台大資工訓練班 C/C++基礎班

期末考

- Internet , toilet or filling the water are always allowed.
- **Plagiarism and discussion is strictly prohibited.**
- Your final score = The score in this exam * 0.5
  +Score of assignments * 0.3
  +Score of attendance (maximum 20)
- 70 is the threshold to get the final certification. The full marks is 600 scores. If you attended all the class and submitted all the assignments, 40 scores in this exam is enough for you to get the certification.
- You can ask for explaining the meaning of the problem, but no instruction on how to solve it. If you cannot connect to the Internet, tell me and I will transmit the data needed in this final exam to you by USB.
- Once you finish your exam, you can hand in your solution before time. If you would like to know the score / pass or not immediately, please stay in the classroom after 12:30

- Handing Method：
  - Email me the result to the following address.
    lkm543@hotmail.com
  - Please name the title of the email by [279][c++][Name]
    [279][c++][王小明]
  - Please attach your codes(.cpp) and DO NOT attach .exe file
  - You can also attach a Readme File (txt) to tell me how to execute your code, or what difficulty you are faced with (Optional)
- I need about 1 week to revise all the codes you submit, please wait me for a while. The official will inform you pass or not later.

- I feel sorry about that I cannot customize the lecture to everybody. So I adjust the exercise of final exam. This final exam comprise 6 problems, you can choose the adequate problems for you. So, each problem has its own motivation to make it more than an exam.

Problem 1&2

- Ideal candidate：
  - Students who are not major in CS or still senior high school students.
  - Laymen who are first time to coding.
  - The lecture before seems a little challenging to you.
- Motivation：
  - Help you review what you learned in the past 11 classes.
  - Make you be able to learn other coding language or deeper c++ skills.
  - The mission can be completed by following the hint.

Problem 3 & 4

- Ideal candidate：
  - Students who have the experiences of coding before this class.
  - The lecture before seems a piece of cake to you.
  - Students who like to brainstorm.
- Motivation：
  - Help you wreck one's brain by exercise.
  - Make you have the slight knowledge of Google Code Jam, ACM, or algorithms.

Problem 5 & 6

- Ideal candidate：
  - Students who work as an engineer, especially the software engineer.
  - Students who would like to be an engineer in the future.
- Motivation：
  - Make you have the slight knowledge of Data Structure and Algorithm (DSA).
  - Make you know how to develop and regulate a program.

● Problem 1 (100%) ***Basic syntax(1)***

➢ Problem 1-1 (20%)

- ✧ **Key words**：File In and Array
- ✧ **Mission**：

  Download the Test Data：http://ppt.cc/a7lX6

  Read the data each line, show it on the screen and store each element in a 2D array, which will be used later.
- ✧ **Hint**：

  It will looks like a 2D array like this array[M][N]

  The example code is in Appendix, change the filename.

➢ Problem 1-2 (20%)

- ✧ **Key words**：for/while/function
- ✧ **Mission**：

  Print the average value of each line.
- ✧ **Hint**：

  for (int i=0; i<len;i++)……

➢ Problem 1-3 (30%)

- ✧ **Key words**：for/while/function
- ✧ **Mission**：

  Print the Standard Derivation(SD) of each line.
- ✧ **Hint**：

  for (int i=0; i<len;i++)……

  #include <math.h>

  a^b=pow(a,b);

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

➢ Problem 1-4 (30%)

- ✧ **Key words**：File out
- ✧ **Mission**：

  Write the data you obtained in Problem 1-1~3, write it to a csv like this：

  Origin Data： 2 3 2 5 3 9

  Output Data： 2,3,2,5,3,9,avg,SD

● Problem 2 (100%) ***OOP(2)***

➢ Problem 2-1(50%)

- ✧ Design a class which can help teachers to administrate students' affairs.
- ✧ Each student has his name, ID, Phone, classEvery attribute should be set to private to protect it. And you have to seperate it into .h and .cpp
- ✧ Your main.cpp should be able to run the codes below by including some necessary headers.

```
int main(){
        student A(" Jenny ",12, " 0919350533 ",279);
        cout << "Name: " <<A.getname()<<endl;
        cout << "ID: " <<A.getID()<<endl;
        cout << "Phone: " <<A.getPhone()<<endl;
        cout << "Class: " <<A.getclass()<<endl;
        return 0;
}
```

