```
/*
* 银行叫号程序
*/
#include<iostream>
#include <queue>
#include <vector>
#include <stdio.h>
#include <tchar.h>
#include <windows.h>
//#include <SDKDDKVer.h>
using namespace std;

//每个用户的服务时间
int serveTime=4;

//时间间隔
int timeInterval=2;

//用户类
class User{
public:

        //用户ID
        int id;

        //等待时间
        int isWait;

        //到达时间
        int arriveTime;
```

```
        //开始服务时间
        int serveTime;

        //用户类型
        char *type;

        //构造函数
        User(){type=NULL;}
        User(int id, int isWait,int arriveTime,int serveTime,char *type){

                this->id=id;
                this->isWait=isWait;
                this->arriveTime=arriveTime;
                this->serveTime=serveTime;
                this->type=type;
        }

        //服务完毕
        void getServed(){cout<<this->type<<this->id<<"已经被服务完毕，离开银行。"<<endl;}
};

//普通用户
class NormalUser: public User{
public:
        NormalUser(){}
        NormalUser(int id, int isWait,int arriveTime,int
serveTime):User(id,isWait,arriveTime,serveTime,"普通用户")
        {}
        void getServed(){cout<<this->type<<this->id<<"已经被服务完毕，离开银行。"<<endl;}
};
```

```
//VIP用户
class VIPUser: public User{
public:
        VIPUser():User(){}
        VIPUser(int id, int isWait,int arriveTime,int
serveTime):User(id,isWait,arriveTime,serveTime,"VIP用户"){}
        void getServed(){
                cout<<this->type<<this->id<<"已经被服务完毕，离开银行。"<<endl;
        }
};

//公用用户
class OfficialUser: public User{
public:
        OfficialUser():User(){}
        OfficialUser(int id, int isWait,int arriveTime,int
serveTime):User(id,isWait,arriveTime,serveTime,"公用用户"){}
        void getServed(){cout<<this->type<<this->id<<"已经被服务完毕，离开银行。"<<endl;}
};

//窗口类
class BankWindow{
public:

        //窗口是否繁忙
        bool isBusy;

        //窗口ID
        int id;

        //用户
```

```
        User client;

        //窗口类型
        char *type;

        //构造函数
        BankWindow(){type=NULL;}
        BankWindow(bool isBusy,int id,User client,char *type){
                this->isBusy=isBusy;
                this->id=id;
                this->client=client;
                this->type=type;
        }

        //处理用户服务
        void HandleUser(){cout<<this->type<<this->id<<"正在为"<<this->client.type<<this->client.id<<"服务"<<endl;}
};

//普通窗口
class NormalBankWindow: public BankWindow{
public:
        NormalBankWindow():BankWindow(){}
        NormalBankWindow(bool isBusy,int id,User client):BankWindow(isBusy,id, client,"普通窗口"){}
        void HandleUser(){
                cout<<this->type<<this->id<<"正在为"<<this->client.type<<this->client.id<<"服务"<<endl;
        }
};
```

```
//VIP窗口
class VIPBankWindow: public BankWindow{
public:
	VIPBankWindow():BankWindow(){}
	VIPBankWindow(bool isBusy,int id,User client):BankWindow(isBusy,id, client,"VIP窗口")
{}
	void HandleUser(){
		cout<<this->type<<this->id<<"正在为"<<this->client.type<<this->client.id<<"服
务"<<endl;
	}
};

//公用窗口
class OfficialBankWindow: public BankWindow{
public:
	OfficialBankWindow():BankWindow(){}
	OfficialBankWindow(bool isBusy,int id,User client):BankWindow(isBusy,id, client,"公用
窗口"){}
	void HandleUser(){
		cout<<this->type<<this->id<<"正在为"<<this->client.type<<this->client.id<<"服
务"<<endl;
	}
};

//模拟类
class Simulater{
public:

	//普通用户队列
	queue<NormalUser> NormalUserQueue;
```

```
//VIP用户队列
queue<VIPUser> VIPUserQueue;

//公用用户队列
queue<OfficialUser> OfficialUserQueue;

//普通窗口链表
vector<NormalBankWindow> nbw;

//VIP窗口链表
vector<VIPBankWindow> vbw;

//公用窗口链表
vector<OfficialBankWindow> obw;

//构造函数
Simulater(){}
bool IsServed(int time,User user){
        if(time-user.serveTime>=serveTime)
                return true;
        return false;
}

//用户入队
void customerEnter(User user){
        if(user.type=="普通用户"){
