**<14j>小组<2016火车调度模拟系统>详细设计**

**版本号：14j-Details-03**

**编制时间：2016年5月29日**

**编制人员：罗暄澍 谢天 刘润程**

1. **模块1<Main>**

**1.1函数1**

**函数名称：**int main()

**函数参数说明：**无

**函数功能：** 定义变量

轨道参数

火车参数

完善参数

确认信息

执行策略

**函数局部变量设计:** TrainInfo train[4];

int i;

int run\_method;

FILE \*setting;

int temp;

char tempinput;

**函数算法逻辑：**

引用函数LoadInfo(&train[1],&train[2],&train[3]);

引用函数SetDefaultSet(train);

for(temp=1; temp<4; temp++)引用函数OutputSet(train,temp);

scanf输入tempinput

for(temp=1; temp<4; temp++)引用函数InputSet(train,temp);

引用函数CompleteInfo(&train[1],&train[2],&train[3]);//完善参数

引用函数EnsureInfo； FileWrite；//确认信息

选择火车执行策略AllControl(train);

1. **模块2<** CreateOrbit **>**

**2.1函数1**

**函数名称：**void LoadInfo(TrainInfo\* train1,TrainInfo\* train2,TrainInfo\* train3)

**函数参数说明：**3辆火车的初始信息

**函数功能：**输入轨道参数、作图、输出数据、赋值

**函数局部变量设计： i**nt Y2,X3;

int L1,W1,L2,W2,L3,W3;

int a[100][100]={0};

int i,j,k;

int state;

**函数算法逻辑:**

输入轨道1参数 scanf输入长和宽

While长宽不符合要求时

重新输入长宽

输入轨道2参数 scanf输入;

While输入不合适时 重新输入

Scanf输入长和宽;

While长宽不合适 scanf输入state

While state不等于0和1时 重新输入 if(state==0) 输入长和宽;

If状态1时 输入长

whileY2不合适时 输入长

重新输入长和宽

输入轨道3参数（同轨道2）

For循环j=0到L a[j][0]=1; a[j][W1]=1;

For循环j=L1到j=L1+L2 a[j][Y2]=1; a[j][Y2+W2]=1; for循环j=X3到j=X3+L3 a[j][W1+W3]=1;

j=0;

for循环k=0到k=w1 a[0][k]=1; a[L1][k]=1;

for循环k=Y2到k=Y2+W2 a[L1+L2][k]=1;

for循环k=W1到k=W1+W3 a[X3][k]=1; a[X3+L3][k]=1;

for循环j=0到j=L1+L2

for循环k=0到k=W1+W3

if当a[j][k]=0时 输出空格

else if当a[j][k]=1时 输出图形□

赋值

1. **模块3<** Length to Coordinate **>**

**3.1函数1**

**函数名称：**Position Length\_To\_Coordinate(TrainInfo a)

函数参数说明：火车运行长度

函数功能：将长度换为火车坐标。

函数局部变量设计： int mod;

Position train\_position;

函数算法逻辑：

求距离除以周长的余数

If余数大于0且小于等于长

X坐标=x+余数

Y坐标=y

if余数大于长且小于等于长+宽

余数-=长

X坐标=x+长

y坐标=y+余数

if余数大于长+宽且<=2倍的长+一倍的宽

余数=长+宽

X坐标=x+长-余数

y坐标=y+宽

else

余数=2倍的长+1倍的宽

X坐标=x

y坐标=y+宽-余数

返回坐标

1. **模块4<** TrainFile **>**

**4.1函数1**

**函数名称：**void SetDefaultSet(TrainInfo set[])

**函数参数说明：**默认数据集合

**函数功能：**默认数据输入

**函数局部变量设计：**无

**函数算法逻辑：**默认数据输入

**4.2函数2**

**函数名称：**void OutputSet(TrainInfo set[],int order)

**函数参数说明：**默认数据集合、A,B or C车

**函数功能：**输出运行速度，起始时间，运行方向，起始位置，停靠时间和类型

**函数局部变量设计：无**

**函数算法逻辑：**

switch(order)

case1 A车输出运行速度，起始时间，运行方向，起始位置，停靠时间和类型

case2 B车输出运行速度，起始时间，运行方向，起始位置，停靠时间和类型

case3 C车输出运行速度，起始时间，运行方向，起始位置，停靠时间和类型

**4.3函数3**

**函数名称：**void InputSet(TrainInfo set[],int order)

**函数参数说明：**默认数据集合、A,B or C车

**函数功能：**默认数据的输入，快车慢车的储存以及运行速度的判断

**函数局部变量设计：** char carname,tempinput;

int fasttypeset[3],slowtypeset[3];

int fasttype=0,slowtype=0,a;

int tempinputflag=0;

