# 实验代码

if(StaIndex == FINISH\_LOAD)  
    {  
        time1 = GetTheShortestTime(6,1)+line1\_sta[5].SelfLoadTime + x1;  
        time2 = GetTheShortestTime(6,2)+line2\_sta[5].SelfLoadTime + x2;  
    }  
    else  
    {  
        if(lineIndex == 1)  
        {  
            time1 = GetTheShortestTime(StaIndex-1,1) +line1\_sta[StaIndex-1].SelfLoadTime + line1\_sta[StaIndex-1].MoveNextStaTime;  
            time2 = GetTheShortestTime(StaIndex-1,2) +line2\_sta[StaIndex-1].SelfLoadTime + line2\_sta[StaIndex-1].MoveOtherLineNextStaTime;  
        }  
        else  
        {  
            time2 = GetTheShortestTime(StaIndex-1,2) + line2\_sta[StaIndex-1].SelfLoadTime +line2\_sta[StaIndex-1].MoveNextStaTime;  
            time1 = GetTheShortestTime(StaIndex-1,1) + line1\_sta[StaIndex-1].SelfLoadTime +line1\_sta[StaIndex-1].MoveOtherLineNextStaTime;  
        }  
    }  
    return (time1<time2)?time1:time2;  
}  
  
bool LoadConfigFile()  
{  
    FILE \*fp = fopen("StationInfo.txt","rb");  
    if(!fp) return false;  
    int i=0,line = 0,j=0;  
    while(!feof(fp))  
    {  
        if(line == 0)  
        {  
            fscanf(fp,"%d %d",&e1,&e2);  
            line++;  
            continue;  
        }  
        if(line == 1)  
        {  
            fscanf(fp,"%d %d",&x1,&x2);  
            line++;  
            continue;  
        }  
        if(line < 8)  
        {  
            line++;  
            fscanf(fp,"%d %d %d",&line1\_sta[i].SelfLoadTime,&line1\_sta[i].MoveNextStaTime,&line1\_sta[i].MoveOtherLineNextStaTime);  
            i++;  
            continue;  
        }  
        fscanf(fp,"%d %d %d",&line2\_sta[j].SelfLoadTime,&line2\_sta[j].MoveNextStaTime,&line2\_sta[j].MoveOtherLineNextStaTime);  
        j++;  
    }  
    fclose(fp);  
    fp = NULL;  
    return true;  
}  
void ShowConfigInfo(void)  
{  
    std::cout << "Configrature information:" << std::endl;  
    std::cout << "Into line1 time:" <<e1<<"   Out line1 time:"<< x1<<std::endl;  
    std::cout << "Into line2 time:" <<e2<<"   Out line2 time:"<< x2<<std::endl;  
    std::cout << "Stations in line1(left: load time    middle: move to this line next station time     right: move to line2 time):"<<std::endl;  
    for(int i=0;i<6;i++)  
    {  
        std::cout << "station" << i+1 << ":" << line1\_sta[i].SelfLoadTime << "  "<<line1\_sta[i].MoveNextStaTime<<"  "<<line1\_sta[i].MoveOtherLineNextStaTime <<std::endl;  
    }  
    std::cout << "Stations in line2(left: load time    middle: move to this line next station time     right: move to line1 time):" <<std::endl;  
    for(int i=0;i<6;i++)  
    {  
        std::cout << "station" <<i+1<<":"<<line2\_sta[i].SelfLoadTime<< "  "<<line2\_sta[i].MoveNextStaTime <<"  " <<line2\_sta[i].MoveOtherLineNextStaTime<<std::endl;  
    }  
  
}  
int main()  
{  
    if(LoadConfigFile() == false)  
    {  
        std::cout << "Open StationInfo.txt  failed!" << std::endl;  
        return -1;  
    }  
    //ShowConfigInfo();  
    CreatTimeAndPathTable(6);  
    int time1 = line1\_time[5]+x1;  
    int time2 = line2\_time[5]+x2;  
    std::cout <<"the Shortest time:" << ((time1<time2)?time1:time2)<<std::endl;  
    std::cout << "the best path:"<<std::endl;  
    for(int i=1;i<6;i++)  
    {  
        int iPath = line1\_path[i];  
        if(time2 < time1) iPath = line2\_path[i];  
        std::cout << i+1<<":"<<iPath<<std::endl;  
    }  
    return 0;  
}