电子信息与工程学院

姓名	学号	专业	联系方式
黄炜恺	17311023	微电子科学与工程	13725951349

大作业实验报告

一、分析与设计

1、需求分析

模拟一个只有一条跑道的飞机场，用泊松分布模拟每单位时间请求起飞或降落的飞机，创建一类等待队列，根据用户提供的最大等待队列长度判断接受还是拒绝请求，优先考虑飞机降落的情况，每单位时间判断一次跑道的状态是正在降落或起飞或闲置，最后统计出飞机数目及时间等量。

2、类结构分析

在 Plane 类中的私有数据成员记录飞机的航班号、起始时间、起降状态；在公有函数中有控制飞机行为的功能：拒绝起飞\降落、起飞\降落。

在 Runway 类中的私有数据成员有队列长度、时间等输出时要记录的数据；公有函数描述跑道的行为：判断能否起飞\降落、接受起飞\降落、对队列里的飞机进行起飞\降落操作及输出最终结果。

在 Extended_queue 类中的私有数据成员有队列长度、队头队尾位置、和以飞机类对象为元素的数组；公有函数的功能有：判断队列是否为空\满、出队、入队、取队头元素。

另外，声明 Runway 和 Extended_queue 是 Plane 的友元类，Runway 是 Extended_queue 的友元类，Plane 是 Runway 的友元类，使不同类的函数可以更方便被其他类调用。

3、设计细节

将 const 根据需求放在函数的相应位置防止代码出错。

用 setw（）控制输出长度。

用<<flush 防止程序发生意外中断而来不及送出数据。

cerr 不经过缓冲而直接输出，一般用于迅速输出出错信息，是标准错误，默认情况下被关联到标准输出流，但它不被缓冲，也就说错误消息可以直接发送到显示器，而无需等到缓冲区或者新的换行符时才被显示。