**函数算法逻辑：**

输入输入类型

判断慢车储存or 快车储存

输入运行速度

判断快车速度是否快于慢车

判断慢车速度是否慢于快车

输入运行方向(顺时针为1,逆时针为2)

输入起始位置

输入停靠时间

调用函数OutputSet(set,order);

getchar();

输入tempinput

1. **模块5<** EnsureInfo **>**

**5.1函数1**

**函数名称：**void EnsureInfo(TrainInfo\* train)

**函数参数说明：**火车信息

**函数功能：**输出信息（车号/轨道号。快慢，方向轨道基准点，启动位置，探测点，坐标，启动时间，速度，停靠时间，所在轨道）

**函数局部变量设计：**无

**函数算法逻辑：**

输出车号/轨道号

if当车的类型为1时 输出快车

if当车的类型为2时 输出慢车

if当车的方向为1时 输出顺时针

if当车的方向为2时 输出逆时针

输出轨道基准点坐标

输出启动位置

if((\*train).train\_num==1)输出探测点1、2、3、4及一维坐标

else输出探测点1、2及一维坐标

输出启动时间

输出火车速度

输出停靠时间

输出所在轨道

1. **模块6<** CompleteInfo **>**

**6.1函数1**

**函数名称：**void CompleteInfo(TrainInfo\* train1,TrainInfo\* train2,TrainInfo\* train3)

**函数参数说明：**3辆火车的初始信息

**函数功能：**指针指向重合轨道火车，确定正方向，初始位置确定，初始化状态

**函数局部变量设计：**int mod;

**函数算法逻辑：**

求余数

if余数大于等于0且小于等于长

二维坐标x=x+余数;

二维坐标y=y;

if余数大于长且小于等于长+宽

余数-=长;

二维坐标x=x+长;

二维坐标y=y+余数;

if余数大于长+宽且<=2倍的长+1倍的宽

余数-=长+宽;

二维坐标x=x+长-余数

二维坐标y=y+宽;

else

余数-=2倍的长+1倍的宽;

二维坐标x=x;

二维坐标y=y+宽-余数;

train2,train3同train1

判断火车方向信息

指针指向重合轨道火车

正方向为顺时针

初始位置确定

初始化状态

1. **模块7<** Finite State Machine **>**

**7.1函数1**

**函数名称：**void ControlTrain(TrainInfo\* train)

**函数参数说明：**火车信息

**函数功能：**修改状态，运行中，等待进站，在站台等待，进站，出站，执行操作，更新2维坐标

**函数局部变量设计：**无

**函数算法逻辑：**

If当火车运行状态为0时

If当ReadytoJudge(\*train)=1时

If当\*train).train\_num=1时

If火车距离>=L+W

if(ReadytoJudge((\*((\*train).tptr2)))==1)

if火车运行速度>tptr2火车运行速度

火车运行状态为3

Tptr2运行状态为1

else

火车运行状态为1

Tptr2运行状态为3

else

火车运行状态为3

if火车距离<L+W

if(ReadytoJudge((\*((\*train).tptr1)))==1)

火车运行状态为3

tptr1运行状态为1

else

火车运行状态为1;

tptr1运行状态为3

else

火车运行状态为3

else

if(ReadytoJudge((\*((\*train).tptr1)))==1)