➢ Problem 2-2(50%)

- ✧ Now, a librarian comes to you and asks for your kind help. He want you to design a program which can be used in recording the situation of borrowing books.
- ✧ Extend and inherit from the aformentioned class. Your class in this subproblem should include: The book borrowed, the expiracy date
- ✧ Your main.cpp should be able to run the code below by including some necessary headers.

```
int main(){
        const int borrowNumber = 2;
        string borrowHistory[borrowNumber][2]={"HarryPotter","2016-08-19","TheHungerGames","2016-08-31"};
        libstudent A("Jenny",12,"0919350533",279,borrowHistory,borrowNumber);
        cout << "Name: " <<A.getname()<<endl;
        cout << "ID: " <<A.getID()<<endl;
        cout << "Phone: " <<A.getPhone()<<endl;
        cout << "Class: " <<A.getclass()<<endl;
        cout << "Borrow History: " <<A.getborrowHistory()<<endl;
        return 0;
}
```

Problem 3 (100%) *Google Code Jam - Store Credit*

- You receive a credit C at a local store and would like to buy two items. You first walk through the store and create a list L of all available items. From this list you would like to buy two items that add up to the entire value of the credit. The solution you provide will consist of the two integers indicating the positions of the items in your list (smaller number first).
- Please download the test data (Small and Big)
  - http://ppt.cc/pDc0X
  - http://ppt.cc/tjfpQ
- The first line of input gives the number of cases, N. N test cases follow. For each test case there will be:
  - One line containing the value C, the amount of credit you have at the store.
  - One line containing the value I, the number of items in the store.
  - One line containing a space separated list of I integers. Each integer P indicates the price of an item in the store.
  - Each test case will have exactly one solution.

```
Input
3
100
3
5 75 25
200
7
150 24 79 50 88 345 3
8
8
2 1 9 4 4 56 90 3
```

- For each test case, output one line containing "Case #**x**: " followed by the indices of the two items whose price adds up to the store credit. The lower index should be output first.
  - Limits
    - $5 \leq \mathbf{C} \leq 1000$
      $1 \leq \mathbf{P} \leq 1000$
  - Small dataset
    - **N** = 10
      $3 \leq \mathbf{I} \leq 100$
  - Large dataset
    - **N** = 50
      $3 \leq \mathbf{I} \leq 2000$

```
Output
Case #1: 2 3
Case #2: 1 4
Case #3: 4 5
```

- Grading Standard
  - Read the data successfully. (25%)
    - Noted the length of each set of data is different.
  - Able to handle one set of data. (25%)
  - Able to handle small set of data and write your result to a txt. (25%)
  - Able to give the answer of big set of data within 10 seconds. (25%)

Problem 4 (100%) ***ACM Contest - Seperating Pebbles***

- Dr. Y has travelled to an ancent ruin in which a large number of circular-shaped ('o') and cross-shaped ('+') pebbles are scatterd on an open field. Being a puzzle enthusiast, Dr. Y tried to figure out if she can draw one straight line so that circular-shaped pebbles and cross-shaped pebbles are on the opposite side of the line and no pebbles on the line. For example, in Fig. 1 (left), Dr. Y can draw a stright line to separate the two types of pebbles, whereas in Fig. 1 (right), it would not be possible. Please write a program to determine if a given group of pebbles can be separated by a single straght line.