运用引用作函数参数实现对实参的改变。

4、代码

```cpp
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <time.h>
#include <stdlib.h>
#include <random>
#include <iomanip>
#include <string.h>
#include <iomanip>
using namespace std;

static double queue_limit;//队列的最大长度（起飞、降落）

enum Runway_activity {Idle, Land, Takeoff};
enum Plane_status{null, arriving, departing};
enum Error_code{success,underflow,overflow};

//泊松分布随机数
std::default_random_engine generator;
int poisson(double x)
{
  std::poisson_distribution<int> distribution(x);
  int number = distribution(generator);
  return number;
}

//检查输入是否正确
void EnterNum(double& i)
{
    label:
    while(!(cin >> i))
    {
      cin.clear();
      cin.sync();
      cerr << "ERROR !!! Please enter an nonnegative integer number :";
    }
    int a = i;
    if (a != i || i <= 0 )
    {
      cerr << "ERROR !!! Please enter an nonnegative integer number :";
      goto label;
    }
}
```

```
44
45  template <typename T> class Extended_queue;
46  class Runway;
47
48  class Plane{
49  public:
50      Plane();
51     Plane(int flt, int time, Plane_status status);
52     void refuseland(Runway &) const;
53     void refusetakeoff(Runway &) const;
54     void land(int time) const;
55     void fly(int time) const;
56
57     friend class Runway;
58     template <typename T> friend class Extended_queue;//声明两个友元类 它们可以调用本类的成员
59
60  private:
61     int flt_num;//航班号
62     int clock_start;//计时器 标注飞机到达机场的时刻
63     Plane_status state;//飞机状态
64  };
65
66  template <typename T> class Extended_queue
67  {
68   public:
69     Extended_queue();
70    bool full()const;
71    bool empty() const;
72    Error_code serve();
73    Error_code append(const T &item);
74    Error_code retrieve(T &item) const;
75    friend class Runway;
76
77   private:
78    int size;//队列最大长度
79    int front;//队头位置
80    int rear;//队尾位置
81    T entry[15];//数组最大长度为15
82  };
```

```
class Runway
{
    public:
        Runway(int limit);
        Error_code can_land(const Plane& current);
        Error_code can_depart(const Plane& current);
        Runway_activity activity(int time, Plane &moving);
        void shut_down(int time) const;
        void accept_land (Plane &plane);
        void accept_takeoff (Plane &plane);
        friend class Plane;
    private:
        Extended_queue <Plane> landing;
        Extended_queue <Plane> takeoff;
        int queue_limit;//队列的最大长度
        int num_land_requests=0; //要求降落的飞机数目
        int num_takeoff_requests=0;//要求起飞的飞机数目
        int num_landing=0;//已降落的飞机数目
        int num_takeoffs=0;//已起飞的飞机数目
        int num_land_accepted=0;//在降落队列里的飞机数目
        int num_takeoff_accepted=0;//在起飞队列里的飞机数目
        int num_land_refused=0;//被拒绝的要降落飞机数目
        int num_takeoff_refused=0;//被拒绝的要起飞飞机数目
        int land_wait=0; //飞机等待降落的总时间
        int takeoff_wait=0;//飞机等待起飞的总时间
        int idle_time=0;//机场处于空闲状态的总时间
};

//构造函数
Plane::Plane()
{
    flt_num = 0;
    clock_start = 0;
    state = null;
}

Plane::Plane(int flt, int time, Plane_status status):flt_num(flt),clock_start(time),state(status){}

//拒绝降落
void Plane::refuseland(Runway &runway) const
{
    cout << "    Plane number " << setw(3) << left << flt_num << " told to try to land again later" << endl ;
    runway.num_land_refused++;
}
```

```

//拒绝起飞
void Plane::refusetakeoff(Runway &runway) const
{
    cout << "    Plane number " << setw(3) << left << flt_num << " told to try to takeoff again later" << endl ;
    runway.num_takeoff_refused++;
}

//飞机降落
void Plane::land(int time) const
{
    cout << "Plane number " << setw(3) << left << flt_num << " landed after " << (time - clock_start) << " time units in the land queue" << endl;
}

//飞机起飞
void Plane::fly(int time) const
{
    cout << "Plane number " << setw(3) << left << flt_num << " took off after " << (time - clock_start) << " time units in the takeoff queue" << endl;
}

//初始化
template <typename T>
Extended_queue <T>::Extended_queue()
{
    size = 0;
    front = 0 ;
    rear = 0;
}


//检查是否满
template <typename T>
bool Extended_queue<T>::full()const
{
    return (size == queue_limit) ? true : false;
}


//检查是否空
template <typename T>
bool Extended_queue <T> ::empty() const
{
    return (size == 0) ? true : false;
}
```

```
172
173  //出队
174  template <typename T>
175  Error_code Extended_queue <T> :: serve ()
176  {
177      if (size)
178      {
179          delete &entry[front] ;
180          front = ((front == queue_limit-1) ? 0 : (front + 1));
181          size--;
182          return success;
183      }
184      else
185          return underflow;
186  }
187
188  //入队
189  template <typename T>
190  Error_code Extended_queue <T> :: append (const T&item)
191  {
192      if(size >= queue_limit)
193          return overflow;
194      size++;
195      entry [rear] = item;
196      rear = ((rear+1) == queue_limit) ? 0 : (rear+1);
197      return success;
198  }
199
200  //取队头元素
201  template <typename T>
202  Error_code Extended_queue <T> :: retrieve(T&item) const
203  {
204      if (size == 0)
205          return underflow;
206      item = entry[front];
207      return success;
208  }
209
210   Runway::Runway(int limit) : queue_limit(limit){}
```

```
211
212  //判断是否可以降落
213   Error_code Runway::can_land(const Plane& current)
214   {
215    cout << "    Plane number " << setw(3) << left << current.flt_num << " ready to land" <<  endl;
216    num_land_requests++;
217    if (landing.full())
218      return overflow;
219    else
220      return success;
221   }
222
223  //判断是否可以起飞
224  Error_code Runway::can_depart(const Plane& current)
225  {
226    cout << "    Plane number " << setw(3) << left << current.flt_num << " ready to take off" << endl;
227    num_takeoff_requests++;
228  if (takeoff.full())
229      return overflow;
230    else
231      return success;
232  }
233
234  //可以进降落队列
235  void Runway::accept_land (Plane &plane)
236  {
237      landing.append(plane);
238      num_land_accepted++;
239  }
240
241  //可以进起飞队列
242  void Runway::accept_takeoff (Plane &plane)
243  {
244      takeoff.append(plane);
245      num_takeoff_accepted++;
246  }
```

```
247
248    //操作队列里的飞机 优先判断降落队列
249   Runway_activity Runway::activity(int time, Plane &moving)
250   {
251       if(!landing.empty())
252         {
253           landing.retrieve(moving);
254           landing.serve();
255           num_landing++;
256           land_wait+=time-moving.clock_start;
257           return Land;
258         }
259       else if(!takeoff.empty())
260         {
261           takeoff.retrieve(moving);
262           takeoff.serve();
263           num_takeoffs++;
264           takeoff_wait+=time-moving.clock_start;
265           return Takeoff;
266         }
267       else
268       {
269         idle_time++;
270         return Idle;
271       }
272
273   }
```