                this->NormalUserQueue.push((NormalUser&)user);
        }
        else if(user.type=="VIP用户"){
                this->VIPUserQueue.push((VIPUser&)user);
        }
```

```
            else{
                    this->OfficialUserQueue.push((OfficialUser&)user);
            }

    }

    //模拟用户进入银行
    void simulaterCustomerEnter(User user){
            this->customerEnter(user);
            cout<<user.type<<user.id<<"进入银行"<<endl;
    }

    //模拟叫号
    void simulaterCallCustomer(int time){
            //检查普通窗口是否有空闲的
            for(int j=0;j<nbw.size();j++)
                    {
                            //窗口空闲或者该窗口的用户刚好办完业务
                            if(nbw[j].isBusy&&this->IsServed(time,nbw[j].client)||!
(nbw[j].isBusy))
                            {
                                    //窗口的用户刚好办完业务，该用户离开银行
                                    if(nbw[j].isBusy&&this->IsServed(time,nbw[j].client))
                                    {
                                            nbw[j].client.getServed();
                                    }
                                    nbw[j].isBusy=false;
                                    nbw[j].client.id=-1;

                                    //有普通用户正在排队，呼叫该用户到空闲窗口办理业务
                                    if(!this->NormalUserQueue.empty()){
```

```
                                        NormalUser user=this-
>NormalUserQueue.front();
                                        this->NormalUserQueue.pop();
                                        this->callCustomer(user,nbw[j],time);
                                }
                        }
                        //窗口正在服务用户
                        else{
                                nbw[j].HandleUser();
                        }
                }
        //检查VIP窗口是否有空闲的
        for(int j=0;j<vbw.size();j++)
           {
                //窗口空闲或者该窗口的用户刚好办完业务
                        if(vbw[j].isBusy&&this->IsServed(time,vbw[j].client)||!
(vbw[j].isBusy))
                        {
                                //窗口的用户刚好办完业务，该用户离开银行
                                if(vbw[j].isBusy&&this->IsServed(time,vbw[j].client))
                                {
                                        vbw[j].client.getServed();
                                }
                                vbw[j].isBusy=false;
                                vbw[j].client.id=-1;
                                //没有VIP用户正在排队
                                if(this->VIPUserQueue.empty()){
                                        //有普通用户正在排队
                                        if(!this->NormalUserQueue.empty()){
                                                NormalUser user=this-
>NormalUserQueue.front();
```

```
                                        this->NormalUserQueue.pop();
                                        this->callCustomer(user,vbw[j],time);
                                    }
                                }
                                //有VIP用户正在排队
                                else{
                                        VIPUser user=this->VIPUserQueue.front();
                                        this->VIPUserQueue.pop();
                                        this->callCustomer(user,vbw[j],time);
                                }
                            }
                            //窗口正在服务用户
                            else{
                                    vbw[j].HandleUser();
                            }
                    }
                //检查公用窗口是否有空闲的
                for(int j=0;j<obw.size();j++)
                    {
                            //窗口空闲或者该窗口的用户刚好办完业务
                                if(obw[j].isBusy&&this->IsServed(time,obw[j].client)||!