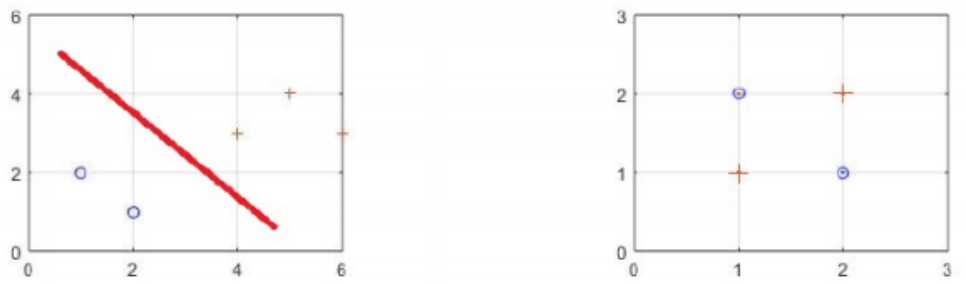
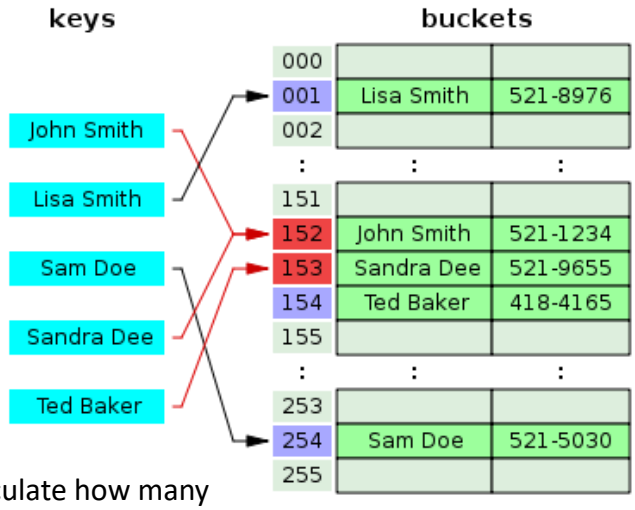


Figure 1: (left), pebbles can be separated by a straight line. (right), pebbles can not be separated by a straight line.

- Input:
  - The number of pebbles of kinds A
  - The positions (x,y) of these kinds A pebbles
  - The number of pebbles of kinds B
  - The positions (x,y) of these kinds B pebbles
- Output:
  - Whether the two kinds of pebbles can be seperated by a line or not.

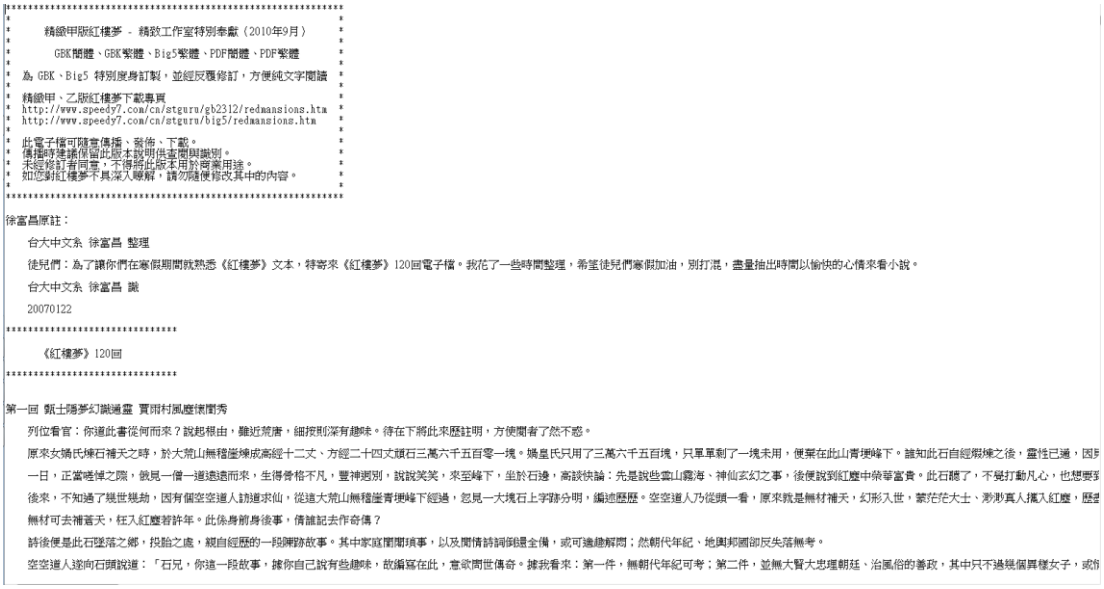
■ Problem 5 (100%) ***Data Structure - Hash Table***

- In computing, a hash table (hash map) is a data structure used to implement an associative array, a structure that can map keys to values. A hash table uses a hash function to compute an index into an array of buckets or slots, from which the desired value can be found.