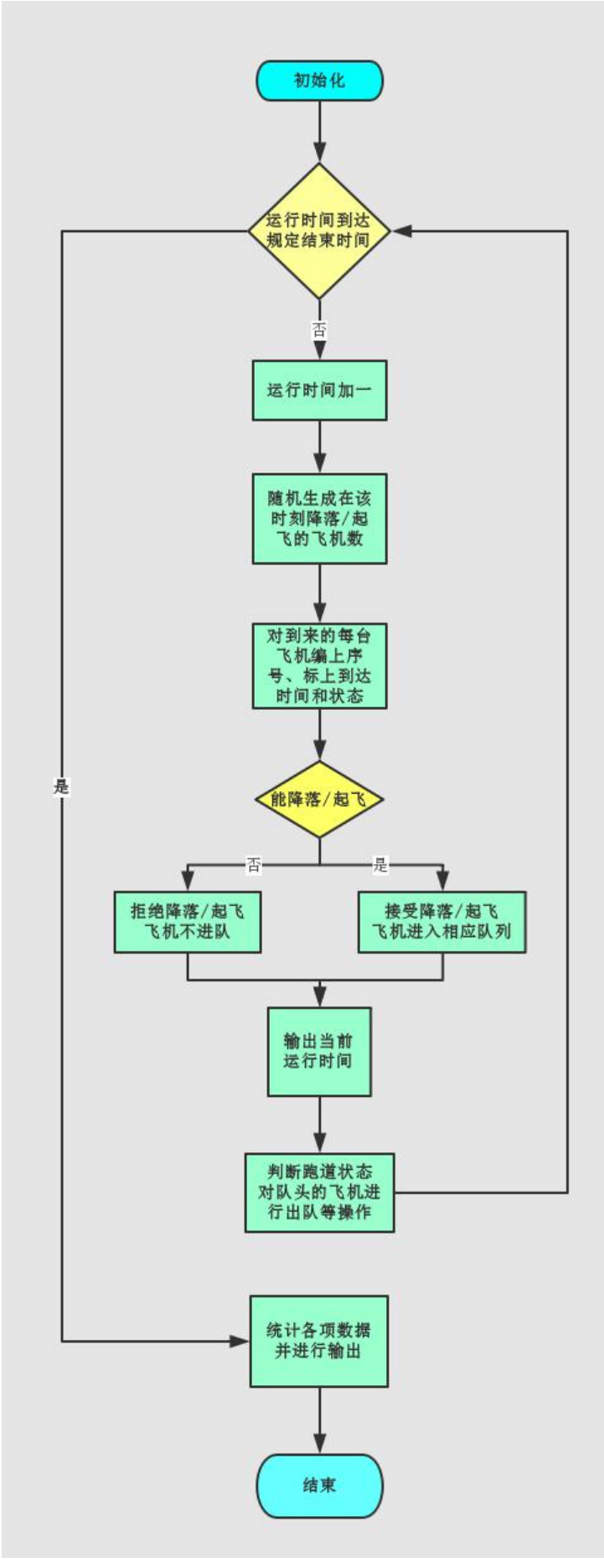
```
274
275  //跑道数据在这里总结计算并打印出来
276   void Runway::shut_down(int time) const
277         {
278
279                cout<< "Simulation has concluded after  "<<time<<"  time units"<<endl
280
281
282                      <<"Total number of planes processed                      " //处理的飞机总数
283
284                      <<(num_land_requests+num_takeoff_requests)<<endl
285
286                      <<"Total number of planes asking to land                 "//要求着陆的飞机总数
287                      << num_land_requests<<endl
288
289                      <<"Total number of planes asking to take off             "//要求起飞的飞机总数
290                      << num_takeoff_requests<<endl
291
292                      <<"Total number of planes accepted for landing           "//接受着陆的飞机总数
293                      << num_land_accepted<<endl
294
295                      <<"Total number of planes accepted for takeoff           "//接受起飞的飞机总数
296                      << num_takeoff_accepted<<endl
297
298                      <<"Total number of planes refused for landing            "//拒绝着陆的飞机总数
299                      << num_land_refused<<endl
300
301                      <<"Total number of planes refused for takeoff            "//拒绝起飞的飞机总数
302                      << num_takeoff_refused<<endl
303
304                      <<"Total number of planes that landed                    "//降落的飞机总数
305                      << num_landing<<endl
306
307                      <<"Total number of planes that took off                  "//起飞的飞机总数
308                      << num_takeoffs<<endl
309
310                      <<"Total number of planes left in landing queue          "//还留在起飞队列里的飞机总数
311                      << landing.size<<endl
312
313                      <<"Total number of planes left in takeoff queue          "//还留在降落队列里的飞机总数
314                      << takeoff.size<<endl;
```

```
316               cout     <<"Percentage of time runway idle                      "
317               << 100.0 * ((float) idle_time)/((float) time) << "%" << endl;//跑道闲置时间的百分比
318
319               cout     <<"Average wait in landing queue                       "
320               << ((float) land_wait)/((float) num_landing) << " time units" << endl;//平均在降落队列里等待的时间
321
322               cout     <<"Average wait in takeoff queue                       "
323               << ((float) takeoff_wait)/((float) num_takeoffs)<< " time units" << endl;//平均在起飞队列里等待的时间
324
325               cout     <<"Average observed rate of planes wanting to land     "
326               << ((float) num_land_requests)/((float) time)<< " per time unit" << endl;//想要降落的飞机的平均观察率
327
328               cout     <<"Average observed rate of planes wanting to take off "
329               << ((float) num_takeoff_requests)/((float) time)<< " per time unit" << endl;//希望起飞的飞机的平均观察率
330         }
```

```cpp

