計算機程式語言

作業四

姓名	潘廣霖	
學號	B03203004	
原始程式檔名	HW04_002.cpp (_ = [A-D])	
評分項目		
分數比重	項目	得分
40%	程式是否能正確執行?	
40%	程式之使用者介面與輸出結果?	
	是否有繳交原始程式檔與執行檔?	
	程式中的註解是否恰當?	
	程式之結構與邏輯是否正確?	
	程式碼的格式是否合乎要求?	
20%	程式之綜合評分	
總分		
評語:		

A.

```
//================================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-10-29
//  FILENAME    : HW04A002.CPP
//  DESCRIPTION : Find all perfect number within 0..10000
//================================================================

#include "stdafx.h"
#include<iostream>
#include<cstdlib>

using namespace std;

// return true if `Num` is a perfect number
bool PerfectNumber(long int Num) {
    int sum = 0;
    // compute all factors within sqrt(n)
    for (int i = 1; i*i <= Num; i++) {
        if (Num%i != 0) continue;
        sum += i;
        // if it is not a square number then add another factor
        if (i*i < Num) sum += Num/i;
    }
    // exclude the number itself
    sum -= Num;
    return (sum == Num);
}

int main() {
    cout << "Perfect numbers within 1 to 10000: " << endl;
    for (int i = 1; i < 10000; i++) {
        if (PerfectNumber(i))
          cout << i << endl;
    }
    system("pause");
}
```

B.

```
//==================================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-10-29
//  FILENAME    : HW04B002.CPP
//  DESCRIPTION : Calculate avg and stdev of numbers, rejecting outlier
using z test.
//==================================================================

#include "stdafx.h"
#include<iostream>
#include<cstdlib>
#include<cmath>

using namespace std;

double __avg(double arr[], int cnt) {
    double sum = 0;
    for(int i = 0; i < cnt; i++)
        sum += 1.0 * arr[i];
    return sum / cnt;
}

double __stdev(double arr[], int cnt) {
    int sum2 = 0;
    for(int i = 0; i < cnt; i++) {
        sum2 += arr[i] * arr[i];
    }
    int avg = __avg(arr, cnt);
    return sqrt(sum2 / cnt - avg * avg);
}

void calculate(double arr[], int cnt, double& avg, double& stdev) {
    avg = __avg(arr, cnt);
    stdev = __stdev(arr, cnt);
}

int main() {
    int n;
    cout << "Number of records [1~10]: n = ";
    cin >> n;

    if (n < 1 || n > 10) {
        cout << "Error! n should be within 1 to 10. Exiting...";
        system("pause");
        return 0;
    }

    double arr[10];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    double avg;
    double stdev;

    calculate(arr, n, avg, stdev);

    double sub = avg - stdev * 2;
    double sup = avg + stdev * 2;

    cout << "Ori Average  = " << avg << endl;
    cout << "Ori St. Dev. = " << stdev << endl;
    cout << "95% CI   = [" << sub << ", " << sup << "]" << endl;

    double newarr[10];
    int k = 0;

    for (int i = 0; i < n; i++) {
        if (arr[i] < sub || arr[i] > sup) {
            cout << "Rejecting " << arr[i] << " because it is away from
(+/-)2 stdev.";
        } else {
            newarr[k++] = arr[i];
        }
    }

    calculate(newarr, k, avg, stdev);

    cout << "New Average  = " << avg << endl;
    cout << "New St. Dev. = " << stdev << endl;

    system("pause");
}
```

C.

```
//================================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-10-30
//  FILENAME    : HW04C002.CPP
//  DESCRIPTION : Sort a list of pairs of numbers on either field.
//================================================================


#include "stdafx.h"
#include<iostream>
#include<iomanip>
#include<algorithm>
#include<cstdlib>

using namespace std;

void swap2(int a[2], int b[2]) {
    int i = a[0], j    = a[1];
     a[0] = b[0]; a[1] = b[1];
     b[0] = i;    b[1] = j;
}

// sorton is the i-th field we want to sort on
// sortorder is 0 if desc otherwise asc.
void sortDesc(int data[][2], int len, int sorton = 1, int sortorder = 0) {
    // bubble sort; inefficient but it's enough here
    int k = 0;
    bool ok = false;
    for (int i = 0; i < len; i++) {
        if (ok) break;
        ok = true;
        for (int j = i; j < len; j++) {
            if (
              (!sortorder && data[i][sorton] < data[j][sorton]) ||
              (sortorder  && data[i][sorton] > data[j][sorton])
            ) {
                ok = false;
                swap2(data[i], data[j]);
            }
        }
    }
}

int main() {
    // maximum item is 100
    int a[100][2];
    int n = 0;
    cout << "Enter some data in stock: (100 items at most)" << endl;
    cout << "Syntax: [<PartNo.> <Quantity>], enter \"0 0\" to end the
input." << endl;

    while(n < 100) {
        cout << "#" << setw(3) << right << (n+1) << ": ";
        int i, j;
        cin >> i >> j;
        if (i == 0 && j == 0)
            break;
        else { a[n][0] = i; a[n][1] = j; }
        n++;
    }

    sortDesc(a, n);

    cout << endl;
    cout << "Sorted by quantity desc.: " << endl;
    for (int i = 0; i < n; i++) {
        cout << setw(8) << left  << a[i][0]
             << setw(8) << right << a[i][1] << endl;
    }

    sortDesc(a, n, 0, 1);
    cout << "Sorted by product no. asc.: " << endl;
    for (int i = 0; i < n; i++) {
        cout << setw(8) << left  << a[i][0]
             << setw(8) << right << a[i][1] << endl;
    }

}
```

D.

```
//================================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-11-08
//  FILENAME    : HW04D002.CPP
//  DESCRIPTION : A 2-dice simulator.
//================================================================

#include "stdafx.h"
#include<iostream>
#include<algorithm>
#include<iomanip>
#include<cstdlib>
#include<ctime>

using namespace std;

void printPercentage(ostream& os, double num) {
    os << (num * 100) << "%";
}

int dice() { return rand() % 6 + 1; }
int toss() { return dice() + dice(); }

void showToss(int cnt) {
    int result[13] = {0};
    double ther_prob_base = 1.0 / 36;

    for (int i = 0; i < cnt; i++) {
        result[toss()]++;
    }

    cout << "Tossing for " << cnt << " times..." << endl;

    // set precision for all the floats we are going to output
    cout << setprecision(3) << fixed;

    for (int i = 2; i <= 12; i++) {
        double exp_prob = ((double)result[i] / cnt);
        double ther_prob = ther_prob_base * min(i-1, 13-i);

        cout << setw(2) << right << i
             << ": "
             << setw(6) << right << result[i];

        cout << "  "
             << setw(6);
        // experimental probability
        printPercentage(cout, exp_prob);
        cout << " ("
             << setw(6);
        // theorical probility
        printPercentage(cout, ther_prob);

        cout << ", "
             << setw(6) << showpos;
        // difference
        printPercentage(cout, ther_prob - exp_prob);

        cout << noshowpos
             << ")"
             << endl;
    }

    cout << endl;
}

int main(int argc, _TCHAR* argv[]) {

    // randomize; using time as random seed
    srand(time(NULL));

    showToss(1000);
    showToss(10000);
    showToss(100000);

    cin.get();
}
```