Object-Oriented Programming (UEE1303)

一、簡答題

小數字 第二個快大數字 第一個快

(1) (5%) 請問 Big-O 的複雜度函數代表什麼意義? 如果有兩個演算法, 第一個複雜度是 $1000N^2$, 第二個是 $2N^3$, 請問哪一個的時間複雜度較佳? 所有情況下都會比較快嗎?

(2) (5%) 請問在標頭檔(header file)當中使用#ifndef 的目的為何? 在編譯器之中, 他又會如何處理#ifndef ?

(3) (5%) 當你將程式碼放在不同檔案去做編譯的時候, 請問要如何在檔案間分享變數與函數? 如果你不希望某些變數及函數被別的檔案存取, 又要怎麼做設定呢?

(4) (5%) 請問使用物件繼承(inheritance)的好處為何? 哪些東西可以從 base class 繼承, 哪些東西不能繼承呢? 如果有東西不能繼承, 請解釋為何不能繼承.

(5) (5%) 請問在物件繼承的時候, 使用 redefined function 跟 virtual function 有甚麼差別? 請舉例說明.

(6) (5%) 在一個 class 裡面, destructor 是一個很特別的 member function, 請問它最主要的作用為何? 在繼承物件的時候, destructor 有需要做什麼特別的處理嗎??

(7) (5%) 請問在使用 template function 的時候, 任何一種 data type 都可以套用嗎? 有沒有什麼先備條件或是不能使用的狀況? 請舉例說明你的理由.

(8) (5%) 請問使用 void pointer 與 iterator 有何差別?

(9) (5%) 請問使用 vector 與 list 的差異在哪? 甚麼時候用 vector 比較好? 甚麼時候改用 list 會比較好呢?

二、(30%) 假設有一個 class 命名為 Hand, 裡面包含了五張牌, 每張牌需至少包含兩個變數, 一個整數存點數(1-13), 一個字串存花色(Spade, Heart, Club, Diamond), 牌的紀錄方式請自行設計, 但建議用 vector 把五張牌集合起來. 請依照下列敘述完成指定的部分:

(1) 請設計一個 class Hand, 至少需包含一個 default constructor, 以及另一個可自行輸入五張牌內容的 constructor (輸入方式請自行設計), 同時設計一個 printout 的函數, 可以把五張牌的內容印出來. 請將 class Hand 的宣告及這三個 member function 的程式碼都寫出來. (8%)

(2) 請設計另一個 class 作為衍生類別(derived class), 命名為 Pair, 用來儲存手牌中只有一對相同的狀況, 除了繼承 class Hand 之外, 另外儲存一個整數, 紀錄是哪一張牌(1-13)湊成一對. 請將你設計的 class Pair 宣告完整的寫出來. (3%)

(3) 承(2), 請修改 class Pair 的兩個 constructor function, 讓他可以正確的設定 Pair 的初始值, 如果可以, 請盡量呼叫原本 class Hand 的 constructor. 另外, printout 這個 function 也要做修改, 會先印出相同的兩張牌, 之後再印其他三張牌. 請將修改後的程式碼寫出來. (5%)

(4) 請設計另一個 class 作為衍生類別, 命名為 Straight, 用來儲存手牌中的點數是相連的五個數字的狀況(不限花色), 除了繼承 class Hand 之外, 另外儲存一個整數, 紀錄這五張牌最小的點數為何. 除了將你設計的 class Straight 宣告完整的寫出來之外, 請把修改後的 printout function 程式碼寫出來, 它可以呼叫 algorithm 裡面的 function 做排序, 讓輸出的牌可以按點數從小印到大. (6%)

(5) 請在 base class 及兩個 derived class 增加一個 virtual function, 命名為 check, 可以檢查目前的五張牌是否符合該物件的規則. 請寫出修改後的三個 class 宣告及三個不同版本的 function check 程式碼. (8%)

Object-Oriented Programming (UEE1303)

三、(11%) 標準的函式庫當中提供的絕對值函數有分三種, abs 是處理整數的, labs 是處理長整數, fabs 是處理浮點數, 使用者必須自行切換不同函數, 不是很方便.