//用户在此声明仿真所需的time units（时间单位）的数量，队列的最大长度和机场的平均到达率和离开率。
void initialize(double &end_time, double &queue_limit, double &arrival_rate, double&departure_rate)
{
    cout << "This program simulates an airport with only one runway." << endl
    << "One plane can land or depart in each unit of time." << endl;
    cout << "Up to what number of planes can be waiting to land "
    << "or take off at any time? " << flush;
    EnterNum(queue_limit);
    cout << "How many units of time will the simulation run?" << flush;
    EnterNum(end_time);

    bool acceptable;
    do {
    cout << "Expected number of arrivals per unit time?" << flush;
    cin >> arrival_rate;
    cout << "Expected number of departures per unit time?" << flush;
    cin >> departure_rate;
    if (arrival_rate < 0.0 || departure_rate < 0.0)
    cerr << "These rates must be nonnegative." << endl;
    else
    acceptable = true;
    if (acceptable && arrival_rate + departure_rate > 1.0)
    cerr << "Safety Warning: This airport will become saturated. " << endl;
    } while (!acceptable);
}
```

```cpp

//跑道闲置
void run_idle(int &time)
{
  cout << "Runway is Idle. " << endl;
}

int main()
{
    double end_time;//运行总时间单位
    //int queue_limit;//队列的最大长度
    double flight_number = 0;//初始航班号为0
    double arrival_rate, departure_rate;
    initialize(end_time,queue_limit,arrival_rate,departure_rate);
    Runway small_airport(queue_limit);
    for (int current_time = 0; current_time<end_time;current_time++)
     {
        int number_arrivals = poisson(arrival_rate);//随机生成该时刻将会降落的飞机数
        for (int i = 0; i < number_arrivals; i++)
            {
                Plane current_plane(flight_number++, current_time, arriving);
                if (small_airport.can_land(current_plane) == success)
                    small_airport.accept_land (current_plane);
                else
                    current_plane.refuseland(small_airport);
            }
        int number_departures = poisson(departure_rate);//随机生成该时刻将会起飞的飞机数
        for (int j = 0; j<number_departures; j++)
            {
                Plane current_plane(flight_number++, current_time, departing);
                if (small_airport.can_depart(current_plane) == success)
                    small_airport.accept_takeoff (current_plane);
                else
                    current_plane.refusetakeoff(small_airport);
            }
        cout << setw(3) << left << current_time << ":" ;
        Plane moving_plane;//准备操作的飞机???
        switch (small_airport.activity(current_time,moving_plane))
        {
            case Land:
                moving_plane.land(current_time);
                break;
            case Takeoff:
                moving_plane.fly(current_time);
                break;
            case  Idle:
                run_idle(current_time);
                break;
        }
    }
    small_airport.shut_down(end_time);
    return 0;
 }
```

