队列和广度优先搜索算法BFS

19数媒技杨雪婷

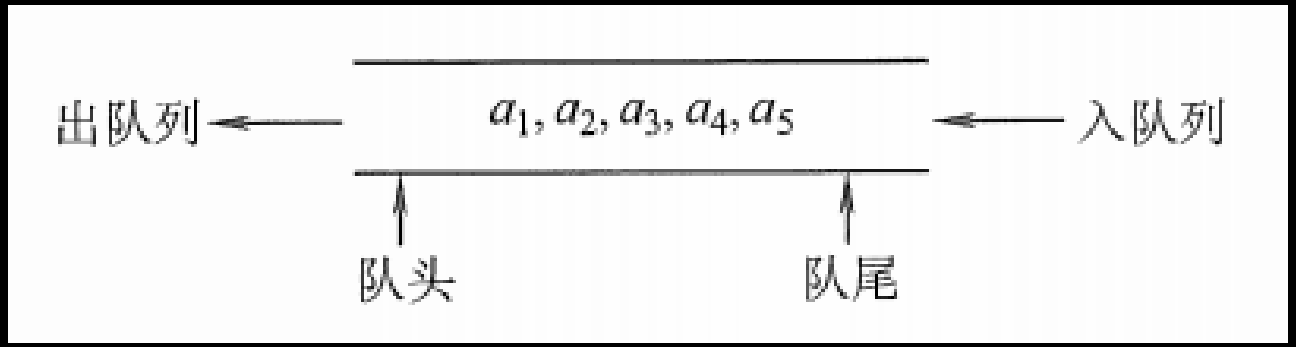
介绍内容：

1. 队列
2. 优先队列
3. bfs
4. 例题

队列 QUEUE

1、 区别栈和队列

栈：先进后出
队列：先进先出



2、重点熟悉STL中的函数使用

使用需添加

```
# include<queue>
```

定义

```
queue<type> name;
```

常用STL函数

函数	作用
q.push()	入队
q.pop()	出队
q.front()	返回首元素
q.back()	返回末元素
q.size()	输出现有元素的个数
q.empty()	队列为空返回1，反之返回0

元素访问

queue的访问比较特殊，每次只能访问队首元素。

```
#include <iostream>
#include <queue>
using namespace std;
int main()
{
    queue< int > q;
    q.push(1);q.push(2);q.push(3);
    while( !q.empty() )
    {
        cout << q.front() << " ";
        q.pop();
    }
}
```

优先队列 PRIORITY QUEUE

使用需添加

```
# include<queue>
```

和queue的区别：

1. 找队首元素：q.front() 改为 q.top()
2. 队顶不再是最早入队的元素，返回值由优先队列定义的规则决定

常用定义：

```
//升序队列
priority_queue <int,vector<int>,greater<int> > q;
//降序队列，大顶堆，默认
priority_queue <int,vector<int>,less<int> >q;
```

自定义结构体的优先队列：

方法1 -- 运算符重载<

```
struct tmp1
{
    int x;
    tmp1(int a) {x = a;}
    bool operator<(const tmp1& a) const
    {
        return x < a.x; //大顶堆
    }
};
//main函数里:
priority_queue<tmp1> d;
```

方法2 -- 重写仿函数

```
struct tmp2
{
    bool operator() (tmp1 a, tmp1 b)
    {
        return a.x < b.x; //大顶堆
    }
};
```

main函数里：

```
priority_queue<tmp1, vector<tmp1>, tmp2> f;
```

广度优先搜索 BFS

题目：找出最短路径

- 给迷宫的长宽：m,n
- 给出迷宫m x n的 0 1 矩阵（0为墙，1可走）
- 从左上角走到右下角
  测试样例：

```
5 5
0 1 0 0 0
0 1 0 1 0
0 0 0 0 0
0 1 1 1 0
0 0 0 1 0
```

关键思路：辐射得往外扩散地找路

参考链接

关键代码：

```
queue<node> q;//sx,sy起点坐标
q.push(node(sx,sy))
while(q.size()){
    node t = q.front()
    q.pop()
    //当前t.x , t.y
    for(int  i = 0 ;i < 4 ;i++)
    {
        int newx = t.x + dx[i]
        int newy = t.y + dy[i]
        if(can(newx,newy)){
            q.push(node(newx,newy))
        }
    }
}
```

例题

题目：给出一个整数n（不大于200），找出这个数的由0,1组成的最小倍数

思路：

- 初始是什么呢
- 状态是什么呢（类比上下左右走来想）
- 什么时候判断‘走到终点’了呢

关键代码

```
string bfs(int n){
    Que.push('1');
    while(Que.size()){
        string now = Que.front();
        Que.pop();
        // 将string转成int，对n取模得到m
        if(m == 0) return now;
        if(vis[m]) continue;
        else{
            vis[m] = true;
            //如果没有出现这种情况，从当前可以改变的状态（添0 、 添1）
            Que.push(string(now+'0')); Que.push(string(now+'1'));
        }
    }
}
```

讲完了，大家快乐做题叭！

感谢 18数媒技 李笙润