《面向对象程序设计语言》作业（4.14）

地信班 109092023XXX 许愿

5. 有 n 个人围成一圈，顺序排号。从第 1 个人开始报数（从 1-3 报数），凡报到 3 的人退出圈子，问最后剩下的人原来排在第几号。

```
#include <iostream>
using namespace std;
int main(){
    const int length = 10; // 总人数
    int n = 10; // 当前人数
    // 初始化队列
    int loop[length] = {0};
    for(int i = 0; i < n; i++){
        loop[i] = i + 1;
    }
    // count 为计数器，i 为当前报数的人
    int count = 0;
    int i = 0;
    while (n > 1){ // 只要人数还大于 1 就一直循环
        if (loop[i] != 0){ // 这个人还存在
            count++;
            if (count == 3){
                loop[i] = 0; // 消除这个人
                count = 0; // 重新从 0 开始计数
                n--; // 剩余人数减 1
            }
        }
        i++; // 当前报数的人增加
        if (i == length){ // 回环
            i = 0;
        }
    }
    // 搜索位置，不为 0 的就是剩下的人
    for (int j = 0; j < length; j++){
        if (loop[j] != 0){
            cout << "最后剩下的人原来排在第" << loop[j] << "号" <<
endl;
        }
    }
    return 0;
}
```

```
=Microsoft-MIEngine-Error-igxlp
eter=mi'
最后剩下的人原来排在第4号
```

7.有一字符串，包含 n 个字符。写一函数，将此字符串中从第 m 个字符开始的全部字符复制成为另一个字符串。

```cpp
#include <iostream>
using namespace std;
void copy_str(char *str, char *copy, int m){
    int i = 0;
    while (str[m] != '\0'){ // 不为结束
        copy[i] = str[m];
        i++;
        m++;
    }
    copy[i] = '\0'; // 结束
}
int main(){
    const int string_length = 100000; // 长度
    char str[string_length], copy[string_length];
    cout << "请输入字符串：";
    cin.getline(str, 100); // 输入字符串

    int m;
    cout << "请输入开始复制的字符位置：";
    cin >> m;

    copy_str(str, copy, m);
    cout << "复制后的字符串为：" << copy << endl;
    return 0;
}
```

```
eter=mi'
请输入字符串：dju32gd9320fh239fh23h8d9328ujd934igf34jfdo1i2hediu12bd
请输入开始复制的字符位置：32
复制后的字符串为：4igf34jfdo1i2hediu12bd
PS C:\Users\Xuan\Desktop\24-25-2\CPP\25.4.14work>
```

9. 写一函数，将一个 3*3 的整型矩阵转置。

```cpp
#include <iostream>
using namespace std;
int map[3][3] = {
    {1, 2, 3},
    {4, 5, 6},
```

```
    {7, 8, 9}
};

void zhuanzhi(int map[3][3]){
    int temp[3][3] = {0};
    for (int i = 0; i < 3; i++){
        for (int j = 0; j < 3; j++){
            temp[j][i] = map[i][j];
        }
    }
    for (int i = 0; i < 3; i++){
        for (int j = 0; j < 3; j++){
            map[i][j] = temp[i][j];
        }
    }
}

int main(){
    cout << "原矩阵: " << endl;
    for (int i = 0; i < 3; i++){
        for (int j = 0; j < 3; j++){
            cout << map[i][j] << " ";
        }
        cout << endl;
    }
    zhuanzhi(map);
    cout << "转置后的矩阵: " << endl;
    for (int i = 0; i < 3; i++){
        for (int j = 0; j < 3; j++){
            cout << map[i][j] << " ";
        }
        cout << endl;
    }
    return 0;
}
```

```
eter=mi'
原矩阵:
1 2 3
4 5 6
7 8 9
转置后的矩阵:
1 4 7
2 5 8
3 6 9
```

10. 将一个 5*5 的矩阵中最大的元素放在中心，四个角分别放 4 个最

小的元素（按从左到右、从上到下顺序依次从小到大存放），写一函数实现。用 main 函数调用。

```
#include <iostream>
using namespace std;
int main(){
    int a[5][5] =
{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,2
4,25}; // 定义矩阵
    cout << "原矩阵为: " << endl;
    for(int i=0; i<5; i++){
        for(int j=0; j<5; j++){
            cout << a[i][j] << " ";
        }
        cout << endl;
    }
    int *p, temp, *max;
    max = &a[0][0];
    p = &a[0][0];
    for(int i=0; i<25; i++,p++){ // 找出数组中的最大值，并使 max 指
针变量指向该值的地址
        if(*max<*p){
            max = p;
        }
    }
    temp = a[2][2]; // 利用临时变量使 a[2][2]和数组中的最大值互换
    a[2][2] = *max;
    *max = temp;

    int *arr[4]; // 建立一个指针数组，用来存放数组中 4 个较小值的地址
    for(int i=0; i<4; i++){
        arr[i] = &a[2][2]; // 先将该指针数组都指向该数组中最大值的
地址，方便后续遍历时的比较
    }
    int (*q)[5]; // 建立一个指向数组的指针，用于搜索数组中的 4 个最小
值
    q = a;
    for(int i=0; i<5; i++){
        for(int j=0; j<5; j++){
            if(*arr[3] > *(*(q+i)+j)){
                arr[3] = *(q+i)+j;
                for(int a=3; a>0; a--){
```

```
                    if(*arr[a] < *arr[a-1]){ // 冒泡排序，将存放 4 个最小值地址的指针数组按照从小到大的顺序存储
                        int *k;
                        k = arr[a];
                        arr[a] = arr[a-1];
                        arr[a-1] = k;
                    }
                }
            }
        }
    }
    // 借用临时变量，将 4 个最小值按照要求放于合适的位置。
    int temp2;
    temp2=a[0][0];a[0][0]=*arr[0];*arr[0]=temp2;
    temp2=a[0][4];a[0][4]=*arr[1];*arr[1]=temp2;
    temp2=a[4][0];a[4][0]=*arr[2];*arr[2]=temp2;
    temp2=a[4][4];a[4][4]=*arr[3];*arr[3]=temp2;

    // 输出新矩阵
    cout << "新矩阵为: " << endl;
    for(int i=0; i<5; i++){
        for(int j=0; j<5; j++){
            cout << a[i][j] << " ";
        }
        cout << endl;
    }
    return 0;
}
```

```
ndowsDebugLauncher.exe'
e-Error-m0a4kazv.rm5' '-
原矩阵为:
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
新矩阵为:
1 5 21 13 2
6 7 8 9 10
11 12 25 14 15
16 17 18 19 20
3 22 23 24 4
```