(1) 請設計一個 template function, 命名為 absolute, 不管輸入上述這三種數字的任何一種, 都可以回傳正確的絕對值, 並把函數的宣告及函數的程式碼都寫出來. (5%)

(2) 請試著擴充(1)的 function, 就算你輸入的是 vector (內容是上述三種數字型別的其中一種), 也可以把每個 element 的絕對值都算出來存回 vector 之中, 請把修改後的函數程式碼寫出來. 舉例來說, 假設 a 為 int, v 為 vector<int>, v 實際用到的長度為 size, 呼叫的方式為 absolute(a)跟 absolute(v, size). (6%)

四、(8%) 假設有一個儲存容器(container), 它的 key 是一個字串, 代表文章中的一個單字(word), 在原始的設定中, set/map 中的 key 是不能重複的, 但是在一篇文章之中, 同一個單字可能會出現很多次, 可能沒辦法直接使用內建的功能來記錄. 請試著利用 set or map 設計一個程式, 可以從檔案讀取一篇文章, 並記錄文章中出現的單字, 如果出現超過一次以上, 也能夠記錄出現的次數. 最後並設計一個 printout 的函數, 把容器內的所有單字及出現的次數印出來.

五、(6%) 請寫一個簡單的程式, 利用 C++11 的 regular expression, 來檢查使用者輸入的字串是否符合 MM/DD/YYYY 的格式, 如果格式不對, 請印出錯誤訊息. MM, DD 分別代表月份與日期, 一定得是兩位數字, 如果是一月一號, 應表示成 01/01, MM 不能超過 12, 而 DD 不能超過 31(不考慮大月小月及閏年), YYYY 則表示西元年份, 範圍從 1900 到今年.

Reference Solution for the Final Exam of Object-Oriented Programming

(一)

(1) (5%) 請問Big-O的複雜度函數代表什麼意義?(2) 如果有兩個演算法，第一個複雜度是1000N^2，第二個是2N^3，請問哪一個的時間複雜度較佳? (1) 所有情況下都會比較快嗎? (2)

Sol:

(i) BigO就是描述演算法複雜度上界的漸進符號，當一個演算法「實際」的複雜度(或執行成本對輸入資料量函數)為f(n)時，想要用BigO描述其複雜度上界時，必須滿足以下定義：

$$f(n) = O(g(n)) : \exists k > 0\ \exists n_0\ \forall n > n_0\ |f(n)| \leq k \cdot g(n)$$

假設有一演算法實際複雜度為 $f(n) = 3n + 4$，有一組 $k = 4;\ g(n) = n;\ n_0 = 4$ 滿足

$$\forall n > 4,\ 0 \leq f(n) = 3n + 4 \leq 4n$$

(ii) 1000N^2

(iii) 不是，在N不夠大的時候，O(2N^3) 會比較快

(2) (5%) 請問在標頭檔 (header file)當中使用 #ifndef 的目的為何 ? (3) 在編譯器之中，他又會如何處理 #ifndef ? (2)

Sol:

條件指示符#ifndef 的最主要目的是防止標頭檔案的重複包含和編譯。它的功能是，如果識別符號未被#define 命令定義過則對#ifndef 包裹住的程式段進行編譯，否則就跳過該程式段不做編譯。

(3) (5%) 當你將程式碼放在不同檔案去做編譯的時候，請問要如何在檔案間分享變數與函數? (2) 如果你不希望某些變數及函數被別的檔案存取，又要怎麼做設定呢? (3)

Sol:

在一個檔案定義變數後，在其他檔案宣告時使用keyword: extern。若是函數則不需另加extern，在檔案前端重做一次函數宣告即可使用。

若不希望變數及函數被其他檔案存取，可將之放進unnamed namespace。

(4) (5%) 請問使用物件繼承(inheritance)的好處為何? (2) 哪些東西可以從base class繼承，哪些東西不能繼承呢? (1) 如果有東西不能繼承，請解釋為何不能繼承. (2)

Sol:

(i) 繼承可以使用原物件裡相同的概念及變數，而不需要重新宣告

assignment operator、destructor
copy constructor, private data

(ii) 所有base class的member variables跟functions都能被繼承，而copy constructors, the assignment operator, destructors還有private data不能被繼承

(iii) copy constructors, the assignment operator, destructors是因為可能會牽涉到指標及動態記憶體，直接複製指標恐怕會影響其他物件，故C++從語法上直接禁止繼承；而private data則是因為保護等級的設定，要設定成protected才能繼承

statically bound　　dynamic bounded

(5) (5%) 請問在物件繼承的時候，使用 redefined function 跟 virtual function有甚麼差別？請舉例說明.

