代码：

#include <iostream>

using namespace std;

class Dblstack{

private:

int top[2], bot[2];

int\* v;

int m;

public:

Dblstack(int M){

this->m = M;

v = new int[m];

top[0] = -1;

top[1] = m;

bot[0] = 0;

bot[1] = m - 1;

}

~Dblstack() {

delete[] v;

}

int getv(int n) {

return this->v[n];

}

bool CheckMax() {//判断双栈栈满，返回1为栈满，0为栈不满

int i = top[0] - bot[0] + 1;

int j = bot[1] - top[1] + 1;

if (i + j == m)return 1;

else return 0;

}

bool Checktop0(){//判断0栈空，返回1为栈空，0为栈非空

if (top[0] == -1)return 1;

else return 0;

}

bool Checktop1() {//判断1栈空，返回1为栈空，0为栈非空

if (top[1] == m)return 1;

else return 0;

}

bool push(int n, const int m) {//压栈

if ((m!=0&&m!=1)||this->CheckMax() == 1)return 0;

else {

if (m == 0) {

this->v[top[0] + 1] = n;

top[0]++;

}

if (m == 1) {

this->v[top[1] - 1] = n;

top[1]--;

}

}

}

bool pop(int & n,const int m) {//退栈

if ((m != 0 && m != 1)||(this->Checktop0()==1&&this->Checktop1()==1))return 0;

else {

if (m == 0) {

n=v[top[0]];

top[0]--;

return 1;

}

if (m == 1) {

n=v[top[1]];

top[1]++;

return 1;

}

}

}

};

int main()

{

Dblstack dbl(3);

dbl.push(1, 0);

dbl.push(2, 0);

dbl.push(3, 1);

for (int i = 0; i < 3; i++) {

cout << dbl.getv(i)<<" ";

}

cout << endl;

int n=0;

dbl.pop(n,0);

cout << n << " ";

dbl.pop(n, 0);

cout << n << " ";

dbl.pop(n, 1);

cout << n << " ";

}

结果：