if火车运行速度>tptr1火车速度

火车运行状态为3; tptr1运行状态为1

else

火车运行状态为1; tptr1运行状态为3

else 火车运行状态为3

else if火车状态为1

if当\*train).train\_num=1时

if火车距离>=L+W

if当tptr1运行状态为0时 运行状态为0

if火车距离<=L+W

if当tptr1运行状态为0时 运行状态为0

else

if当tptr1运行状态为0时 运行状态为0

if火车状态为2

if当火车停靠时间为0时运行状态为4

if火车状态为3

if火车方向为1

if当\*train).train\_num=1时

if当火车距离+火车速度>=1站台且当火车距离+火车速度>=2站台

站台2

else

if当火车距离+火车速度>=1站台

站台2

else

if当\*train).train\_num=1时

if当火车距离+火车速度<=1站台且当火车距离+火车速度<=2站台

站台2

else

if当火车距离+火车速度<=1站台

站台2

else if火车状态为4

1车顺时针

下半部分 火车运行状态为0

上半部分 火车运行状态为0

1车逆时针

上半部分 火车运行状态为0

下半部分 火车运行状态为0

2或3顺时针

i运行距离+速度>检测点2

火车运行状态为0

2或3逆时针

If运行距离+速度<检测点2

火车运行状态为0

If火车状态为1

(\*train).train\_speed=0;

一车顺时针上

1维火车距离=一维检测点\_1

一车顺时针下

1维火车距离=一维检测点\_1

一车逆时针上

1维火车距离=一维检测点\_1

一车逆时针下

1维火车距离=一维检测点\_1;

2车或3车

1维火车距离=一维检测点\_1;

在站台停着

If 1车

If当(\*train).train\_distance\_1D>=(\*train).train\_orbit\_L+(\*train).train\_orbit\_W时

1维火车距离=站台1

else

1维火车距离=站台2

火车速度=0;

停靠时间-=1;

进站

If1车 火车速度=speed\_temp1;

If2车 火车速度=speed\_temp2;

If3车 火车速度=speed\_temp3;

出站

If1车 火车速度=speed\_temp1;

If2车 火车速度=speed\_temp2;

If3车 火车速度=speed\_temp3;

If1车 停靠时间=time\_temp1;

If2车 停靠时间=time\_temp2;

If3车 停靠时间=time\_temp3;

不断更新next

if((\*train).train\_distance\_1D<0)

(\*train).train\_distance\_1D+=2\*(\*train).train\_orbit\_L+2\*(\*train).train\_orbit\_W;

更新2维坐标

**7.2函数2**

**函数名称：**int ReadytoJudge(TrainInfo a)

**函数参数说明：**火车运行长度

**函数功能：**判断下一秒是否进入探测区。

**函数局部变量设计：**无

**函数算法逻辑：**

If 1车情况

if当车的方向为1时

进入探测区，返回1

否则，返回0

if当车的方向为2时

进入探测区，返回1

否则，返回0

If 2、3车情况

if当车的方向为1时

if(a.train\_next>=a.train\_detect\_1D\_1||a.train\_next<=a.train\_detect\_1D\_2) 返回1

else

返回0

if当车的方向为2时

if(a.train\_next>=a.train\_detect\_1D\_2||a.train\_next<=a.train\_detect\_1D\_1) 返回1

else

返回0

1. **模块8<**File Write **>**

**8.1函数1**

**函数名称：**void FileWrite(TrainInfo set[],FILE \*fp,int temp)

**函数参数说明：**默认数据集合、文件输入、set(temp)

**函数功能：**文件输入

**函数局部变量设计：**无

**函数算法逻辑：**

fprintf火车车号、火车类型、轨道基准点坐标、启动位置

if(set[temp].train\_num==1)

fprintf探测点1、2、3、4及一维坐标

else

fprintf探测点1、2及一维坐标

fprintf启动时间、火车速度、停靠时间、所在轨道:长 宽

1. **模块9<** outputCli>

**9.1函数1**

**函数名称：** PrintTrainInfo(TrainInfo\* train,int OutputTime)

**函数参数说明：**火车信息，输出时间

**函数功能：**命令行输出：当前时间，当前策略，3辆火车运行状态，3辆火车当前位置，公共轨道的状态

**函数局部变量设计：**无

**函数算法逻辑：**

输出当前时间

输出 [A车]

输出运行方向

if当车的方向为1时 输出顺时针

else输出逆时针

输出运行状态

if当车的运行状态为0时 输出运行-非公共轨道

if当车的运行状态为1时 输出暂停-等待公共轨道

if当车的运行状态为2时 输出停靠-占用公共轨道(还需等待%ds)

if当车的运行状态为3时 输出f运行-公共轨道

if当车的运行状态为4时 输出运行-公共轨道

if当车的运行状态为5时 输出接受命令暂停等状态

输出当前位置

同理输出B,C车信息

输出占用轨道状态

FILE \*outputfile;

