Phineas");

    cout << "[1] ------------------------" << endl;
    bank.show_status(Justin);
    bank.show_status(Amy);

    cout << "[2] ------------------------" << endl;
    bank.save(Amy, 4090);
    bank.save(Amy, 0);
    bank.save(Amy, -216);

    cout << "[3] ------------------------" << endl;
    bank.transfer(Amy, Justin, 1080);
    bank.transfer(Amy, Justin, 4050);
    bank.transfer(Amy, Justin, 0);

    cout << "[4] ------------------------" << endl;
    bank.withdraw(Amy, 12000);
    bank.withdraw(Amy, 3000);
    bank.withdraw(Amy, 0);

    cout << "[5] ------------------------" << endl;
    bank.show_status(Amy);
    bank.show_status(Justin);

    cout << "[6] ------------------------" << endl;
    Amy.show_transaction_record();

    cout << "[7] ------------------------" << endl;
    bank.show_status(Phineas);
    bank.save(Phineas, 10000);
    bank.withdraw(Phineas, 10);
    bank.show_status(Phineas);
    Phineas.calculate_interest(10); // 10 years
    bank.show_status(Phineas);

    return 0;
}
```

SavingAccount 繼承自 BankAccount，並增加了一個成員函式 calculate_interest，此函式以年為單位且年利率為 1.6%，用來計算一段時間的本利和，並直接覆蓋當前存款金額。請參考本利和的複利公式：

$$ 期末本利和 = 期初本金金額 \times (1 + 每期利率)^{期數} $$

計算結果請無條件捨棄小數。

檔案名稱	**檔案用途與撰寫要求**	**-**
main.cpp	主程式	題目提供原始碼
bank.h	BankAccount 類別的定義	題目提供原始碼
bank.cpp	BankAccount 類別的實作	需繳交
saving_account.h	SavingAccount 類別的定義，需繼承 BankAccount 類別，且所有成員變數請放置於「private」區塊、所有成員函式請放置於「public」區塊，否則不予計分	需繳交
saving_account.cpp	SavingAccount 類別的實作	需繳交

請完成名為 **bank.cpp、saving_account.h** 與 **saving_account.cpp** 的 C++ 語言程式，相關檔案撰寫要求已詳述於上方表格。

本題將使用以下的 Makefile 進行編譯：

```
all: main.cpp bank.o saving_account.o bank.h saving_account.h
    c++ main.cpp bank.o saving_account.o
saving_account.o: bank.h saving_account.cpp saving_account.h
    c++ saving_account.cpp -c
bank.o: bank.cpp bank.h
    c++ bank.cpp -c
clean:
    rm -f *.o *~ *.*~ a.out*
```

此題的執行結果可參考如下：

```
[3:23 user@ws p3]./a.out
[1] ----------
Justin's current balance: 0
Amy's current balance: 0
[2] ----------
[ SUCCESS ] Amy saved \$4090
[ FAIL ] Amy failed to save \$0
[ FAIL ] Amy failed to save \$-216
[3] ----------
[ SUCCESS ] Amy transfered \$1080 to Justin
[ FAIL ] Amy failed to transfer \$4050 to Justin
[ FAIL ] Amy failed to transfer \$0 to Justin
[4] ----------
[ FAIL ] Amy failed to withdraw \$12000
[ SUCCESS ] Amy withdrew \$3000
[ FAIL ] Amy failed to withdraw \$0
[5] ----------
Amy's current balance: 10
Justin's current balance: 1080
[6] ----------
```

