# Group 3 Assignment

| 组员编号 | 姓名 | 学号 |
| --- | --- | --- |
| 1 | 张晟赫 | D24091111148 |
| 2 | 罗琢 | D24091110572 |
| 3 | 王国霖 | D24091111111 |
| 4 | 孙梦娜 | D24091110375 |

作业要求：1）以小组形式完成作业，写上小组人员姓名及学号；

2）在下面文档对应题后补全代码，并合理注释；

3）运行每一题的代码，给出示例，并录屏上传至畅课。

1. 输入3个double类型的值，判断这3个值是否可以表示一个三角形的三条边。（10分）

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

// 定义了一个返回值为布尔类型的函数
bool IsTriangle(double a,double b,double c) {

    // 判断是否为三角形的条件,三边满足下列条件
    if((a+b>c) && (a+c>b) && (b+c>a)) {
        return true; // 是三角形返回true
    }
    return false; // 不是三角形返回false
}

int main() {
    double a,b,c; //初始化a b c三条边
    cout<<"请分别输入三条边的值(用空格隔开)";
    cin>>a>>b>>c; //输入三条边的值

    // 调用IsTriangle函数
    if(IsTriangle(a,b,c)) {
        printf("这3个值可以表示一个三角形的三条边。");
    }
    else
        printf("这3个值不可以表示一个三角形的三条边。");

    return 0;
}
```

2. 打印一个九九乘法表（提示：可使用setw()进行输出排列）。（10分）

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

int main() {

    //用两个for循环遍历1-9
    for(int i=1;i<=9;i++) { //i的值从1到9 大于9退出for循环
        for(int j=1;j<=i;j++) { //j的值从1到i 大于i退出for循环
```

```
            cout<<setw(2)<<j<<" * "<<i<<" = "<<setw(2)<<i*j<<" ";
            // 引入setw()进行输出排列

        }
        cout<<endl; //i的for循环末尾进行换行
    }

    return 0;
}
```

3. 判断某个数是否素数，以及打印1—1000之内的素数。（20）

```
#include <iostream>
using namespace std;
bool isPrime(int number) {

    if(number == 1 || number == 0) // 用or判断
        return  false; //0和1不是质数 返回false
    else if(number>=2){
        for(int i = 2; i < number; i++){
        // 质数是一个大于1的自然数,从2开始,小于number结束
            if (number % i == 0) {
                return false; // 返回false
            }
        }
    }
    return true;
    // 都满足不能被其他自然数整除的数为质数 返回ture

}

int main() {
    for(int i=1;i<=1000;i++) {
    //for循环遍历1-1000

        if(isPrime(i))
        //调用isPrime函数 判断是否为素数

            cout<<i<<"是素数"<<endl;
            //true 打印为素数

    }
    return 0;
}
```

4. 随机输入一行字符串（例："JxfWD s57dT && 234"），找出其中的大写字母、小写字母、空格、数字以及其他字符各有多少个，统计数量并打印。最后，将其中的大写英文字母改为小写，再输出该字符串。（25分）

```
#include <iostream>
#include <cstring>
using namespace std;
//函数:作用是大写字母转换为小写字母
void ConvertUpper2Lower(char strArr[]) {
    int len = strlen(strArr); //返回长度

    for (int i = 0; i < len; i++) {
        // 判断是否为大写字幕
```

```
        if (strArr[i] >= 'A' && strArr[i] <= 'Z') {
            strArr[i] = strArr[i] + 32;
            // 如果是大写字母转换为小写字母
        }
    }

}
//函数:作用输出各种的数量
void PrintNum(int Arr[]) {
    int len = 5;
    for(int i = 0 ; i< len ;i++) {
        if(i ==0)
            printf("大写字母数量：%d\n", Arr[i]);
        else if(i==1)
            printf("小写字母数量：%d\n", Arr[i]);
        else if(i==2)
            printf("空格数量：%d\n", Arr[i]);
        else if(i==3)
            printf("数字数量：%d\n", Arr[i]);
        else if(i==4)
            printf("其他字符数量：%d\n", Arr[i]);
    }

}
int main() {
    const int N = 60;
    char strArr[N];
    //例子 JxfWD s57dT && 234
    cout << "请随机输入一行字符串：";
    // 读取输入的字符串
    cin.getline(strArr, 50);

    // 初始化数组 定义了一个整形数组 分别存储大写字母 小写字母 空格 数字 以及其他字符的数量
    int NumArr[] = {0,0,0,0,0};

    int len = strlen(strArr); //获取数组长度
    for (int i = 0; i < len; i++) {
        if (strArr[i] >= 'A' && strArr[i] <= 'Z') {
            NumArr[0] = NumArr[0] + 1;   // 如果是大写字母，数量+1
        } else if (strArr[i] >= 'a' && strArr[i] <= 'z') {
            NumArr[1] = NumArr[1] + 1;   // 同理
        } else if (strArr[i] == ' ') {
            NumArr[2] = NumArr[2] + 1;   // 同理
        } else if (strArr[i] >= '0' && strArr[i] <= '9') {
            NumArr[3] = NumArr[3] + 1;   // 同理
        } else {
            NumArr[4] = NumArr[4] + 1;   // 同理
        }
    }
    PrintNum(NumArr); // 调用PrintNum函数
    ConvertUpper2Lower(strArr); //调用ConvertUpper2Lower函数
    cout << "转换后的字符串：" << strArr << endl;

    return 0;
}
```

5. 随机输入10个整形数，分别采用冒泡排序和快速排序两种方式对齐进行升序排列并打印出来。（35分）

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

//函数:冒泡排序函数
void bubbleSort(vector<int>& arr) {
    int n = arr.size();
    // 控制排序轮数
    for (int i = 0; i < n - 1; ++i) {
        // 比较并交换
        for (int j = 0; j < n - i - 1; ++j) {
            if (arr[j] > arr[j + 1]) {
                swap(arr[j], arr[j + 1]);
                //如果当前元素大于下一个元素,进行交换
            }
        }
    }
}

//函数:快速排序函数
void quickSort(vector<int>& arr, int low, int high) {
    if (low < high) {
        int p = arr[low];//选定p
        int left = low, right = high;
        //分区过程，将小于p的放在左边，大于p的放在右边
        while (left < right) {
            while (left < right && arr[right] >= p) --right;
            arr[left] = arr[right];
            while (left < right && arr[left] <= p) ++left;
            arr[right] = arr[left];
        }
        arr[left] = p;
        // 左边进行递归排序
        quickSort(arr, low, left - 1);
        // 右边进行递归排序
        quickSort(arr, left + 1, high);
    }
}
```

```
int main() {

    // 初始化数组
    vector<int> numbers(10);
    for(int i=0;i<10;i++) {
        cin>>numbers[i];
    }
    // 初始化数组
    vector<int> bubbleSorted = numbers;
    vector<int> quickSorted = numbers;

    cout << "未排序的数组: ";
    // for each循环 遍历输出源数组
    for (int num : numbers) {
        cout << num << " ";
    }
    cout << endl;
```

```
    // 调用冒泡排序函数
    bubbleSort(bubbleSorted);
    cout << "冒泡排序结果: ";
    // for each循环 遍历输出数组
    for (int num : bubbleSorted) {
        cout << num << " ";
    }
    cout << endl;

    // 调用快速排序函数
    quickSort(quickSorted, 0, quickSorted.size() - 1);
    cout << "快速排序结果: ";
    // for each循环 遍历输出数组
    for (int num : quickSorted) {
        cout << num << " ";
    }
    cout << endl;

    return 0;
}
```