

# Project 2

## 飞机场模拟

班级：2016 级软工程教务三班

欧穗新      16340173

欧阳梓轩   16340174

潘鉴        16340176

彭靖寒      16340178

## 【题目要求】

问题:

考虑一个繁忙的机场这个机场只有一个飞机跑道, 每个单位时间当中只有一架飞机可以降落, 或者只有一架飞机可以起飞, 但是不允许同时起飞和降落;

飞机的到达和起飞是随机的, 因此在任何时候, 飞机跑道有可能是空的, 或者是有一架飞机正在起飞或者着陆, 并且同时可以有若干个飞机正在等待起飞或着陆;

请编写程序模拟飞机场的起飞降落调度;

具体要求 (书本中已经给出 `random` 类、`plane` 类、`runway` 类以及主函数代码):

- P1. 将所有用于飞机场模拟的函数和方法组合成一个完整的程序。用飞机场模拟程序做若干次试运行实验, 调整准备着陆和起飞的飞机数的期望值, 并找出在飞机不会被拒绝服务的条件下这些数字的尽可能大的近似值。如果队列的长度增加或减少, 那么这些值将会有何变化?
- P2. 修改模拟程序, 使飞机场有两条飞机跑道, 其中一条总是用于着陆, 另一条总是用于起飞。比较双跑道机场能服务的总飞机数和单条飞机跑道的飞机场的相应数字, 前者是否是后者的两倍?
- P3. 修改模拟程序, 使飞机场有两条飞机跑道, 其中一条总是用于着陆, 另一条总是用于起飞。如果某个队列是空的, 那么两条跑道都能用于其他的队列。如果着陆队列总是满的并且另一架飞机要到达着陆, 那么将停止起飞, 并将两条跑道都用于清理搁置的着陆飞机。
- P4. 修改模拟程序, 使飞机场有 3 条飞机跑道, 其中各保留一条总用于着陆和起飞, 第三条用于着陆, 但在着陆队列为空的情况下, 第三条亦可用于起飞。
- P5. 修改最初的模拟程序(单条跑道), 使得当每架飞机到达着陆时, 它将有(作为它的数据成员的)一个(随机产生的)油位, 以剩余的时间单元度量。如果飞机没有足够的油位在队列中等待, 则允许它立即着陆。因此着陆队列里的飞机可能需要再等待附加的单元, 因此可能用完自身的燃料。作为着陆函数的一部分检查这一点, 并且查明在飞机由于燃料用尽而开始坠毁前机场有多忙。
- P6. 写一个占位程序来代替随机数函数, 这个占位程序既能用于调试程序又允许用户正确地控制每一个时间单元内每个队列到达的飞机数。

## 【数据结构与算法】

### 1. 数据结构:

Plane 类:

数据:

Int flt\_num = 0; (表示这个飞机的航班号)

Int clock\_start = 0; (表示这个飞机发出请求的时间)

Int plane\_status = 0; (表示这个飞机的状态, 有“请求降落”、“请求起飞”、“NULL”三个状态)

#### 方法:

- (1) 默认构造函数 plane() 以及传参数构造函数 plane(flt, time, plane\_status), 参数依次表示航班号, 请求时间/到达机场时间, 飞机状态;
- (2) refuse() 函数, 当等待起飞/降落的飞机队列达到该队列长度限制的时候, 拒绝最近的起飞/降落请求, 并且要求该飞机前往另一个机场;
- (3) land(int time) 函数, 在 time 时间点, 完成一架飞机的着陆请求;
- (4) fly(int) 函数, 在 time 时间点, 完成一架飞机的起飞前请求;
- (5) int started() 函数, 返回一个 int 数字表示这个飞机在当前机场发出请求的时间;

Runway 类:

#### 数据:

```
queue<Plane> landings; 飞机场的着陆队列;
queue<Plane> takeoffs; 飞机场的起飞队列;
queue_limit;
int num_land_requests; //请求着陆的飞机
int num_takeoff_requests; //请求起飞的飞机
int num_land_finished; //已经在这个机场完成着陆的飞机
int num_takeoff_finished; //已经在这个机场完成起飞的飞机
int num_land_accepted; //已经被接受但是在等待着陆的飞机
int num_takeoff_accepted; //已经被接受但是在等待起飞的飞机
int num_land_refused; //被拒绝着陆要求的飞机
int num_takeoff_refused; //被拒绝起飞要求的飞机
int land_wait_total; //所有飞机等待着陆的总时间
int takeoff_wait_total; //所有飞机等待起飞的总时间
int idle_time_total; //跑道闲置的总时间
```

#### 方法:

- (1) 构造函数 Runway(int limit) 接受一个 limit (等待起飞/降落队列的最大长度), 对队列进行初始化;
- (2) Error\_code can\_land(Plane &current); 是否允许飞机 current 着陆函数, 如果等待队列还未满, 则允许, 否则不允许; 记录相关请求信息、通过/拒绝请求信息

- (3) `Error_code can_depart(const Plane &current);` 是否允许飞机 `current` 起飞函数, 如果等待队列还未满, 则允许, 否则不允许; 记录相关请求信息、通过/拒绝请求信息
- (4) `Runway_activity activity(int time, Plane &moving);` 调度当前时单位的跑道, 如果等待着陆队列没空则完成队列首元素的着陆, 否则, 如果等待起飞队列没空则完成队列首元素的起飞, 否则, 这个单位时间内跑道为空; 记录相关通过请求 (起飞/着陆) 信息, 记录该飞机等待时间;
- (5) `void shut_down(int time) const;` 当 `time` 达到调度结束时间, 结束程序, 打印一系列跑道调度信息;

Random 类:

#### 数据:

`int seed;` (伪随机数的种子);  
`int multiplier, add_on;` (用来执行算数操作的常数);

#### 方法:

- (1) 构造函数 `Random(int)` 接受一个参数, 如果是 1, 则把种子置为 1, 表示用户希望使用一系列可再生的伪随机数; 如果不是 1, 则生产一个不可再生的伪随机数;
- (2) `Int Reseed ()` 函数, 再生产种子, 并且返回;
- (3) `Double random_real ()` 将来自 `reseed` 的结果转到一个希望的域上, 并且返回这个转换后的结果
- (4) `int poisson(double mean)` 函数, 接受一个泊松分布的期望值 `mean`, 返回一个以 `mean` 为期望的泊松分布序列中的一个数 (随机的);

## 2. 算法:

#### 初始化:

`end_time = 0;` (总调度时间)  
`queue_limit = 0;` (等待队列最大长度)  
`flight_number = 0;` (航班号/请求号)  
`arrival_rate = 0, departure_rate = 0;` (单位时间着陆/起飞请求数)  
 打印交互信息并要求用户输入所需调度时间、队列大小、单位时间着陆/起飞请求数: `intro_and_ini(end_time, queue_limit, arrival_rate, departure_rate);`  
`Random variable;` (初始化伪随机数);

Runway small\_airport(queue\_limit); (初始化跑道);

**从时间 0 开始到 end\_time 进行循环调度跑道:**

从时间 0 到时间 end\_time 进行遍历 {

产生该时间点发出着陆请求的飞机数目 number\_arrivals;

遍历 number\_arrivals 个请求 {

Plane current\_plane(flight\_number++, current\_time, arriving);

(初始化当前发出着陆请求的飞机)

if (等待着陆的飞机队列未满) {

记录相关请求信息、通过/拒绝请求信息;

拒绝该飞机的着陆请求 (要求该飞机前往其他机场着陆);

}

}

产生该时间点发出起飞请求的飞机数目 number\_departures;

遍历 number\_departures 个请求 {

Plane current\_plane(flight\_number++, current\_time, departing);

(初始化当前发出起飞请求的飞机)

if (等待起飞的飞机队列未满) {

记录相关请求信息、通过/拒绝请求信息;

拒绝该飞机的起飞请求; (要求该飞机等待一段时间后再重新请求)

}

}

初始化当前时间点需要处理的飞机 moving\_plane;

获取当前跑道的状态 (如果等待着陆队列没空则完成队列首元素的着陆, 并 pop 首元素, 跑道状态为完成着一个陆请求; 否则, 如果等待起飞队列没空则完成队列首元素的起飞, 并 pop 首元素, 跑道状态为完成一个起飞请求; 否则, 这个单位时间内跑道状态为空);

If (跑道状态为空) {

打印跑道为空的提示信息;;

} else if (跑道正在处理着陆) {

完成飞机 moving\_plane 的着陆;

打印着陆信息 (时间点, 等待时间等);

} else if (跑道正在处理起飞) {

完成飞机 moving\_plane 的起飞;

打印起飞学习（时间点，等待时间等）；

}

}

结束程序，打印一系列跑道调度信息（被处理的飞机数，花费在等待上的平均时间，被拒绝服务的飞机数目等等）；

## 【测试数据、结果及分析】

P1: (由于实际使用的测试数据太大，行数太多，所以只能给出调度总时间较少的测试样例相关输入输出)

### 1. 编译运行，并输入测试样例

C:\Users\14735\Desktop\exercise for c++\project2--big pro>a

This programe simulates an airport with only one runway.