```
Amy's▲transaction▲record:▲↵
1▲[▲SUCCESS▲]▲Amy▲saved▲\$4090↵
2▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲save▲\$0↵
3▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲save▲\$-216↵
4▲[▲SUCCESS▲]▲Amy▲transfered▲\$1080▲to▲Justin↵
5▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲transfer▲\$4050▲to▲Justin↵
6▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲transfer▲\$0▲to▲Justin↵
7▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲withdraw▲\$12000↵
8▲[▲SUCCESS▲]▲Amy▲withdrew▲\$3000↵
9▲[▲▲▲FAIL▲▲]▲Amy▲failed▲to▲withdraw▲\$0↵
[7]▲----------------------↵
Phineas's▲current▲balance:▲0↵
[▲SUCCESS▲]▲Phineas▲saved▲\$10000↵
[▲SUCCESS▲]▲Phineas▲withdrew▲\$10↵
Phineas's▲current▲balance:▲9990↵
Phineas's▲current▲balance:▲11708↵
[3:23▲user@ws▲p3]
```



- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。
- 類別撰寫要求請參考上方表格
- 本題應繳交檔案如下（至於 main.cpp 與 Makefile 則不需繳交）：
  - bank.cpp
  - saving_account.h
  - saving_account.cpp

p4 學生 + 員工 = 工讀生（多重繼承）

本題需要透過學生（Student）與員工（Employee）的類別定義多重繼承工讀生（StudentEmployee）類別。

學生（Student）類別相關定義請參考下列的 student.h 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include <string>
using namespace std;
class Student
{
protected:
    string name;
    string student_id;
    string school_department;
    char grade;

public:
    void set_name(string name);
```

```
    void set_student_id(string id);
    void set_school_department(string department);
    void set_grade(char c);

    string get_name();
    string get_student_id();
    string get_school_department();
    char get_grade();
};
```

員工Employee類別相關定義請參考下列的 employee.h 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include <string>
using namespace std;

class Employee
{
protected:
    string name;
    string employee_id;
    string department;
    string job_title;
    int daily_rate;
    int attemdance_days;
    int salary;
public:
    void set_name(string name);
    void set_employee_id(string id);
    void set_department(string department);
    void set_job_title(string job_title);
    void set_daily_rate(int daily_rate);
    void set_attemdance_days(int days);

    void calculate_pay();

    string get_name();
    string get_employee_id();
    string get_department();
    string get_job_title();
    int get_salary();
};
```

請參考下列的 main.cpp 程式（請依據檔名旁的路徑至 ws 取得程式碼）：

```
#include "student_employee.h"
#include <iostream>
using namespace std;
```

```
int main()
{
    Student Renata;
    Employee Mick;
    StudentEmployee Julia;

    Renata.set_name("Renata");
    Renata.set_school_department("CSIE");
    Renata.set_student_id("cbb114901");
    Renata.set_grade('a');
    cout << Renata.get_student_id() << "(" << Renata.get_name() << ") "
         << ", Department: " << Renata.get_school_department() << ", Grade:
" << Renata.get_grade() << endl;

    Mick.set_name("Mick");
    Mick.set_employee_id("892314");
    Mick.set_department("Frontend");
    Mick.set_job_title("Engineer");
    Mick.set_attemdance_days(22);
    Mick.set_daily_rate(1800);
    Mick.calculate_pay();
    cout << Mick.get_name() << "(" << Mick.get_employee_id() << ", "
         << Mick.get_department() << " " << Mick.get_job_title() << ")
Salary: " << Mick.get_salary() << endl;

    Julia.Student::set_name("Julia");
    Julia.set_school_department("CSIE");
    Julia.set_student_id("cbb114902");
    Julia.set_grade('b');
    Julia.set_employee_id("809345");
    Julia.set_department("Library");
    Julia.set_work_hour(32);
    Julia.set_hourly_rate(196);
    Julia.calculate_pay();
    cout << Julia.get_student_id() << "(" << Julia.Student::get_name() <<
"), Department: "
         << Julia.get_school_department() << ", Grade: " <<
Julia.get_grade()
         << ", employee id: " << Julia.get_employee_id() << "(" <<
Julia.get_department()
         << ") Salary: " << Julia.get_salary() << endl;
    Julia.set_grade('c');
    cout << Julia.get_student_id() << "(" << Julia.Student::get_name() <<
"), Department: "
         << Julia.get_school_department() << ", Grade: " <<
Julia.get_grade()
         << ", employee id: " << Julia.get_employee_id() << "(" <<
Julia.get_department()
         << ") Salary: " << Julia.get_salary() << endl;
```