Compile就決定了　　執行才決定呼叫的版本

Sol:

redefined function為statically bound(bound at compile time)，在編譯時就已經決定要採用哪一個版本的function，若有重複的版本會被覆蓋；而virtual function則是dynamically bound(bound at runtime)，在程式執行中才會依照呼叫的物件型態來決定採用的版本，比較能應付各種不同的狀況。舉例來說，當一個derived class D要更改non-virtual function mf定義的時候，還是可以找到base class原本的mf，而 derived class D要更改virtual member function mf inherited from class B的時候 會 覆寫原本B的mf。

(6) (5%) 在一個 class 裡面，destructor 是一個很特別的 member function, 請問它最主要的作用為何？(3) 在繼承物件的時候，destructor 有需要做什麼特別的處理嗎 ?? (2)

Sol:

Destructor可以進行Class在使用後的善後工作，如清除動態記憶體配置等。

一般來說，Destructor不會被繼承，若在Derived Class中沒有宣告，會自動生成一個空的default版本，因此，若Derived Class中有指標或動態記憶體配置時，就必須要在Derived Class中重新定義中重新定義Destructor，若有複雜的繼承關係，應考慮使用virtual function來設計destructor，確保程式在每一種情況下都能呼叫到正確的版本來清除動態記憶體。

(7) (5%) 請問在使用 template function的時候，任何一種 data type都可以套用嗎？(2) 有沒有什麼先備條件或是不能使用的狀況？請舉例說明你的理由(3)

Sol:

其中一個例子：並非所有的template function都能處理任意的data type，ex.一個排序的template function，若未事先定義如何排序class的data，就將Class丟進去做比較，那程式就會出問題。

(8) (5%) 請問使用void pointer與iterator有何差別?

Sol:

Void pointer: 基本上還是一個指標變數，可存放記憶體位置，可以指向任意型態的變數，但必須設定好型別之後才能透過dereference來使用，也可以delete所指的記憶體空間

Iterator: 基本上是一個物件，只是透過operator overloading的設計，讓它可以當成pointer使用，它也可以用在任意型態的變數上，但不同型態結構的container就有不同的 iterator，另外，iterator不能delete所指向的變數，只能單純做存取的動作

(9) (5%) 請問使用vector與list的差異在哪? 甚麼時候用vector比較好? 甚麼時候改用list會比較好呢?

Sol:

Vector: 可以存取任意一個位置的變數，但是增加資料只能從最後面加入，要從中間刪除一筆資料也比較不方便

List: 沒辦法透過index快速定位某個特定變數，只能從頭慢慢找起，但是要增加資料或是刪除資料很方便，基本上任意一個位置都能做加入或刪除的動作

需要任意取值時用vector

當取值時有按照前後順序時用list，插入跟刪除的時候速度會較快

(二) 假設有一個 class命名為Hand, 裡面包含了五張牌,每張牌需至少包含兩個變數, 一個整數存點數 (1-13), 一個字串存花色(Spade, Heart, Club, Diamond), 牌的紀錄方式請自行設計, 但建議用vector把五張牌集合起來. 請依照下列敘述完成指定的部分:

(1) 請設計一個class Hand, 至少需包含一個default constructor, 以及另一個可自行輸入五張牌內容的 constructor (輸入方式請自行設計), 同時設計一個printout的函數 , 可以把五張牌的內容印出來. 請將 class Hand的宣告及這三個member function的程式碼都寫出來. (8%)