5、流程图


初始化
运行时间到达规定结束时间
否
运行时间加一
随机生成在该时刻降落/起飞的飞机数
对到来的每台飞机编上序号、标上到达时间和状态
能降落/起飞
否
是
拒绝降落/起飞飞机不进队
接受降落/起飞飞机进入相应队列
输出当前运行时间
判断跑道状态对队头的飞机进行出队等操作
是
统计各项数据并进行输出
结束

6、实验结果

```
This program simulates an airport with only one runway.
One plane can land or depart in each unit of time.
Up to what number of planes can be waiting to land or take off at any time? -5
ERROR !!! Please enter an nonnegative integer number :5.5
ERROR !!! Please enter an nonnegative integer number :asd
ERROR !!! Please enter an nonnegative integer number :5
How many units of time will the simulation run?1000
Expected number of arrivals per unit time?0.48
Expected number of departures per unit time?0.48
0  :Runway is Idle.
    Plane number 0   ready to take off
    Plane number 1   ready to take off
1  :Plane number 0   took off after 0 time units in the takeoff queue
2  :Plane number 1   took off after 1 time units in the takeoff queue
3  :Runway is Idle.
    Plane number 2   ready to land
    Plane number 3   ready to land
    Plane number 4   ready to take off
4  :Plane number 2   landed after 0 time units in the land queue
    Plane number 5   ready to land
5  :Plane number 3   landed after 1 time units in the land queue
    Plane number 6   ready to land
    Plane number 7   ready to take off
    Plane number 8   ready to take off
6  :Plane number 5   landed after 1 time units in the land queue
    Plane number 9   ready to land
7  :Plane number 6   landed after 1 time units in the land queue
8  :Plane number 9   landed after 1 time units in the land queue
    Plane number 10  ready to land
9  :Plane number 10  landed after 0 time units in the land queue
```

```
989:Plane number 933 landed after 0 time units in the land queue
990:Runway is Idle.
991:Runway is Idle.
    Plane number 934 ready to land
    Plane number 935 ready to land
    Plane number 936 ready to land
    Plane number 937 ready to take off
992:Plane number 934 landed after 0 time units in the land queue
993:Plane number 935 landed after 1 time units in the land queue
    Plane number 938 ready to land
    Plane number 939 ready to land
994:Plane number 936 landed after 2 time units in the land queue
995:Plane number 938 landed after 1 time units in the land queue
996:Plane number 939 landed after 2 time units in the land queue
997:Plane number 937 took off after 5 time units in the takeoff queue
998:Runway is Idle.
999:Runway is Idle.
Simulation has concluded after  1000  time units
Total number of planes processed                        940
Total number of planes asking to land                   448
Total number of planes asking to take off               492
Total number of planes accepted for landing             447
Total number of planes accepted for takeoff             447
Total number of planes refused for landing              1
Total number of planes refused for takeoff              45
Total number of planes that landed                      447
Total number of planes that took off                    447
Total number of planes left in landing queue            0
Total number of planes left in takeoff queue            0
Percentage of time runway idle                          10.6%
Average wait in landing queue                           0.463087 time units
Average wait in takeoff queue                           4.13423 time units
Average observed rate of planes wanting to land         0.448 per time unit
Average observed rate of planes wanting to take off     0.492 per time unit

Process returned 0 (0x0)   execution time : 18.791 s
Press any key to continue.
```

7、心得体会

在实验前必须明确各个类的功能，使用友元类或友元函数可以进行类之间的灵活调用，不过会损失安全性。

对于代码冗长的程序可以用分步的办法提高效率，如先实现主要功能再处理输出数据，先把相对独立的功能写好测试成功后再放入主版本（如泊松分布函数和检验输入函数），先写出各类的主要功能再合并测试等。

灵活使用枚举变量可以增强程序的可读性。