- **Objective** : If we would like to calculate how many words of each kind of word in Red_Mansions (紅樓夢), we have to build and use hash table; otherwise, the time consuming will be not tolerable.

➢ Problem 5-1 (30%) Traditional Way



```
*****************************************************
*                                                   *
*      精緻甲版紅樓夢 - 精致工作室特別奉獻（2010年9月）*
*        GBK簡體、GBK繁體、Big5繁體、PDF簡體、PDF繁體*
*                                                   *
*  為 GBK、Big5 特別度身訂製，並經反覆修訂，方便純文字閱讀*
*  精緻甲、乙版紅樓夢下載專頁                        *
*  http://www.speedy7.com/cn/stguru/gb2312/redmansions.htm*
*  http://www.speedy7.com/cn/stguru/big5/redmansions.htm*
*                                                   *
*  此電子檔可隨意傳播、發佈、下載。                  *
*  傳播時建議保留此版本說明供查閱與識別。            *
*  未經修訂者同意，不得將此版本用於商業用途。        *
*  如您對紅樓夢不具深入瞭解，請勿隨便修改其中的內容。*
*                                                   *
*****************************************************

徐富昌原註：
    台大中文系 徐富昌 整理
    徒兒們：為了讓你們在寒假期間就熟悉《紅樓夢》文本，特寄來《紅樓夢》120回電子檔。我花了一些時間整理，希望徒兒們寒假加油，別打混，盡量抽出時間以愉快的心情來看小說。
    台大中文系 徐富昌 識
    20070122
*****************************
    《紅樓夢》120回
*****************************

第一回 甄士隱夢幻識通靈 賈雨村風塵懷閨秀
    列位看官：你道此書從何而來？說起根由，雖近荒唐，細按則深有趣味。待在下將此來歷註明，方使閱者了然不惑。
    原來女媧氏煉石補天之時，於大荒山無稽崖煉成高經十二丈、方經二十四丈頑石三萬六千五百零一塊。娘皇氏只用了三萬六千五百塊，只單單剩了一塊未用，便棄在此山青埂峰下。誰知此石自經煅煉之後，靈性已通，因
    一日，正當嗟悼之際，忽見一僧一道遠遠而來，生得骨格不凡，豐神迥別，說說笑笑，來至峰下，坐於石邊，高談快論：先是說些雲山霧海、神仙玄幻之事，後便說到紅塵中榮華富貴。此石聽了，不覺打動凡心，也想要
    後來，不知過了幾世幾劫，因有個空空道人訪道求仙，從這大荒山無稽崖青埂峰下經過，忽見一大塊石上字跡分明，編述歷歷。空空道人乃從頭一看，原來就是無材補天，幻形入世，蒙茫茫大士、渺渺真人攜入紅塵，歷
    無材可去補蒼天，枉入紅塵若許年。此係身前身後事，情誰記去作奇傳？
    詩後便是此石墜落之鄉，投胎之處，親自經歷的一段榮枯故事。其中家庭閨閣瑣事，以及聞情詩詞俚曲全備，或可遣趣解悶；然朝代年紀、地輿邦國卻反失落無考。
    空空道人遂向石頭說道：「石兄，你這一段故事，據你自己說有些趣味，故編寫在此，意欲閨世傳奇。據我看來：第一件，無朝代年紀可考；第二件，並無大賢大忠理朝廷、治風俗的善政，其中只不過幾個異樣女子，或
```

- Download the txt：http://ppt.cc/mmXqq
- **Key words**：File In and Array
- **Mission**：Download and read the data , calculate the number of each different word with array.
- **Hint**：You can make a 2D array, the first column store the word, the second store the times. If the time consumes too much, try a smaller data made by yourself.