Up to what number of planes can be waiting to land or take off at any time? 5

How many units of time will the simulation run? 10

Expected number of arrivals per unit time? 0.5

Expected number of departures per unit time? 0.5

### 2. 输出：打印每个时间点等待队列情况，跑道调度情况，以及程序结束时跑道的调度情况

```
选择命令提示符
How many units of time will the simulation run? 10
Expected number of arrivals per unit time? 0.5
Expected number of departures per unit time? 0.5
Plane number 0 ready to take off.
At time 0: Plane number 0 takes off after 0 time units int the takeoffs queue.
Plane number 1 ready to land.
Plane number 2 ready to land.
At time 1: Plane number 1 landed after 0 time units int the landings queue.
Plane number 3 ready to land.
At time 2: Plane number 2 landed after 1 time unit int the landings queue.
Plane number 4 ready to take off.
At time 3: Plane number 3 landed after 1 time unit int the landings queue.
At time 4: Plane number 4 takes off after 1 time unit int the takeoffs queue.
At time 5: the runway is idle
Plane number 5 ready to take off.
At time 6: Plane number 5 takes off after 0 time units int the takeoffs queue.
At time 7: the runway is idle
Plane number 6 ready to take off.
Plane number 7 ready to take off.
At time 8: Plane number 6 takes off after 0 time units int the takeoffs queue.
Plane number 8 ready to take off.
At time 9: Plane number 7 takes off after 1 time unit int the takeoffs queue.

Simulation has concluded after 10 time units.
Total number of planes processed during the Simulation: 9.
Total number of planes asking to land: 3
Average rate of plane wanting to land: 0.3
Total number of planes asking to takeoff: 6
Average rate of plane wanting to takeoff: 0.6

Total number of planes refused to land: 0
Average rate of planes refused to land: 0

Total number of planes refused to takeoff: 0
Average rate of planes refused to takeoff: 0

Total number of planes accepted to land: 3
Total number of planes accepted to takeoff: 6
Total number of planes landed successfully: 3
Total number of planes takeoff successfully: 5
Total time all the planes wait(for landing): 2
Average time that every planes wait(for landing): 0.666667

Total time all the planes wait(for taking off): 2
Average time that every planes wait(for taking off): 0.4

Total time the Runway idle: 2
```

统计结果：

由于概率小于 0.01 时认为是小概率事件，我们把拒绝（起飞/着陆）概率低于 0.01 记为不被拒绝（概率则简记为 0），对一些不同前提下的实验进行数据统计，得到如下数据：（在下面所有的三张图中，蓝色表示没有被请求拒绝的飞机，黄色部分表示存在被拒绝起飞请求但是不存在被拒绝的着陆请求，灰色表示存在被拒绝着陆但是不存在被拒绝的起飞请求）

- (1) 当队列长度为 3，进行 10000 次实验，起飞/降落请求概率从 0.1-1.0 进行改变，得到如下数据：（其中 B 代表起飞请求率，A 代表着陆请求率，C 代表被拒绝着陆的概率，D 代表被拒绝起飞的概率）

A\B	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.0	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.008	C:0.000 D:0.018	C:0.000 D:0.032	C:0.000 D:0.050	C:0.000 D:0.082	C:0.000 D:0.115	C:0.000 D:0.153
0.1	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.002	C:0.000 D:0.008	C:0.000 D:0.014	C:0.000 D:0.033	C:0.000 D:0.060	C:0.000 D:0.087	C:0.000 D:0.127	C:0.000 D:0.163	C:0.000 D:0.213
0.2	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.002	C:0.000 D:0.014	C:0.000 D:0.033	C:0.000 D:0.059	C:0.000 D:0.099	C:0.000 D:0.138	C:0.000 D:0.186	C:0.001 D:0.230	C:0.001 D:0.281
0.3	C:0.004 D:0.000	C:0.003 D:0.001	C:0.003 D:0.008	C:0.003 D:0.029	C:0.003 D:0.063	C:0.004 D:0.095	C:0.003 D:0.141	C:0.002 D:0.189	C:0.003 D:0.250	C:0.003 D:0.298	C:0.003 D:0.350
0.4	C:0.007 D:0.000	C:0.007 D:0.002	C:0.008 D:0.030	C:0.007 D:0.065	C:0.008 D:0.102	C:0.007 D:0.157	C:0.006 D:0.210	C:0.008 D:0.269	C:0.009 D:0.328	C:0.008 D:0.384	C:0.008 D:0.432
0.5	C:0.018 D:0.000	C:0.017 D:0.006	C:0.017 D:0.047	C:0.016 D:0.095	C:0.017 D:0.159	C:0.021 D:0.227	C:0.018 D:0.304	C:0.017 D:0.366	C:0.017 D:0.426	C:0.014 D:0.472	C:0.017 D:0.515
0.6	C:0.032 D:0.000	C:0.034 D:0.017	C:0.035 D:0.080	C:0.031 D:0.155	C:0.032 D:0.234	C:0.034 D:0.314	C:0.031 D:0.398	C:0.029 D:0.438	C:0.032 D:0.495	C:0.029 D:0.536	C:0.030 D:0.576
0.7	C:0.051 D:0.000	C:0.049 D:0.022	C:0.051 D:0.122	C:0.047 D:0.229	C:0.050 D:0.316	C:0.054 D:0.405	C:0.048 D:0.481	C:0.054 D:0.538	C:0.050 D:0.589	C:0.049 D:0.627	C:0.048 D:0.664
0.8	C:0.078 D:0.000	C:0.077 D:0.053	C:0.075 D:0.201	C:0.074 D:0.319	C:0.076 D:0.419	C:0.076 D:0.508	C:0.072 D:0.583	C:0.073 D:0.637	C:0.074 D:0.671	C:0.075 D:0.711	C:0.072 D:0.734
0.9	C:0.111 D:0.000	C:0.111 D:0.105	C:0.110 D:0.287	C:0.108 D:0.442	C:0.110 D:0.536	C:0.110 D:0.614	C:0.111 D:0.673	C:0.106 D:0.725	C:0.105 D:0.754	C:0.104 D:0.769	C:0.108 D:0.793
1.0	C:0.145 D:0.000	C:0.144 D:0.181	C:0.144 D:0.411	C:0.144 D:0.554	C:0.145 D:0.645	C:0.143 D:0.701	C:0.149 D:0.749	C:0.150 D:0.788	C:0.152 D:0.811	C:0.149 D:0.831	C:0.146 D:0.848

由上面的图看出，在没有被拒绝的飞机的情况下，

- A. 尽可能大的着陆请求率为：1，
- B. 起飞请求率为：1；
- C. 若使得二者都尽可能大，方差尽可能小，则取着陆请求 0.3，起飞请求率 0.2；

(2) 当队列长度为 5，进行 10000 次实验，起飞/降落请求概率从 0.1-1.0 进行改变，得到如下数据：（其中 B 代表起飞请求率，A 代表着陆请求率,C 代表被拒绝着陆的概率，D 代表被拒绝起飞的概率）