Sol:

```
class Hand{
 public:
        Hand();
        Hand(vector<int> in_vi,vector<string> in_vs);
        void printout();
        virtual bool check();
 protected:
        vector<int> vi;
        vector<string> vs;
};
Hand::Hand(){
 int temp=0;
```

```
 string temps="noinputs";
 for(int i=0;i<5;i++){
        vi.push_back(temp);
        vs.push_back(temps);
 }
}
Hand::Hand(vector<int> in_vi,vector<string> in_vs){
 for(int i=0;i<5;i++){
        vi.push_back(in_vi);
        vs.push_back(in_vs);
 }
}
void Hand::printout(){
 for(int i=0;i<5;i++){
        cout<<vi[i]<<" "<<vs[i]<<endl;
 }
}
```

(2) 請設計另一個class作為衍生類別(derived class), 命名為Pair, 用來儲存手牌中只有一對相同的狀況, 除了繼承class Hand之外, 另外儲存一個整數, 紀錄是哪一張牌(1-13)湊成一對. 請將你設計的class Pair宣告完整的寫出來. (3%)

Sol:

```
class Pair():: public Hand{
 public:
        Pair();
        Pair(vector<int> in_vi,vector<string> in_vs);
        void printout();
bool check();
protected:
 int same;
}
```

(3) 承(2), 請修改class Pair的兩個constructor function, 讓他可以正確的設定Pair的初始值, 如果可以, 請盡量呼叫原本class Hand的constructor. 另外 , printout這個function也要做修改,會先印出相同的兩張牌, 之後再印其他三張牌. 請將修改後的程式碼寫出來. (5%)

Sol:

```
Pair::Pair(){
 Hand();
 same=0;
}

Pair:: Pair(vector<int> in_vi,vector<string> in_vs){
 Hand(int in_vi;string in_vs);
 for(int i=0;i<5;i++)
        for(int j=0;j<5;j++)
               if(in_vi[i]==in_vi[j])
                      same=in_vi[i];
```

```
}

void Pair::printout(){
 for(int i=0;i<5;i++)
        if(vi[i]==same)
                cout<<vi[i]<<" "<<vs[i]<<endl;
 for(int i=0;i<5;i++)
        if(vi[i]!=same)
                cout<<vi[i]<<" "<<vs[i]<<endl;
}
```

(4) 請設計另一個class作為衍生類別，命名為Straight，用來儲存手牌中的點數是相連的五個數字的狀況（不限花色），除了繼承class Hand之外，另外儲存一個整數，紀錄這五張牌最小的點數為何．除了將你設計的class Straight宣告完整的寫出來之外，請把修改後的printout function程式碼寫出來，它可以呼叫algorithm裡面的function做排序，讓輸出的牌可以按點數從小印到大．(6%)

Sol:

```
Class Straight::public Hand(){
 public:
        Straight();
        Straight(vector<int> in_vi,vector<string> in_vs);
        bool check();
protected:
        void printout();
        int smallest;

}
void Straight::printout(){
 vector<int> temp;
 for(int i=0;i<5;i++)
        temp.push_back(vi[i]);
 sort(temp,temp+5);
 for(int i=0;i<5;i++)
        for(int j=0;j<5;j++)
                if(temp[i]==temp[j])
                        cout<<vi[j]<<" "<<vs[j]<<endl;
}
```

(5) 請在base class及兩個 derived class增加一個 virtual function，命名為 check，可以檢查目前的五張牌是否符合該物件的規則．請寫出修改後的三個 class宣告及三個不同版本的function check程式碼．(8%)

Sol:

```
bool Hand::check(){
 for(int i=0;i<5;i++){
        if(vi[i]<1||vi[i]>13)
                return 0;
        if(vs[i]!="Spade"&& vs[i]!="Heart"&&vs[i]!="Club"&& vs[i]!="Diamond")
```

```
            return 0;
        return 1;
  }
}

bool Pair::check(){
  if(!Hand::check()) return 0;
  int onlyonepair;
  for(int i=0;i<4;i++){
        for(int j=i+1;j<5;j++)
            if(vi[i]==vi[j])
                onlyonepair++;
  }
  if(onlyonepair==1) return 1;
  else return 0;
}

bool Straight::check(){
  If(!Hand::check()) return 0;
  vector<int> temp;
  for(int i=0;i<5;i++)
        temp.push_back(vi[i]);
  sort(temp,temp+5);
  for(int i;i<4;i++)
        if(temp[i+1]!=temp[i]+1)
            return 0;
  return 1;
}
```

(三) (11%) 標準的函式庫當中提供的絕對值函數有分三種, abs是處理整數的, labs是處理長整數, fabs是處理浮點數, 使用者必須自行切換不同函數, 不是很方便.