➢ Problem 5-2 (20%) Hash Function

- Mission：Build a Hash Function, which means the parameter of this function is a word and its return value is an integer. Your function should look like this：

```
int hashFunction (string str) {
        Your codes here……
}
```

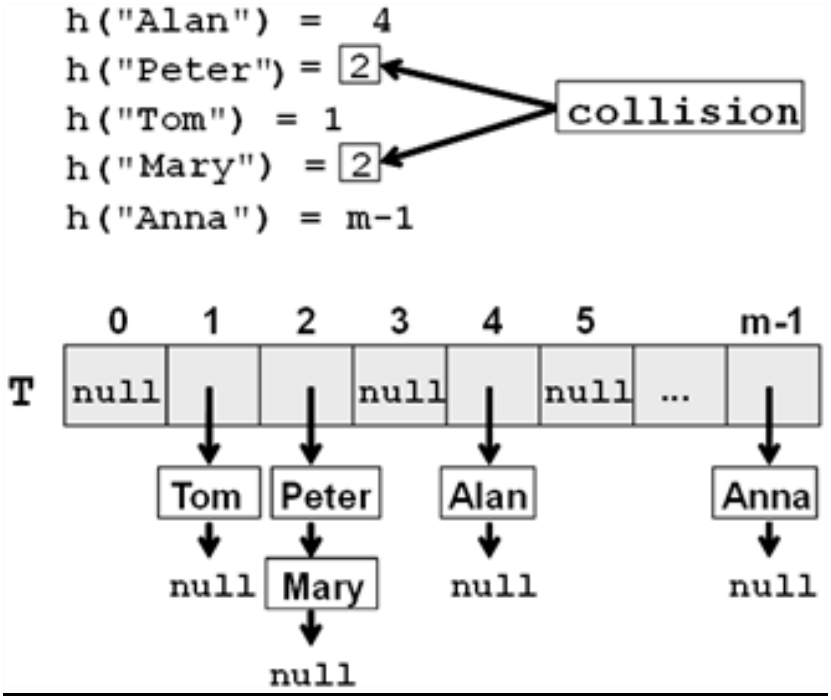
We use string here, for the char can only deposit English alphabet. Every word encoded in UTF-8 has its own value, so we can use it to manufacture a simple hash table. I have written an function for you to transfer the Chinese word to int by unicode. You can use it by copy & paste. By calling this function, your hash function can be achieve in just one line. The method to transfer a word to an integer is shown below.

```
// This function will transfer your word to an integer.
//The value ranges from 0~256^4
unicode_to_int(std::string &utf8)
{
    uint32_t unicode = 0;
    uint8_t first_byte = (uint8_t)utf8[0];
    uint8_t len =
        (first_byte & 0xf0) == 0xf0 ? 4 :
        (first_byte & 0xe0) == 0xe0 ? 3 :
        (first_byte & 0xc0) == 0xc0 ? 2 : 1;
    int k = 1;
    unicode = 0;
    for(auto i = 0; i <= len; ++i) {
        if (((int)utf8[i])>0)
            unicode += ((int)utf8[i])*k;
        else
            unicode += ((int)utf8[i])*-1*k;
        k*=256;
    }
    return (int) unicode;
}
```

- Hint：
  - To reduce every return integer uniformly, you may use %
  - Ref about UTF-8 https://zh.wikipedia.org/wiki/UTF-8

➢ Problem 5-3 (50%) Building a Hash Table

- **Mission**：Modified the code on problem 7-1 to a Hash Table
- **Hint**：
  - You have to judge how big the array you need and the range of the integer returned by hash function.
  - Collision avoidance or handler, which means two different words point to same hash value. You can take a look at the next slide.



- **Hash Collisions**:

  Sometimes, the hash function will transfer two different words into the same hash value. (Like Peter and Mary on the top)

- **Solution**:

  We will make it a 2D array, once the collision happened, deposit the later one in the array.

■ Problem 6 (100%) ***Algorithm - Quick Sort***

- Quicksort is an efficient sorting algorithm, serving as a systematic method for placing the elements of an array in order. When implemented well, it can be about two or three times faster than its main competitors, merge sort and heapsort.

  Quicksort is a comparison sort, meaning that it can sort items of any type for which a "less-than" relation (formally, a total order) is defined. In efficient implementations it is not a stable sort, meaning that the relative order of equal sort items is not preserved. Quicksort can operate in-place on an array, requiring small additional amounts of memory to perform the sorting.