A\B	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.0	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.002	C:0.000 D:0.007	C:0.000 D:0.016	C:0.000 D:0.035	C:0.000 D:0.062	C:0.000 D:0.101
0.1	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.001	C:0.000 D:0.001	C:0.000 D:0.003	C:0.000 D:0.011	C:0.000 D:0.033	C:0.000 D:0.061	C:0.000 D:0.106	C:0.000 D:0.158
0.2	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.001	C:0.000 D:0.002	C:0.000 D:0.003	C:0.000 D:0.013	C:0.000 D:0.035	C:0.000 D:0.079	C:0.000 D:0.123	C:0.000 D:0.180	C:0.000 D:0.237
0.3	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.002	C:0.000 D:0.005	C:0.000 D:0.018	C:0.000 D:0.032	C:0.000 D:0.070	C:0.000 D:0.129	C:0.000 D:0.195	C:0.000 D:0.257	C:0.000 D:0.317
0.4	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.004	C:0.000 D:0.018	C:0.000 D:0.040	C:0.000 D:0.076	C:0.000 D:0.132	C:0.000 D:0.221	C:0.000 D:0.284	C:0.000 D:0.355	C:0.000 D:0.409
0.5	C:0.000 D:0.000	C:0.000 D:0.000	C:0.000 D:0.008	C:0.001 D:0.037	C:0.001 D:0.090	C:0.001 D:0.162	C:0.001 D:0.250	C:0.002 D:0.325	C:0.001 D:0.389	C:0.001 D:0.450	C:0.001 D:0.495
0.6	C:0.005 D:0.000	C:0.005 D:0.000	C:0.005 D:0.021	C:0.005 D:0.077	C:0.005 D:0.166	C:0.006 D:0.250	C:0.004 D:0.339	C:0.005 D:0.433	C:0.005 D:0.502	C:0.004 D:0.550	C:0.004 D:0.588
0.7	C:0.010 D:0.000	C:0.011 D:0.005	C:0.009 D:0.066	C:0.009 D:0.175	C:0.010 D:0.299	C:0.011 D:0.393	C:0.010 D:0.476	C:0.012 D:0.558	C:0.011 D:0.613	C:0.011 D:0.656	C:0.015 D:0.687
0.8	C:0.025 D:0.000	C:0.027 D:0.049	C:0.023 D:0.188	C:0.020 D:0.341	C:0.022 D:0.478	C:0.022 D:0.559	C:0.021 D:0.620	C:0.025 D:0.680	C:0.026 D:0.727	C:0.029 D:0.753	C:0.026 D:0.774
0.9	C:0.051 D:0.000	C:0.050 D:0.125	C:0.051 D:0.372	C:0.045 D:0.519	C:0.044 D:0.631	C:0.045 D:0.692	C:0.041 D:0.746	C:0.044 D:0.782	C:0.042 D:0.797	C:0.050 D:0.832	C:0.046 D:0.847
1.0	C:0.086 D:0.000	C:0.086 D:0.294	C:0.087 D:0.557	C:0.079 D:0.692	C:0.087 D:0.771	C:0.089 D:0.800	C:0.088 D:0.831	C:0.089 D:0.860	C:0.092 D:0.872	C:0.088 D:0.897	C:0.088 D:0.907

由上面的图看出，在没有被拒绝的飞机的情况下，

- A. 尽可能大的着陆请求率为：1，
- B. 起飞请求率为：1；
- C. 若使得二者都尽可能大，则取着陆请求 0.5，起飞请求率 0.2；



D. 若使得二者都尽可能大，而且方差最小，则取着陆请求 0.3，起飞请求率 0.3

(3) 当队列长度为 7，进行 10000 次实验，起飞/降落请求概率从 0.1-1.0 进行改变，得到如下数据：（其中 B 代表起飞请求率，A 代表着陆请求率，C 代表被拒绝着陆的概率，D 代表被拒绝起飞的概率）

A\B	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0.003	C: 0 D: 0.012	C: 0 D: 0.033	C: 0 D: 0.067
0.1	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0.001	C: 0 D: 0.003	C: 0 D: 0.011	C: 0 D: 0.031	C: 0 D: 0.076	C: 0 D: 0.135
0.2	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0 D: 0	C: 0.000 D: 0.001	C: 0.000 D: 0.003	C: 0.000 D: 0.012	C: 0.000 D: 0.038	C: 0.000 D: 0.081	C: 0.000 D: 0.145	C: 0.000 D: 0.216
0.3	C: 0 D: 0	C: 0.000 D: 0.000	C: 0.000 D: 0.000	C: 0.000 D: 0.001	C: 0.000 D: 0.004	C: 0.000 D: 0.018	C: 0.000 D: 0.049	C: 0.000 D: 0.097	C: 0.000 D: 0.163	C: 0.000 D: 0.239	C: 0.000 D: 0.300
0.4	C: 0.000 D: 0.000	C: 0.000 D: 0.000	C: 0.000 D: 0.000	C: 0.000 D: 0.003	C: 0.000 D: 0.010	C: 0.000 D: 0.046	C: 0.000 D: 0.099	C: 0.000 D: 0.182	C: 0.000 D: 0.26	C: 0.000 D: 0.334	C: 0.000 D: 0.395
0.5	C: 0.000 D: 0.000	C: 0.000 D: 0.000	C: 0.000 D: 0.000	C: 0.000 D: 0.014	C: 0.000 D: 0.049	C: 0.000 D: 0.121	C: 0.000 D: 0.216	C: 0.000 D: 0.316	C: 0.000 D: 0.390	C: 0.000 D: 0.452	C: 0.000 D: 0.499
0.6	C: 0.000 D: 0.000	C: 0.001 D: 0.000	C: 0.001 D: 0.009	C: 0.000 D: 0.058	C: 0.000 D: 0.130	C: 0.000 D: 0.247	C: 0.000 D: 0.356	C: 0.000 D: 0.445	C: 0.000 D: 0.505	C: 0.000 D: 0.558	C: 0.000 D: 0.607
0.7	C: 0.003 D: 0.000	C: 0.004 D: 0.009	C: 0.006 D: 0.060	C: 0.007 D: 0.184	C: 0.005 D: 0.316	C: 0.004 D: 0.425	C: 0.003 D: 0.516	C: 0.002 D: 0.568	C: 0.002 D: 0.627	C: 0.002 D: 0.662	C: 0.002 D: 0.697
0.8	C: 0.011 D: 0.000	C: 0.011 D: 0.033	C: 0.013 D: 0.230	C: 0.010 D: 0.349	C: 0.009 D: 0.472	C: 0.011 D: 0.568	C: 0.014 D: 0.651	C: 0.012 D: 0.700	C: 0.012 D: 0.724	C: 0.012 D: 0.748	C: 0.012 D: 0.782
0.9	C: 0.028 D: 0.000	C: 0.028 D: 0.157	C: 0.028 D: 0.421	C: 0.030 D: 0.562	C: 0.035 D: 0.658	C: 0.033 D: 0.725	C: 0.031 D: 0.804	C: 0.030 D: 0.820	C: 0.035 D: 0.849	C: 0.031 D: 0.864	C: 0.037 D: 0.880
1.0	C: 0.076 D: 0.000	C: 0.075 D: 0.447	C: 0.078 D: 0.667	C: 0.076 D: 0.777	C: 0.077 D: 0.855	C: 0.082 D: 0.881	C: 0.072 D: 0.905	C: 0.070 D: 0.908	C: 0.074 D: 0.921	C: 0.068 D: 0.925	C: 0.080 D: 0.940