```
    Julia.set_grade('A');
    cout << Julia.get_student_id() << "(" << Julia.Student::get_name() <<
"), Department: "
         << Julia.get_school_department() << ", Grade: " <<
Julia.get_grade()
         << ", employee id: " << Julia.get_employee_id() << "(" <<
Julia.get_department()
         << ") Salary: " << Julia.get_salary() << endl;

    return 0;
}
```

有關於薪水計算：

1. 員工薪水的計算方式為「$出勤天數 \times 日薪$」，出勤天數請參考 main.cpp 第 22 行；日薪請參考 main.cpp 第 23 行。
2. 工讀生薪水的計算方式為「$工作時數 \times 時薪$」，工作時數請參考 main.cpp 第 34 行；日薪請參考 main.cpp 第 35 行。

鳳梨大學對學生成為校內工讀生有成績上的要求，學生的成績必須是 'B'（含）以上。為簡化起見，當學生的成績未符合工讀生的條件，其薪資輸出為 0。

檔案名稱	**檔案用途與撰寫要求**	**-**
main.cpp	主程式	題目提供原始碼
student.h	Student 類別的定義	題目提供原始碼
student.cpp	Student 類別的實作	需繳交
employee.h	Employee 類別的定義	題目提供原始碼
employee.cpp	Employee 類別的實作	需繳交
student_employee.h	StudentEmployee 類別的定義，需**多重繼承 Student 以及 Employee 類別**，且所有成員變數請放置於「private」區塊、所有成員函式請放置於「public」區塊，否則不予計分	需繳交
student_employee.cpp	StudentEmployee 類別的實作	需繳交

請完成名為 **student.cpp**、**employee.cpp**、**student_employee.h**與 **student_employee.cpp** 的 C++ 語言程式，各檔案撰寫要求已詳述於上方表格。

本題將使用以下的 Makefile 進行編譯：

```
all: main.cpp student.o employee.o student_employee.o
    c++ main.cpp student.o employee.o student_employee.o
student_employee.o: student_employee.cpp student_employee.h student.h
employee.h
    c++ student_employee.cpp -c
employee.o: employee.cpp employee.h
    c++ employee.cpp -c
student.o: student.cpp student.h
```

```
    c++ student.cpp -c
clean:
    rm -f *.o *~ *.*~ a.out*
```

此題的執行結果可參考如下：

```
[3:23▲user@ws▲p4]./a.out↵
cbb114901(Renata)▲,▲Department:▲CSIE,▲Grade:▲A↵
Mick(892314,▲Frontend▲Engineer)▲Salary:▲39600↵
cbb114902(Julia),▲Department:▲CSIE,▲Grade:▲B,▲employee▲id:▲809345(Library)▲Salary:▲6272↵
cbb114902(Julia),▲Department:▲CSIE,▲Grade:▲C,▲employee▲id:▲809345(Library)▲Salary:▲0↵
cbb114902(Julia),▲Department:▲CSIE,▲Grade:▲A,▲employee▲id:▲809345(Library)▲Salary:▲6272↵
[3:23▲user@ws▲p4]
```



- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。
- 類別撰寫要求請參考上方表格
- 本題應繳交檔案如下：
  - student.cpp
  - employee.cpp
  - student_employee.h
  - student_employee.cpp

From:
https://junwu.nptu.edu.tw/dokuwiki/ - **Jun Wu的教學網頁**
國立屏東大學資訊工程學系
CSIE, NPTU
Total: 343751

Permanent link:
https://junwu.nptu.edu.tw/dokuwiki/doku.php?id=cpp:2026spring:hw9

Last update: **2026/05/20 14:51**