(obw[j].isBusy))
                                {
                                        //窗口的用户刚好办完业务，该用户离开银行
                                        if(obw[j].isBusy&&this->IsServed(time,obw[j].client))
                                        {
                                                obw[j].client.getServed();
                                        }
                                        obw[j].isBusy=false;
                                        obw[j].client.id=-1;
```

```
                                        //没有公用用户正在排队
                                        if(this->OfficialUserQueue.empty()){
                                                //有普通用户正在排队
                                                if(!this->NormalUserQueue.empty()){
                                                        NormalUser user=this-
>NormalUserQueue.front();
                                                        this->NormalUserQueue.pop();
                                                        this->callCustomer(user,obw[j],time);
                                                }

                                        }
                                        //有公用用户正在排队
                                        else{
                                                OfficialUser user=this-
>OfficialUserQueue.front();
                                                this->OfficialUserQueue.pop();
                                                this->callCustomer(user,obw[j],time);
                                        }
                                }
                                //窗口正在服务用户
                                else{
                                        obw[j].HandleUser();
                                }
                        }
                }

        //显示呼叫用户
        void callCustomer(User user,BankWindow &window,int time){
                if(window.isBusy)
                        return;
                else{
```

```
                        //请user到窗口window办理业务
                        window.isBusy=true;
                        user.serveTime=time;
                        user.isWait=time-user.arriveTime;
                        window.client=user;
                        cout<<"请"<<user.id<<"号"<<user.type<<"到"
<<window.type<<window.id<<"办理业务"<<endl;
                }
        }

        //初始化
        void Initialize(){
                //初始化三个普通窗口，1个VIP窗口，1个公用窗口
                NormalUser user(-1,0,0,0);
                NormalBankWindow nbw1(false,1,user);
                NormalBankWindow nbw2(false,2,user);
                NormalBankWindow nbw3(false,3,user);
                VIPBankWindow vbw1(false,4,user);
                OfficialBankWindow obw1(false,5,user);
                nbw.push_back(nbw1);
                nbw.push_back(nbw2);
                nbw.push_back(nbw3);
                vbw.push_back(vbw1);
                obw.push_back(obw1);
        }
        //模拟主函数
        void simulate(){
                int id=1;
        this->Initialize();
                for(int i=0;i<=12;i+=timeInterval)
                {
```

```
cout<<i<<"s: "<<endl;

//在0s，2s，6s时，有新客户进入银行，模拟用户进入银行
if(i==0)
{
        for(int j=0;j<3;j++)
        {
                NormalUser nu(id++,0,i,0);
                this->simulaterCustomerEnter(nu);
        }
        VIPUser vu(id++,0,0,0);
        this->simulaterCustomerEnter(vu);
        OfficialUser ou(id++,0,0,0);
        this->simulaterCustomerEnter(ou);
}
if(i==2)
{
        for(int j=0;j<5;j++)
        {
                NormalUser nu(id++,0,i,0);
                this->simulaterCustomerEnter(nu);
        }
}
if(i==6)
{
        for(int j=0;j<5;j++)
        {
                NormalUser nu(id++,0,i,0);
                this->simulaterCustomerEnter(nu);
        }
        VIPUser vu(id++,0,i,0);
```

```
                        this->simulaterCustomerEnter(vu);
                }

                //模拟叫号
                this->simulaterCallCustomer(i);

                //检查是否有用户正在排队

if(NormalUserQueue.empty()&&VIPUserQueue.empty()&&OfficialUserQueue.empty())
                {
                        cout<<"没有用户正在排队！"<<endl;
                }
                else{
                        cout<<"有"
<<NormalUserQueue.size()+VIPUserQueue.size()+OfficialUserQueue.size()<<"个用户正在排队！"
<<endl;
                }
                //睡眠2s
                Sleep(2000);
        }

    }
};
int main()
{
    cout<<"--------------欢迎使用银行叫号系统-----------------"<<endl;
    Simulater simulater;
    simulater.simulate();
    cout<<"--------------------The end-----------------------"<<endl;
    return 0;
}
```