由上面的图看出，在没有被拒绝的飞机的情况下，

A. 尽可能大的着陆请求率为：1，

B. 起飞请求率为：1；

C. 若使得二者都尽可能大，而且方差最小，则取着陆请求率 0.4，起飞请求率 0.4

综合三张表，我们可以看出，随着队列最大长度的增大，最大的着陆请求率和起飞请求率逐渐增加（在同时考虑到均值和方差的情况下）；

P2、P3：编译运行，并输入测试样例打印每个时间点等待队列情况，跑道调度情况，以及程序结束时跑道总的调度情况

```
This programe simulates an airport with two runways, one is for landing, the other is for taking off.
Up to what number of planes can be waiting to land or take off at any time?5
How many units of time will the simulation run?20
Expected number of arrivals per unit time?1
Expected number of departures per unit time?1
Plane number 0 ready to land.
Plane number 1 ready to take off.
At time 0: Plane number 0 landed in No. 1 runway after 0 time units in the landing queue.
At time 0: Plane number 1 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 2 ready to land.
Plane number 3 ready to land.
Plane number 4 ready to land.
Plane number 5 ready to land.
Plane number 6 ready to take off.
At time 1: Plane number 2 landed in No. 1 runway after 0 time units in the landing queue.
At time 1: Plane number 6 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 7 ready to land.
Plane number 8 ready to land.
Plane number 9 ready to take off.
Plane number 10 ready to take off.
At time 4: Plane number 8 landed in No. 1 runway after 0 time units in the landing queue.
At time 4: Plane number 9 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 11 ready to land.
Plane number 12 ready to take off.
At time 5: Plane number 11 landed in No. 1 runway after 0 time units in the landing queue.
At time 5: Plane number 10 takes off in No. 2 runway after 1 time unit in the takeoff queue.
Plane number 13 ready to land.
Plane number 14 ready to take off.
At time 6: Plane number 13 landed in No. 1 runway after 0 time units in the landing queue.
At time 6: Plane number 12 takes off in No. 2 runway after 1 time unit in the takeoff queue.
Plane number 15 ready to land.
At time 7: Plane number 15 landed in No. 1 runway after 0 time units in the landing queue.
At time 7: Plane number 14 takes off in No. 2 runway after 1 time unit in the takeoff queue.
Plane number 16 ready to land.
Plane number 17 ready to land.
Plane number 18 ready to take off.
At time 8: Plane number 16 landed in No. 1 runway after 0 time units in the landing queue.
At time 8: Plane number 18 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 19 ready to take off.
Plane number 20 ready to take off.
At time 9: Plane number 17 landed in No. 1 runway after 1 time unit in the landing queue.
At time 9: Plane number 19 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 21 ready to land.
Plane number 22 ready to land.
At time 10: Plane number 21 landed in No. 1 runway after 0 time units in the landing queue.
At time 10: Plane number 20 takes off in No. 2 runway after 1 time unit in the takeoff queue.
Plane number 23 ready to land.
Plane number 24 ready to take off.
At time 11: Plane number 22 landed in No. 1 runway after 1 time unit in the landing queue.
At time 11: Plane number 24 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 25 ready to land.
Plane number 26 ready to take off.
Plane number 27 ready to take off.
Plane number 28 ready to take off.
At time 12: Plane number 23 landed in No. 1 runway after 1 time unit in the landing queue.
At time 12: Plane number 26 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 13: Plane number 25 landed in No. 1 runway after 1 time unit in the landing queue.
At time 13: Plane number 27 takes off in No. 2 runway after 1 time unit in the takeoff queue.
Plane number 29 ready to land.
Plane number 30 ready to land.
At time 14: Plane number 29 landed in No. 1 runway after 0 time units in the landing queue.
At time 14: Plane number 28 takes off in No. 2 runway after 2 time units in the takeoff queue.
Plane number 31 ready to land.
Plane number 32 ready to land.
Plane number 33 ready to land.
Plane number 34 ready to take off.
At time 16: Plane number 32 landed in No. 1 runway after 1 time unit in the landing queue.
At time 16: Plane number 34 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 35 ready to land.
Plane number 36 ready to take off.
At time 17: Plane number 33 landed in No. 1 runway after 1 time unit in the landing queue.
At time 17: Plane number 36 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 37 ready to land.
Plane number 38 ready to take off.
Plane number 39 ready to take off.
At time 18: Plane number 35 landed in No. 1 runway after 1 time unit in the landing queue.
At time 18: Plane number 38 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 40 ready to land.
Plane number 41 ready to land.
Plane number 42 ready to land.
Plane number 43 ready to take off.
At time 19: Plane number 37 landed in No. 1 runway after 1 time unit in the landing queue.
At time 19: Plane number 39 takes off in No. 2 runway after 1 time unit in the takeoff queue.

=====
This is the runway for landing.
Simulation has concluded after 20 time units.
Total number of planes processed during the Simulation: 26.
Total number of planes asking to land: 26
Total number of planes refused to land: 0
Total number of planes accepted to land: 26
Total number of planes landed successfully: 23
Total time all the planes wait( for landing ): 13
Average time that every planes wait( for landing ): 0.565217
Total time the Runway idle: 0
Percentage of time Runway idle: 0
Average rate of plane wanting to land: 1.3

=====
This is the runway for taking off.
Simulation has concluded after 20 time units.
Total number of planes processed during the Simulation: 18.
Total number of planes asking to takeoff: 18
Total number of planes refused to takeoff: 0
Total number of planes accepted to takeoff: 18
Total number of planes takeoff successfully: 17
Total time all the planes wait( for taking off ): 8
Average time that every planes wait( for taking off ): 0.470588
Total time the Runway idle: 0
Percentage of time Runway idle: 0
Average rate of plane wanting to takeoff: 0.9
```

```

This programe simulates an airport with two runways, one is for landing, the other is for taking off.
Up to what number of planes can be waiting to landor take off at any time??
How many units of time will the simulation run?20
Expected number of arrivals per unit time?1
Expected number of departures per unit time?1
Plane number 0 ready to land.
At time 0: the No. 2 runway is idle
Plane number 1 ready to land.
Plane number 2 ready to land.
Plane number 3 ready to land.
Plane number 4 ready to land.
Plane number 5 ready to land.
Plane number 6 ready to take off.
At time 2: Plane number 3 landed in No. 1 runway after 0 time units in the landing queue.
At time 2: Plane number 6 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 7 ready to take off.
At time 3: Plane number 4 landed in No. 1 runway after 1 time unit in the landing queue.
At time 3: Plane number 7 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 4: the No. 2 runway is idle
Plane number 8 ready to take off.
At time 5: Plane number 8 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 5: the No. 1 runway is idle
Plane number 9 ready to take off.
Plane number 10 ready to take off.
At time 6: Plane number 9 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 6: Plane number 10 takes off in No. 1 runway after 0 time units in the takeoff queue.
Plane number 11 ready to land.
Plane number 12 ready to land.
Plane number 13 ready to take off.
At time 7: Plane number 11 landed in No. 1 runway after 0 time units in the landing queue.
At time 7: Plane number 13 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 8: the No. 2 runway is idle
Plane number 14 ready to take off.
At time 9: Plane number 14 takes off in No. 2 runway after 0 time units in the takeoff queue.
At time 9: the No. 1 runway is idle
At time 10: the No. 2 runway is idle
At time 10: the No. 1 runway is idle
At time 11: the No. 2 runway is idle
At time 11: the No. 1 runway is idle
Plane number 15 ready to land.
Plane number 16 ready to take off.
Plane number 17 ready to take off.
Plane number 18 ready to take off.
At time 12: Plane number 15 landed in No. 1 runway after 0 time units in the landing queue.
At time 12: Plane number 16 takes off in No. 2 runway after 0 time units in the takeoff queue.
Plane number 19 ready to take off.
At time 13: Plane number 17 takes off in No. 2 runway after 1 time unit in the takeoff queue.
At time 13: Plane number 18 takes off in No. 1 runway after 1 time unit in the takeoff queue.
Plane number 20 ready to take off.
Plane number 21 ready to take off.
Plane number 22 ready to take off.
Plane number 23 ready to take off.
At time 14: Plane number 19 takes off in No. 2 runway after 1 time unit in the takeoff queue.
At time 14: Plane number 20 takes off in No. 1 runway after 0 time units in the takeoff queue.
Plane number 24 ready to land.
Plane number 25 ready to take off.
Plane number 26 ready to take off.
At time 15: Plane number 24 landed in No. 1 runway after 0 time units in the landing queue.
At time 15: Plane number 21 takes off in No. 2 runway after 1 time unit in the takeoff queue.
At time 16: Plane number 22 takes off in No. 2 runway after 2 time units in the takeoff queue.
At time 16: Plane number 23 takes off in No. 1 runway after 2 time units in the takeoff queue.
Plane number 27 ready to land.
Plane number 28 ready to land.
Plane number 29 ready to take off.
Plane number 30 ready to take off.
At time 17: Plane number 27 landed in No. 1 runway after 0 time units in the landing queue.
At time 17: Plane number 25 takes off in No. 2 runway after 2 time units in the takeoff queue.
Plane number 31 ready to take off.
At time 18: Plane number 28 landed in No. 1 runway after 1 time unit in the landing queue.
At time 18: Plane number 26 takes off in No. 2 runway after 3 time units in the takeoff queue.
Plane number 32 ready to land.
Plane number 33 ready to land.
Plane number 34 ready to take off.
At time 19: Plane number 32 landed in No. 1 runway after 0 time units in the landing queue.
At time 19: Plane number 29 takes off in No. 2 runway after 2 time units in the takeoff queue.

=====

This is the runway for landing.
Simulation has concluded after 20 time units.
Total number of planes processed during the Simulation: 14.
Total number of planes asking to land: 14
Total number of planes refused to land: 0
Total number of planes accepted to land: 14
Total number of planes landed successfully: 13
Total time all the planes wait( for landing ): 5
Average time that every planes wait( for landing ): 0.384615
Total time the Runway idle: 3
Percentage of time Runway idle: 0.15
Average rate of plane wanting to land: 0.7

=====

This is the runway for taking off.
Simulation has concluded after 20 time units.
Total number of planes processed during the Simulation: 21.
Total number of planes asking to takeoff: 21
Total number of planes refused to takeoff: 0
Total number of planes accepted to takeoff: 21
Total number of planes takeoff successfully: 18
Total time all the planes wait( for taking off ): 15
Average time that every planes wait( for taking off ): 0.833333
Total time the Runway idle: 6
Percentage of time Runway idle: 0.3
Average rate of plane wanting to takeoff: 1.05