  Mathematical analysis of quicksort shows that, on average, the algorithm takes O(n log n) comparisons to sort n items. In the worst case, it makes O(n2) comparisons, though this behavior is rare.

- There are many kinds of sorting, the publicly known as the fastest one is quick sort, you can take a look on animation first. Sorting is always used as an interview question once you apply for an algorithm / software developer.

- The steps are:
  - Pick an element, called a pivot, from the array.
  - Partitioning: reorder the array so that all elements with values less than the pivot come before the pivot, while all elements with values greater than the pivot come after it (equal values can go either way). After this partitioning, the pivot is in its final position. This is called the partition operation.
  - Recursively apply the above steps to the sub-array of elements with smaller values and separately to the sub-array of elements with greater values.

➢ Problem 6-1 (20%) Quick Sort – Data Reading

- Download the test data :
- http://ppt.cc/BmA2K
- Mission :
- Read the Data in csv to a 2D array that will be used later.
- These data are in the same length to make it friendly for you to achieve this mission.

- Hint :
- Modified the code from Problem 1

➢ Problem 6-2 (20%) Quick Sort - Coding Functions

- In this part, you need to code two functions which will be used later. The first one is swap, it will help you arrange the positions of two elements in an array
- **Mission 1 (10%)**

  Write a function that can help you swap two input values. Your function should like below :

```
void swapFunction (int &a, int &b)
{Your codes here}
```

- **Mission 2 (10%)**

  The second one is that we have to decide a pivot, which means a comparing standard in this algorithm. We can use the left, right, or a random as the pivot. In this exercise, we use random pivot.

  Write a function that can return the random pivot index and value to you. Your function should like below :

```
void pivot (int *arr, int length, int &pivotIndex, int &pivotValue ){
    Your codes here……
}
```

  - Hint :

    #include <stdlib.h> #include <time.h>
  - srand (time(NULL)) ; rand();

    You may also use % to regulate the range of output

➢ Problem 6-3 (25%) Quick Sort – Partitioning

- **Mission** : Write a function that will partition the array into two part by a random pivot. Below is one example.

  If we choose the pivot 26, it will partition the origin array into left (smaller than 26) and right (bigger than 26)

  Origin: 26 13 73 31 38 After : 13 26 73 31 38
- Hint :

To simplify your algorithm, each integer is qnique.Your function should like it below. It will use the two function you wrote before.

```
Void partitioning (int *arr, int length){
      Your codes here……
}
```

➢ Problem 6-4 (25%) Quick Sort – Recursion

- **Mission** :

  Modified the previous codes (partitioning) you wrote. Use the recursion to achieve the sorting.

- Hint :

  Your can use *(p+i) to regulate the start of array as a parameter in partitioning.

  The partitioning should stop once the amounts of elements which are smaller or larger than the pivot is only one. Your function should like this below.

```
void partitioning (int *arr, int length){
            if (length !=1)
                  Your codes here…
}
```

➢ Problem 6-5 (10%) Quick Sort – Sort all Test Data

- **Mission** :
  - Sort all the data you deposit in an array on Problem 6-1 by the function you wrote on Problem 6-4.
  - Write your result to a csv file.

Appendix for Problem 1

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <stdlib.h>
using namespace std;
int main() {
string str;
	ifstream ifile;
	//This array is used to deposit the data importing from csv
	int data[100][11];
	//You can change the filename to open the specific data
	string filename="";
	ifile.open(filename);
	if (ifile) {
		int c = 0;
		while (getline (ifile,str)){
			stringstream ss(str);
			string temp;
			// ss → 每一行資料轉成串流
			for(int i=0;i<11;i++){
				getline(ss,temp,',');
				//atoi is used to transform string to int
				data[c][i]=atoi(temp.c_str());
			}
			c++;
		}
		cout << "Finish reading data" << endl;
		//Now, you can start to use the data array to complete this problem
	}
	else {
		cout << "Error" <<endl;
	}
	ifile.close();
	return 0;
}
```