(1) 請設計一個 template function, 命名為 absolute, 不管輸入上述這三種數字的任何一種 , 都可以回傳正確的 絕對值, 並把函數的宣告及函數的程式碼都寫出來. (5%)

Sol:

```
template <class T>
T absolute(T x){
        if (x<0) return -x;
        else return x;
  }
```

(2) 請試著擴充(1)的function, 就算你輸入的是vector (內容是上述三種數字型別的其中一種), 也可以把每個element的絕對值都算出來存回vector之中, 請把修改後的函數程式碼寫出來. 舉例來說, 假設a為int, v為vector<int>, v實際用到的長度為size, 呼叫的方式為absolute(a)跟absolute(v, size). (6%)

Sol:

```
template <class T>
T absolute(T x){
  if (x<0) return -x;
  else return x;
}
vector<T> absolute(vector<T> vec, int size){
  for(int i = 0; i < size; ++i){
        vec[i] = absolute(vec[i]);
  }
  return vec;
}
```

(四) (8%) 假設有一個儲存容器(container), 它的key是一個字串, 代表文章中的一個單字(word), 在原始的設定中, set/map中的key是不能重複的, 但是在一篇文章之中, 同一個單字可能會出現很多次, 可能沒辦法直接使用內建的功能來記錄. 請試著利用set or map設計一個程式, 可以從檔案讀取一篇文章, 並記錄文章中出現的單字, 如果出現超過一次以上, 也能夠記錄出現的次數. 最後並設計一個printout的函數, 把容器內的所有單字及出現的次數印出來.

重點: find !!

Sol:

```
int main(int argc, char** argv)
{
        ifstream fin;
        fin.open("article.txt");

        map<string, unsigned int> mymap;
        map<string, unsigned int>::iterator it;

        string buffer;
        while(fin >> buffer)
        {
                it = mymap.find(buffer);
                if(it != mymap.end())
                        ++it->second;
                else
                        mymap[buffer] = 1;
        }

        printout(mymap);

        fin.close();
        return 0;
}

void printout(const map<string, unsigned int>& mymap)
{
        for(const auto& i: mymap)
                cout << i.first << " : " << i.second << endl;
}
```

(五) (6%) 請寫一個簡單的程式, 利用C++11的regular expression, 來檢查使用者輸

入的字串是否符合MM/DD/YYYY的格式, 如果格式不對, 請印出錯誤訊息. MM, DD分別代表月份與日期, 一定得是兩位數字, 如果是一月一號, 應表示成01/01, MM不能超過12, 而DD不能超過31(不考慮大月小月及閏年), YYYY則表示西元年份, 範圍從1900到今年.

Sol: 以下兩種答案皆算對

(i)

```
int main(int argc, char** argv)
{
        string str;
        regex reg("(0[1-9]|1[0-2])/(0[1-9]|[1-2]\\d|3[0-1])/(19\\d{2}|20[0-1]\\d|202[0-1])");

        cin >> str;
        if(!regex_match(str, reg))
                cout << "WRONG FORMAT OF DATE" << endl;

        return 0;
}
```

(ii)

```
int main(int argc, char** argv)
{
        string str;
        regex reg("\\d{2}/\\d{2}/\\d{4}");

        cin >> str;
        if(!regex_match(str, reg))
                cout << "WRONG FORMAT OF DATE" << endl;
        else
        {
                int month, day, year;
                month = stoi(str.substr(0,2));
                day = stoi(str.substr(3,2));
                year = stoi(str.substr(6,4));

                if(month < 1 || month > 12 || day < 1 || day > 31 || year < 1900 || year > 2021)
                        cout << "WRONG FORMAT OF DATE" << endl;
        }

        return 0;
}
```