```

通过多次测试（更多的测试样例难以在有限的篇幅中给出，所以仅仅给出以上两个作为示例），分析多次测试数据（类似于 P1 题的方法，着陆请求率和起飞请求率都从 0.1 逐渐增加到 1.0，然后经过画表、统计分析）可知：


- (1) 当“**着陆请求率+起飞请求率**” $\leq 1$ （单条跑道能够比较好地处理的最大请求量）时，单条跑道和双跑道所能够处理的请求数量**相差不大**；
- (2) 但是随着“**着陆请求率+起飞请求率**” $>1$ 并且**逐渐增大**，双跑道所能够处理的请求数量渐渐超过单条跑道所能够处理的请求数目；
- (3) 当“**着陆请求率+起飞请求率**”快达到 2（双跑道能够比较好地处理的最大请求量）时，双跑道所能够处理的请求数量**渐渐达到**单条跑道所能够处理请求数目的**2 倍**；
- (4) 是随着“**着陆请求率+起飞请求率**” $>2$ 并且**逐渐增大**，双跑道所能够处理的请求数量**达到**单条跑道所能够处理请求数目的**2 倍**，并且基本上**维持两倍**；

P4:

1. 编译运行，并输入测试样例：

```
C:\Users\14735\Desktop\exercise for c++\project2--big pro>a
This programe simulates an airport with only one runway.
Up to what number of planes can be waiting to land or take off at any time?      5
How many units of time will the simulation run?                                10
Expected number of arrivals per unit time?                                     0.5
Expected number of departures per unit time?                                    0.5
```

2. 输出：打印每个时间点等待队列情况，跑道调度情况，以及程序结束时跑道总的调度情况：



```
选择命令提示符
Expected number of arrivals per unit time? 0.5
Expected number of departures per unit time? 0.5

0: Plane NO.0 ready to take off.
0: the landing runway is idle
0: Plane NO.0 takes off after 0 time units int the takeoffs queue.
0: the third runway is idle
Plane NO.1 ready to land.
Plane NO.2 ready to land.
Plane NO.3 ready to take off.
Plane NO.4 ready to take off.
Plane NO.5 ready to take off.
Plane NO.6 ready to take off.
1: Plane NO.1 landed after 0 time units int the landings queue.
1: Plane NO.3 takes off after 0 time units int the takeoffs queue.
1: Plane NO.2 landed after 0 time units int the landings queue.
2: the landing runway is idle
2: Plane NO.4 takes off after 1 time unit int the takeoffs queue.
2: Plane NO.5 takes off after 1 time unit int the takeoffs queue.
3: the landing runway is idle
3: Plane NO.6 takes off after 2 time units int the takeoffs queue.
4: the third runway is idle
Plane NO.7 ready to take off.
4: the landing runway is idle
4: Plane NO.7 takes off after 0 time units int the takeoffs queue.
4: the third runway is idle

Simulation has concluded after 5 time units.
Total number of planes processed during the Simulation: 8.
Total number of planes asking to land: 2
Total number of planes asking to takeoff: 6
Total number of planes refused to land: 0
Total number of planes refused to takeoff: 0
Total number of planes accepted to land: 2
Total number of planes accepted to takeoff: 6
Total number of planes landed successfully: 2
Total number of planes takeoff successfully: 6
Total time all the planes wait( for landing ): 0
Total time all the planes wait( for taking off ): 4
Average time that every planes wait( for landing ): 0
Average time that every planes wait( for taking off ): 0.666667
Total time the Runway idle: 7
Percentage of time Runway idle: 1.4
Average rate of plane wanting to land: 0.4
Average rate of plane wanting to takeoff: 1.2

C:\Users\14735\Desktop\exercise for c++\project2--big pro\Airport\Airport>
```

P5:

编译运行，并输入测试样例，打印每个时间点等待队列情况，跑道调度情况，以及程序结束时跑道总的调度情况：

```
at time 818: Plane number 761 landed in an emergency
at time 819: Plane number 761 takes off after 6 time units int the takeoffs queue.
'lane number 768 ready to land in an EMERGENCY
'lane number 769 ready to take off.
at time 820: Plane number 768 landed in an emergency
'lane number 770 ready to land in an EMERGENCY
'lane number 771 ready to land in an EMERGENCY
'lane number 772 ready to take off.
at time 821: Plane number 770 landed in an emergency
at time 822: Plane number 771 landed in an emergency
at time 823: Plane number 763 takes off after 9 time units int the takeoffs queue.
'lane number 773 ready to land in an EMERGENCY
at time 824: Plane number 773 landed in an emergency
at time 825: Plane number 764 takes off after 10 time units int the takeoffs queue.
'lane number 774 ready to land in an EMERGENCY
'lane number 775 ready to take off.
at time 826: Plane number 774 landed in an emergency
'lane number 776 ready to land.
'lane number 777 ready to take off.
at time 827: Plane number 776 landed after 0 time units int the landings queue.
at time 828: Plane number 769 takes off after 8 time units int the takeoffs queue.
'lane number 778 ready to land.
at time 829: Plane number 778 landed after 0 time units int the landings queue.
at time 830: Plane number 772 takes off after 9 time units int the takeoffs queue.
'lane number 779 ready to land.
```

从上面这张图可以看出，存在飞机发生紧急着陆请求（图中划线部分）

```
'lane number 737 told to try taking off again later
: time 776: Plane number 736 landed in an emergency
'lane number 738 ready to take off.
'lane number 738 told to try taking off again later
: time775: Plane number 731 crashed
percentage of time Runway idle: 0.101673
'lane number 739 ready to land in an EMERGENCY
: time 778: Plane number 739 landed in an emergency
: time775: Plane number 732 crashed
percentage of time Runway idle: 0.101412
: time 780: Plane number 723 takes off after 12 time units int the takeoffs queue.
'lane number 740 ready to land in an EMERGENCY
'lane number 741 ready to take off.
```

从这张图可以看到，飞机场的空闲率 0.101，这反映了这个飞机场的繁忙程度，仅有 10%左右  
空闲时段

```
at time 999: Plane number 999 landed after 0 time units int the landings queue.
Simulation has concluded after 1000 time units.
total number of planes processed during the Simulation: 857.
total number of planes asking to land: 403
total number of planes asking to takeoff: 454
total number of planes refused to land: 0
total number of planes refused to takeoff: 46
total number of planes accepted to land: 403
total number of planes accepted to takeoff: 408
total number of planes landed successfully: 475
total number of planes takeoff successfully: 408
total number of planes crashed: 25
total number of emergencies: 97
total time all the planes wait( for landing ): 153
total time all the planes wait( for taking off ): 1816
average time that every planes wait( for landing ): 0.322105
average time that every planes wait( for taking off ): 4.45098
total time the Runway idle: 92
percentage of time Runway idle: 0.092
不能五笔输入法 全 :ne wanting to land: 0.403
average rate of plane wanting to takeoff: 0.454
```



这张图则可以看出来，最终有 97 架飞机因为紧急情况优先降落，有 25 架飞机坠毁（其实个人认为，应该在第一架飞机坠毁之前就应该停止程序）

P6:

### 1. 编译运行，并输入测试样例:

```
C:\Users\14735\Desktop\exercise for c++\project2--big pro>a
This programe simulates an airport with only one runway.
Up to what number of planes can be waiting to land or take off at any time?      5
How many units of time will the simulation run?                                10
Expected number of arrivals per unit time?                                      0.5
Expected number of departures per unit time?                                    0.5
```

