约瑟夫生死者游戏

问题描述

约瑟夫生者死者游戏的大意是：30个旅客同乘一条船，因为严重超载，加上风高浪大危险万分；因此船长告诉乘客，只有将全船一半的旅客投入海中，其余人才能幸免于难。无奈，大家只得统一这种方法，并议定30个人围成一圈，由第一个人开始，依次报数，数到第9人，便将他投入大海中，然后从他的下一个人数起，数到第9人，再将他投入大海，如此循环，直到剩下15个乘客为止。问哪些位置是将被扔下大海的位置。

项目功能要求

要求使用单循环列表

本游戏的数学建模

假如N个旅客排成一个环形，依次顺序编号1,2, …, N。从某个指定的第S号开始。沿环计数，每数到第M个人就让器出列，且从下一个人开始重新计数，继续进行下去。这个过程一直进行到剩下K个旅客为止。

本游戏要求用户输入的内容

1. 旅客的个数，也就是N的值；
2. 离开旅客的间隔书，也就是M的值；
3. 所有旅客的序号作为一组数据要求存放在某种数据结构中。

本游戏要求输出的内容

1. 离开旅客的序号；
2. 剩余旅客的序号。

算法设计

建立单循环列表，顺序遍历结点。通过删除结点实现旅客死亡并输出旅客序号，最后遍历链表输出存活旅客。

功能实现及代码分析

1、结构体和类设计

将单个旅客封装为结点，内容包括序号和指向下一旅客的指针。

整个循环链表封装成一个类，包含进入链表的头指针、剩余旅客数量和相关参数（N、S、M、K），并包含创建链表和开始游戏的函数。

```
struct Passenger {
    int index;
    Passenger *next;
};

class Josephus {
private:
    Passenger *head;
    int left;   //剩余旅客数量
    int n, s, m, k;

public:
    Josephus(const int &N, const int &S, const int &M, const int &K): n(N), s(S), m(M), k(K), left(N) {
        create();
    }
    ~Josephus();
    void create();    //创建旅客列表
    void startGame(); //开始模拟游戏
};
```

2、检查输入

检查用户输入数据的合法性，均大于0，指定开始号码S应小于总人数N，剩余旅客数K应小于总人数N，若输入有误，会要求用户一直修改至合法。

```
void check(int &n, int &s, int &m, int &k) {
    while (n < 0) {
        cout << "N值有误，请重新输入：";
        cin >> n;
        check(n, s, m, k);
    }
    while (s < 0) {
        cout << "S值有误，请重新输入：";
        cin >> s;
        check(n, s, m, k);
    }
    while (m < 0) {
        cout << "M值有误，请重新输入：";
        cin >> m;
        check(n, s, m, k);
    }
    while (k < 0) {
        cout << "K值有误，请重新输入：";
        cin >> k;
        check(n, s, m, k);
    }
    while (n < s) {
        cout << "N、S值有误，请分别重新输入N和S：";
        cin >> n >> s;
        check(n, s, m, k);
    }
    while (n < k) {
        cout << "N、K值有误，请分别重新输入N和K：";
        cin >> n >> k;
        check(n, s, m, k);
    }
    return;
}
```

演示

```
现有N人围成一圈，从第S个人开始依次报数，报M的人出局，再由下一人开始报数，如此循环，直至剩下K人为止
请输入生死游戏的总人数N：10
请输入游戏开始的位置S：11
请输入死亡数字M：-1
请输入剩余的生者人数K：15
M值有误，请重新输入：4
N、S值有误，请分别重新输入N和S：10 2
N、K值有误，请分别重新输入N和K：10 30
N、K值有误，请分别重新输入N和K：10 6
第1个死者的位置是：      5
第2个死者的位置是：      9
第3个死者的位置是：      3
第4个死者的位置是：      8
最后剩下：6人
剩余的生者位置为：      1     2     4     6     7     10
Process finished with exit code 0
```

3、进入游戏

进入游戏，引导用户输入参数，新建链表并开始模拟过程。

```
int main() {
    cout << "现有N人围成一圈，从第S个人开始依次报数，报M的人出局，再由下一人开始报数，如此循环，直至剩下K人为止" << endl;
    int N = 0, S = 0, M = 0, K = 0;
    cout << "请输入生死游戏的总人数N：";
    cin >> N;
    cout << "请输入游戏开始的位置S：";
    cin >> S;
    cout << "请输入死亡数字M：";
    cin >> M;
    cout << "请输入剩余的生者人数K：";
    cin >> K;
    check(N, S, M, K);
    Josephus newGame(N, S, M, K);
```

```
    newGame.startGame();
    return 0;
}
```

4、创建旅客链表

逐个添加结点，在尾结点处将next指针指向第一个结点，形成循环链表。

```
void Josephus::create() {
    int number = 1;
    Passenger *pre = NULL;
    for (int i = 0; i < n; ++i) {
        Passenger *p = new Passenger;
        p->index = number++;
        if (i == 0) {
            head = p;
        } else if (i == n - 1) {
            pre->next = p;
            p->next = head;
        } else {
            pre->next = p;
        }
        pre = p;
    }
}
```

5、开始模拟游戏

模拟游戏，死者删除结点并输出信息，存活旅客信息通过遍历链表输出。

```
void Josephus::startGame() {
    Passenger *p = head, *pre = NULL;
    while (p->index != s) p = p->next;
    int n = 1, count = 1;
    while (n <= m) {
        if (n < m) {
            pre = p;
            p = p->next;
            n++;
        } else {
            pre->next = p->next;
            cout << "第" << count << "个死者的位置是："
                 << std::right << setw(count > 9 ? 5 : 6)
                 << p->index << endl;
            if (p == head) head = p->next;
            delete p;
            count++;
            if (--left == k) break;
            n = 1;
            p = pre->next;
        }
    }
    cout << "最后剩下：" << k << "人" << endl;
    cout << "剩余的生者位置为：";
    p = head;
    while (p->next != head) {
        cout << "     " << p->index;
        p = p->next;
    }
    cout << "     " << p->index;
}
```

用例演示

无人死亡

```
现有N人围成一圈，从第S个人开始依次报数，报M的人出局，再由下一人开始报数，如此循环，直至剩下K人为止
请输入生死游戏的总人数N：10
请输入游戏开始的位置S：10
请输入死亡数字M：1
```

```
请输入剩余的生者人数K: 10
无人死亡
```

有人死亡

```
现有N人围成一圈，从第S个人开始依次报数，报M的人出局，再由下一人开始报数，如此循环，直至剩下K人为止
请输入生死游戏的总人数N: 30
请输入游戏开始的位置S: 1
请输入死亡数字M: 9
请输入剩余的生者人数K: 15
第1个死者的位置是:  9
第2个死者的位置是:  18
第3个死者的位置是:  27
第4个死者的位置是:  6
第5个死者的位置是:  16
第6个死者的位置是:  26
第7个死者的位置是:  7
第8个死者的位置是:  19
第9个死者的位置是:  30
第10个死者的位置是:  12
第11个死者的位置是:  24
第12个死者的位置是:  8
第13个死者的位置是:  22
第14个死者的位置是:  5
第15个死者的位置是:  23
最后剩下: 15人
剩余的生者位置为:  1  2  3  4  10  11  13  14  15  17  20  21  25  28  29
Process finished with exit code 0
```