If当outputfile=fopen("输出数据.ini","a+")为空时

exit(0);

else 否则调用函数FileWriteOut(train,outputfile,OutputTime);

fflush(outputfile);

**9.2函数2**

**函数名称：**IsInPubluic(TrainInfo a)

**函数参数说明：**车辆信息

**函数功能：**判断是否占用公共轨道

**函数局部变量设计：**无

**函数算法逻辑：**

顺时针方向

占用返回1

否则返回0

逆时针方向

占用返回1

否则返回0

**10.模块10<** Thread>

**10.1函数1**

**函数名称：** TRAIN1RUN(LPVOID lpParam)

**函数参数说明：**地址参数

**函数功能：**火车1线程

**函数局部变量设计：**

**函数算法逻辑：**

TrainInfo \*train =(TrainInfo \*)lpParam;

while(1)进行循环

if当KeyofTrain1=1时 返回0;

clock\_t start,finish;

定义long long duration;

赋值start=clock();

调用函数ControlTrain(train);

赋值finish=clock();

赋值duration=finish-start;

输出1

Sleep(1000-duration)；

返回0;

**10.2函数2**

**函数名称：** TRAIN2RUN(LPVOID lpParam)

**函数参数说明：**地址参数

**函数功能：**火车2线程

**函数局部变量设计：**无

**函数算法逻辑：**

TrainInfo \*train =(TrainInfo \*)lpParam;

while(1) 进行循环

if当KeyofTrain1=1时 返回0;

clock\_t start,finish;

定义long long duration;

赋值start=clock();

调用函数ControlTrain(train);

赋值finish=clock();

赋值duration=finish-start;

输出2

Sleep(1000-duration)；

返回0;

**10.3函数3**

**函数名称：**TRAIN3RUN(LPVOID lpParam)

**函数参数说明：**地址参数

**函数功能：**火车3线程

**函数局部变量设计：**无

**函数算法逻辑：**

TrainInfo \*train =(TrainInfo \*)lpParam;

while(1) 进行循环

if当KeyofTrain1=1时 返回0;

clock\_t start,finish;

定义long long duration;

赋值start=clock();

调用函数ControlTrain(train);

赋值finish=clock();

赋值duration=finish-start;

输出3

Sleep(1000-duration)；

返回0;

**10.4函数4**

**函数名称：** TRAINCONTROL(LPVOID lpParam)

**函数参数说明：**地址参数

**函数功能：**用户挂起--挂起对应线程即可，无操作

**函数局部变量设计：** char ControlRun[999];

int tempRun=0;

int a=ControlRun[0],b=ControlRun[1];

**函数算法逻辑：**

**定义**char ControlRun[999];

定义int tempRun=0;

while(1)进行循环

输入gets(ControlRun);

**定义**int a=ControlRun[0],b=ControlRun[1];

If当a=80且b=65时 KeyofTrain1==1;

If当a=67且b=65时 KeyofTrain1==2;

If当a=80且b=66时 KeyofTrain2==1;

If当a=67且b=66时 KeyofTrain2==2;

If当a=80且b=67时 KeyofTrain3==1;

If当a=67且b=67时 KeyofTrain3==2;

**10.5函数5**

**函数名称：** OutPutFILENCMD(LPVOID lpParam)

**函数参数说明：**地址参数

**函数功能：**每秒输出，包括信息见要求的doc， 文件和命令行

**函数局部变量设计：**HANDLE Thread1,Thread2,Thread3,Thread4;

**函数算法逻辑：**

void AllControl(TrainInfo train[4])

HANDLE Thread1,Thread2,Thread3,Thread4;

TrianRunMutex1=CreateMutex(NULL,FALSE,"TEMP");

TrianRunMutex2=CreateMutex(NULL,FALSE,"TEMP");

TrianRunMutex3=CreateMutex(NULL,FALSE,"TEMP");

Thread1=CreateThread(NULL,0,TRAIN1RUN,&train[1],0,0);

Thread2=CreateThread(NULL,0,TRAIN2RUN,&train[2],0,0);

Thread3=CreateThread(NULL,0,TRAIN3RUN,&train[3],0,0);

Thread4=CreateThread(NULL,0,TRAINCONTROL,0,0,0);

CloseHandle(Thread1);

CloseHandle(Thread2);

CloseHandle(Thread3);

Sleep(100000);