### 2. 输出: 打印每个时间点等待队列情况, 跑道调度情况, 以及程序结束时跑道的调度情况:

Which mode do you want to choose('A' for auto debug, 'C' for customised manipulation)

A

```
0: the runway is idle
1: the runway is idle
2: the runway is idle
3: the runway is idle
   Plane NO.0 ready to land.
   Plane NO.1 ready to land.
   Plane NO.2 ready to land.
4: Plane NO.0 landed after 0 time units int the landings queue.
5: Plane NO.1 landed after 1 time unit int the landings queue.
   Plane NO.3 ready to land.
6: Plane NO.2 landed after 2 time units int the landings queue.
   Plane NO.4 ready to take off.
7: Plane NO.3 landed after 1 time unit int the landings queue.
   Plane NO.5 ready to land.
8: Plane NO.5 landed after 0 time units int the landings queue.
9: Plane NO.4 takes off after 2 time units int the takeoffs queue.
```

```
Simulation has concluded after 10 time units.
Total number of planes processed during the Simulation: 6.
Total number of planes asking to land: 5
Total number of planes asking to takeoff: 1
Total number of planes refused to land: 0
Total number of planes refused to takeoff: 0
Total number of planes accepted to land: 5
Total number of planes accepted to takeoff: 1
Total number of planes landed successfully: 5
Total number of planes takeoff successfully: 1
Total time all the planes wait( for landing ): 4
Total time all the planes wait( for taking off ): 2
Average time that every planes wait( for landing ): 0.8
Average time that every planes wait( for taking off ): 2
Total time the Runway idle: 4
Percentage of time Runway idle: 0.4
Average rate of plane wanting to land: 0.5
Average rate of plane wanting to takeoff: 0.1
```

此处由于我选择了 A, 所以程序选择了 auto\_debug, 也就是选择执行原来的操作;

下面将展示执行单步调试 (由用户输入每一步的起飞请求率、降落请求率):

### 1. 运行程序，并输入测试样例:

```
C:\Users\14735\Desktop\exercise for c++\project2--big pro>a
This programe simulates an airport with only one runway.
Up to what number of planes can be waiting to land or take off at any time?      5
How many units of time will the simulation run?                                10
Expected number of arrivals per unit time?                                      0.5
Expected number of departures per unit time?                                    0.5
```

2. 输出：打印每个时间点等待队列情况，跑道调度情况，以及程序结束时跑道总的调度情况：

```
Which mode do you want to choose('A' for auto debug, 'C' for customised manipulation)
C
Please enter the number of arriving planes: 0.5
Please enter the number of departure planes: 0: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
1: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
2: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
3: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
4: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
5: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
6: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
7: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
8: the runway is idle
Please enter the number of arriving planes: Please enter the number of departure planes:
9: the runway is idle

Simulation has concluded after 10 time units.
Total number of planes processed during the Simulation: 0.
Total number of planes asking to land: 0
Total number of planes asking to takeoff: 0
Total number of planes refused to land: 0
Total number of planes refused to takeoff: 0
Total number of planes accepted to land: 0
Total number of planes accepted to takeoff: 0
Total number of planes landed successfully: 0
Total number of planes takeoff successfully: 0
Total time all the planes wait( for landing ): 0
Total time all the planes wait( for taking off ): 0
Average time that every planes wait( for landing ): -1.#IND
Average time that every planes wait( for taking off ): -1.#IND
Total time the Runway idle: 10
Percentage of time Runway idle: 1
Average rate of plane wanting to land: 0
Average rate of plane wanting to takeoff: 0
```

这上面的程序选择 C，所以程序允许用户自行输入起飞请求率、降落请求率；（选项情况以及输入要求在图中画出）

## 【分工、贡献%、自我评分】

### 1. 分工：

P1 版程序编写及数据处理统计、报告撰写：欧穗新；

P2、P3 版程序编写及数据处理统计：欧阳梓轩；

P4、P6 版程序编写及数据处理统计：彭靖寒；

P5 版程序编写及数据处理统计：潘鉴；

### 2. 贡献%：

欧穗新： 25%

欧阳梓轩： 25%

潘鉴： 25%

彭靖寒： 25%

### 3. 自我评分：

欧穗新 90

欧阳梓轩 85

潘鉴 90

彭靖寒 91

## 【项目总结】

欧穗新：

首先说一下收获：

- (1) 这次的项目，很大一部分是书上给的代码（基础代码都是书上的），然后参照教材的内容，看到了比较好的面向对象的编程方式；
- (2) 这次由于题目比较多（总共有 6 个题目），比较难一个人完成，而在分工下可以比较好的完成，相比第一次的 pro 而言明显感受到分工合作的好处，也进一步体会到分工合作的方式；

然后说一下这次的不足：

- (1) 第二次大 pro，感觉自己做的并不是很好，因为这三周里面刚刚好国庆节是第二周，然而国庆节大家没能达成一致去做这个 pro，所以最后只有一周时间来做这个 pro，时间上感觉很紧，我做的是数据分析部分，这个刚好需要比较多的时间投入，所以受时间限制，数据分析完成的并不完美；

总之，下次应该吸取这一次的经验教训，尽量多提前一些做 pro；同时更多的和小伙伴门进行沟通交流，更好的合作；



欧阳梓轩：

通过这一次项目的学习，锻炼了对已有代码的阅读和理解能力，进一步熟悉了许多文件的数据处理以及各函数之间的协同处理，比如在着陆队列已满时对起飞队列的处理，获得了难得的经验。

潘鉴：

在这次项目中，我们通过模拟飞机场起飞和降落的情况，对队列的理解更深了。这次过程中我们小组共同讨论，然后分工。这次项目让我知道现实生活中有很多要考虑的因素，我们完成基本的实现之外还可能要考虑更多，逐步完善程序。另外，我觉得我的队友都很优秀。

彭靖寒：

通过这次 Project，通过对队列的实际应用，我加深了对队列的理解，了解了队列的机制和原理，并且深化了对队列的应用。这次的飞机模拟模型当然也可以用在实际生活的其他地方，如排队的其他事例，都可以应用这次的飞机模拟模型或其拓展。这也体现出队列在程序和实际生活中应用广泛，需要注意。同时，这次 Project 也让我体会到讨论程序设计的乐趣，加深对程序的理解。

## 【程序清单】

(1) main 函数：

P1 题的 main 函数：

```
#include <iostream>
#include "Plane.hpp"
#include "Runway.hpp"
#include "Random.hpp"
```

```
using namespace std;
```

```

void run_idle (int current_time);
void intro_and_ini (int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate);

```

```

int main () {
    int end_time = 0;
    int queue_limit = 0;
    int flight_number = 0;
    double arrival_rate = 0, departure_rate = 0;
    intro_and_ini(end_time, queue_limit, arrival_rate, departure_rate);

```

Random variable;

Runway small\_airport(queue\_limit);

```

for (int current_time = 0; current_time < end_time; ++current_time)
{
    int number_arrivals = variable.poisson(arrival_rate);
    for (int i = 0; i < number_arrivals; ++i)
    {
        Plane current_plane(flight_number++, current_time, arriving);
        if (small_airport.can_land(current_plane) != success) {
            current_plane.refuse();
        }
    }
    int number_departures = variable.poisson(departure_rate);
    for (int i = 0; i < number_departures; ++i)
    {
        Plane current_plane(flight_number++, current_time, departing);
        if (small_airport.can_depart(current_plane) != success) {
            current_plane.refuse();
        }
    }
    Plane moving_plane;
    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);
    }
}

```

```

    small_airport.shut_down(end_time);
    return 0;
}

void intro_and_ini (int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate) {
    // (这里可以优化UI)
    //introduction:
    cout << "This programe simulates an airport with only one runway." << endl;
    //initialization:
    cout << "Up to what number of planes can be waiting to land or take off at any time?    "
    << flush;
    cin >> queue_limit;
    cout << "How many units of time will the simulation run?
    " << flush;
    cin >> end_time;
    bool acceptable = false;
    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 not be negative! Please input again:
    " << endl;
        } else {
            acceptable = true;
        }
        if (arrival_rate + departure_rate > 1) {
            cerr << "Safe warning: the airport must be very saturated!
    " << endl;
        }
        // if (test_input())
        // //在这里对输入进行测试, 如果输入正确, 返回1, 否则返回0;
        // acceptable = true;
        //下面仅仅是把acceptable记为true;
        //acceptable = true;
    } while (!acceptable);
}

void run_idle (int current_time) {
    cout << "At time " << current_time << ": the runway is idle" << endl;

```

```
}
```

P2、3 题的 main 函数：

```
#include "Plane.hpp"
#include "Runway.hpp"
#include "Random.hpp"
```

```
using namespace std;
```

```
void run_idle( int current_time, int idx );           //idx为跑道编号
void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate );
```

```
int main()
```

```
{
    int end_time = 0;
    int queue_limit = 0;
    int flight_number = 0;
    double arrival_rate = 0, departure_rate = 0;
    intro_and_ini( end_time, queue_limit, arrival_rate, departure_rate );
    Random variable;
    // Runway small_airport( queue_limit );
    Runway landing_runway( queue_limit, landing ), takingoff_runway( queue_limit,
takingoff );
```

```
    for ( int current_time = 0; current_time < end_time; ++current_time ) {
```

```
        int number_arrivals = variable.poisson( arrival_rate );
        bool saturatedLanding = false;           //标记着陆队列是否过满
```

```
        for ( int i = 0; i < number_arrivals; ++i ) {
```

```
            Plane current_plane( flight_number++, current_time, arriving );
            if ( landing_runway.can_land( current_plane ) != success ) {
                saturatedLanding = true;
```

```
            //若起飞跑道的着陆队列也已满，则拒绝
```

```
            if ( takingoff_runway.can_land( current_plane ) != success )
                current_plane.refuse();
```

```
        }
```

```
    }
```

```

int number_departures = variable.poisson( departure_rate );

for ( int i = 0; i < number_departures; ++i ) {
    Plane current_plane( flight_number++, current_time, departing );
    if ( takingoff_runway.can_depart( current_plane ) != success ) {
        current_plane.refuse();
    }
}

Plane moving_plane;

if ( saturatedLanding ) {

    cout << "==== Stop the takeoffs and both runways are used to"
         << " clear the backlog of landing planes. ====" << endl;

    landing_runway.activity( current_time, moving_plane );
    moving_plane.land( current_time );

    //用起飞的跑道着陆
    takingoff_runway.activity( current_time, moving_plane );
    moving_plane.land( current_time, 2 );           //用于起飞的跑道编号为 2

} else if ( landing_runway.empty() || takingoff_runway.empty() ) {

    if ( landing_runway.empty() ) {

        for ( int i = 2; i >= 1; --i ) {           //可以将着陆跑道用于起飞
            switch ( takingoff_runway.activity( current_time, moving_plane ) ) {
                case takeoff:
                    moving_plane.fly( current_time, i );
                    break;
                case idle:
                    run_idle( current_time, i );
            }
        }

    } else {

        for ( int i = 1; i <= 2; ++i ) {           //可以将起飞跑道用于着陆
            switch ( landing_runway.activity( current_time, moving_plane ) ) {
                case takeoff:
                    moving_plane.fly( current_time, i );
                    break;
            }
        }

    }

}

```

```

        case idle:
            run_idle( current_time, i );
        }
    }

} else {

    switch ( landing_runway.activity( current_time, moving_plane ) ) {
        case land:
            moving_plane.land( current_time );
            break;
        case idle:
            run_idle( current_time, 1 );
    }

    switch ( takingoff_runway.activity( current_time, moving_plane ) ) {
        case takeoff:
            moving_plane.fly( current_time );
            break;
        case idle:
            run_idle( current_time, 2 );
    }

    }

}

cout << endl << "===== " <<
endl << endl;
landing_runway.shut_down( end_time );
cout << endl << "===== " <<
endl << endl;
takingoff_runway.shut_down( end_time );

getchar();
getchar();
return 0;
}

void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate ) {

```

*//introduction:*

```

cout << "This programe simulates an airport with two runways, one is for landing, "
      << "the other is for taking off." << endl;

//initialization:
bool acceptable = false;
do {

    cout << "Up to what number of planes can be waiting to land"
          << "or take off at any time?" << flush;
    cin >> queue_limit;
    cout << "How many units of time will the simulation run?" << flush;
    cin >> end_time;
    cout << "Expected number of arrivals per unit time?" << flush;
    cin >> arrival_rate;
    cout << "Expected number of departures per unit time?" << flush;
    cin >> departure_rate;

    acceptable = true;

} while ( !acceptable );

}

void run_idle( int current_time, int idx ) {
    cout << "At time " << current_time << ": the No. " << idx << " runway is idle." << endl;
}

```

**P4 题的 main 函数：**

```

#include "Plane.hpp"
#include "Runway.hpp"
#include "Random.hpp"

using namespace std;

void landing_run_idle( int current_time );
void takingoff_run_idle( int current_time );
void third_run_idle( int current_time );
void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate );

int main() {
    int end_time = 0; //模拟运行的时间

```

```

int queue_limit = 0; //Runway 中队列大小限制
int flight_number = 0; //每一个飞机的序号, 0 1 2 3...
double arrival_rate = 0, departure_rate = 0; //到达率 与 离开率
intro_and_ini( end_time, queue_limit, arrival_rate, departure_rate ); //初始化
Random variable; //伪随机对象, 功能是得到特定时间到达或离开机场的飞机的随机数量

Runway small_airport( queue_limit );

for ( int current_time = 0; current_time < end_time; ++current_time ) { //循环, 每个单位时间内的操作

    int number_arrivals = variable.poisson( arrival_rate ); //当前到达机场的新飞机数量
    for ( int i = 0; i < number_arrivals; ++i ) { //创建新到达的飞机对象
        Plane current_plane( flight_number++, current_time, arriving );
        if ( small_airport.can_land( current_plane ) != success ) { //判断飞机能否加入待降落队列, 成功则加入队列
            current_plane.refuse(); //若不能加入, 则拒绝该飞机加入队列
        }
    }

    int number_departures = variable.poisson( departure_rate ); //当前请求离开的飞机的数量
    for ( int i = 0; i < number_departures; ++i ) { //创建要离开的飞机队列
        Plane current_plane( flight_number++, current_time, departing );
        if ( small_airport.can_depart( current_plane ) != success ) { //判断飞机能否加入待起飞队列, 成功则加入队列
            current_plane.refuse(); //若不能加入, 则拒绝该飞机加入队列
        }
    }

    Plane moving_plane;
    switch ( small_airport.landing_activity( current_time, moving_plane ) ) { //降落跑道的处理

        case land:
            moving_plane.land( current_time );
            break;
        case takeoff:
            break;
        case idle:
            landing_run_idle( current_time );
    }

    switch ( small_airport.takingoff_activity( current_time, moving_plane ) ) { //起飞跑道的处理

        case land:
            break;
        case takeoff:

```



```

        moving_plane.fly( current_time );
        break;
    case idle:
        takingoff_run_idle( current_time );
    }

    switch ( small_airport.activity( current_time, moving_plane ) ) { //第三条跑道处理
        case land:
            moving_plane.land( current_time );
            break;
        case takeoff:
            moving_plane.fly( current_time );
            break;
        case idle:
            third_run_idle( current_time );
    }
}
small_airport.shut_down( end_time );
return 0;
}

```

```

void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate ) {

```

```

    // (这里可以优化UI)

```

```

    //introduction:

```

```

    cout << "This programe simulates an airport with only one runway." << endl;

```

```

    //initialization:

```

```

    bool acceptable = false;

```

```

    do {

```

```

        cout << "Up to what number of planes can be waiting to land"

```

```

        << " or take off at any time? " << flush;

```

```

        cin >> queue_limit;

```

```

        cout << "How many units of time will the simulation run? " << flush;

```

```

        cin >> end_time;

```

```

        cout << "Expected number of arrivals per unit time? " << flush;

```

```

        cin >> arrival_rate;

```

```

        cout << "Expected number of departures per unit time? " << flush;

```

```

        cin >> departure_rate;

```

```

        cout << endl;

```

```

        // if (test_input())

```

```

        ///在这里对输入进行测试, 如果输入正确, 返回1, 否则返回0;

```

```

        // acceptable = true;

```

```

        //下面仅仅是把acceptable记为true;

```

```

        acceptable = true;
    }
}

```

```

    } while ( !acceptable );
}

void landing_run_idle( int current_time ) {
    cout << "\t" << current_time << ":  the landing runway is idle" << endl;
}

void takingoff_run_idle( int current_time ) {
    cout << "\t" << current_time << ":  the taking-off runway is idle" << endl;
}

void third_run_idle( int current_time ) {
    cout << "\t" << current_time << ":  the third runway is idle" << endl;
}

```

**P5 题的 main 函数：**

```

#include "Plane.hpp"
#include "Runway.hpp"
#include "Random.hpp"

using namespace std;

void run_idle( int current_time );
void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate );

int main() {
    int end_time = 0;
    int queue_limit = 0;
    int flight_number = 0;
    double arrival_rate = 0, departure_rate = 0;
    intro_and_ini( end_time, queue_limit, arrival_rate, departure_rate );
    Random variable;
    Runway small_airport( queue_limit );
    bool emergency_situation = false;
    for ( int current_time = 0; current_time < end_time; ++current_time )
    {
        int number_arrivals = variable.poisson( arrival_rate );
        Plane moving_plane;
        for ( int i = 0; i < number_arrivals; ++i )
        {
            Plane current_plane( flight_number++, current_time, arriving );
            if(!emergency_situation && current_plane.get_status() == emergency){

```

```

        moving_plane = current_plane;
        emergency_situation = true;
    }
    else if ( small_airport.can_land( current_plane ) != success ) {
        current_plane.refuse();
    }
}
int number_departures = variable.poisson( departure_rate );
for ( int i = 0; i < number_departures; ++i )
{
    Plane current_plane( flight_number++, current_time, departing );
    if ( small_airport.can_depart( current_plane ) != success ) {
        current_plane.refuse();
    }
}
switch ( small_airport.activity( current_time, moving_plane ) ) {
case land:
    moving_plane.land( current_time );
    if(moving_plane.crashed(current_time)){
        cout << "Percentage of time Runway idle: " <<
small_airport.busy(current_time) << endl;
    }
    break;
case takeoff:
    moving_plane.fly( current_time );
    break;
case emergency_land:
    moving_plane.land( current_time);
    emergency_situation = false;
    break;
case idle:
    run_idle( current_time );
}
}
small_airport.shut_down( end_time );
return 0;
}

void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate ) {
// (这里可以优化UI)
//introduction:
    cout << "This programe simulates an airport with only one runway." << endl;
//initialization:

```

```

bool acceptable = false;
do {
    cout << "Up to what number of planes can be waiting to land"
        << "or take off at any time?" << flush;
    cin >> queue_limit;
    cout << "How many units of time will the simulation run?" << flush;
    cin >> end_time;
    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 (test_input())
    // //在这里对输入进行测试, 如果输入正确, 返回1, 否则返回0;
    // acceptable = true;
    // 下面仅仅是把acceptable记为true;
    acceptable = true;
} while ( !acceptable );
}

void run_idle( int current_time ) {
    cout << "At time " << current_time << ": the runway is idle" << endl;
}

```

**P6 题的 main 函数:**

```

#include "Plane.hpp"
#include "Runway.hpp"
#include "Random.hpp"

using namespace std;

void run_idle( int current_time );
void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate );
//
int arrival_stub(char &cmd_stub, double &arrival_rate, Random& variable);
int departure_stub(char &cmd_stub, double &departure_rate, Random& variable);
//
int main() {
    int end_time = 0; //模拟运行的时间
    int queue_limit = 0; //Runway中队列大小限制
    int flight_number = 0; //每一个飞机的序号, 0 1 2 3...
    //

```

```

char cmd_stub; //占位程序判断符
//
double arrival_rate = 0, departure_rate = 0; //到达率 与 离开率
intro_and_ini( end_time, queue_limit, arrival_rate, departure_rate ); //初始化
Random variable; //伪随机对象，功能是得到特定时间到达或离开机场的飞机的随机数
量
Runway small_airport( queue_limit );
//
cout << "Which mode do you want to choose('A' for auto debug, 'C' for customised
manipulation)";
cin >> cmd_stub; //选择飞机场模拟方式 (A为Debug, C为用户输入)
//
for ( int current_time = 0; current_time < end_time; ++current_time ) { //循环，每个单位
时间内的操作
    //
    int number_arrivals = arrival_stub(cmd_stub, arrival_rate, variable);
    //

    //int number_arrivals = variable.poisson( arrival_rate ); //当前到达机场的新飞机数
量
    for ( int i = 0; i < number_arrivals; ++i ) { //创建新到达的飞机对象
        Plane current_plane( flight_number++, current_time, arriving );
        if ( small_airport.can_land( current_plane ) != success ) { //判断飞机能否加入
待降落队列，成功则加入队列
            current_plane.refuse(); //若不能加入，则拒绝该飞机加入队列
        }
    }

    //
    int number_departures = departure_stub(cmd_stub, departure_rate, variable);
    //

    //int number_departures = variable.poisson( departure_rate ); //当前请求离开的飞
机的数量
    for ( int i = 0; i < number_departures; ++i ) { //创建要离开的飞机队列
        Plane current_plane( flight_number++, current_time, departing );
        if ( small_airport.can_depart( current_plane ) != success ) { //判断飞机能否加入
待起飞队列，成功则加入队列
            current_plane.refuse(); //若不能加入，则拒绝该飞机加入队列
        }
    }

    Plane moving_plane;
    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 );
    }
}
small_airport.shut_down( end_time );
return 0;
}

```

```

void intro_and_ini( int& end_time, int& queue_limit, double& arrival_rate, double&
departure_rate ) {

```

```

    // (这里可以优化UI)

```

```

    //introduction:

```

```

    cout << "This programe simulates an airport with only one runway." << endl;

```

```

    //initialization:

```

```

    bool acceptable = false;

```

```

    do {

```

```

        cout << "Up to what number of planes can be waiting to land"

```

```

        << " or take off at any time? " << flush;

```

```

        cin >> queue_limit;

```

```

        cout << "How many units of time will the simulation run? " << flush;

```

```

        cin >> end_time;

```

```

        cout << "Expected number of arrivals per unit time? " << flush;

```

```

        cin >> arrival_rate;

```

```

        cout << "Expected number of departures per unit time? " << flush;

```

```

        cin >> departure_rate;

```

```

        cout << endl;

```

```

        // if (test_input())

```

```

        // //在这里对输入进行测试, 如果输入正确, 返回1, 否则返回0;

```

```

        // acceptable = true;

```

```

        //下面仅仅是把acceptable记为true;

```

```

        acceptable = true;

```

```

    } while ( !acceptable );

```

```

}

```

```

void run_idle( int current_time ) {

```

```

    cout << "\t" << current_time << ":  the runway is idle" << endl;

```

```

}

```

```

//占位函数(到达)

```

```

int arrival_stub(char &cmd_stub, double &arrival_rate, Random& variable) {
    bool cmd_vaild = false;
    int inter = 0;
    while(!cmd_vaild) {
        switch(cmd_stub){
            case 'A':
                cmd_vaild = true;
                inter = variable.poisson(arrival_rate);
                return inter;
                break;
            case 'C':
                cmd_vaild = true;
                cout << "Please enter the number of arriving planes: ";
                cin >> inter;
                return inter;
                break;
            default:
                cout << "Wrong Command!" << endl;
                cout << "Please enter the command again: ";
                cin >> cmd_stub;
        }
    }

    return inter;
}

```

*//占位函数(离开)*

```

int departure_stub(char &cmd_stub, double &departure_rate, Random& variable) {
    bool cmd_vaild = false;
    int inter = 0;
    while(!cmd_vaild) {
        switch(cmd_stub){
            case 'A':
                cmd_vaild = true;
                inter = variable.poisson(departure_rate);
                return inter;
                break;
            case 'C':
                cmd_vaild = true;
                cout << "Please enter the number of departure planes: ";
                cin >> inter;
                return inter;
                break;
        }
    }

    return inter;
}

```

```

        default:
            cout << "Wrong Command!" << endl;
            cout << "Please enter the command again: ";
            cin >> cmd_stub;
        }
    }

    return inter;
}

```

## (2) random 类函数:

```

#include <limits.h>
#include "Random.hpp"
#include <cmath>
#include <time.h>
Random::Random() {
    seed = time(NULL)%INT_MAX;
    //seed = 1;
    multiplier = 2743;
    add_on = 5923;
}
int Random::reseed() {
    seed = seed*multiplier + add_on;
    return seed;
}
double Random::random_real() {
    double max = INT_MAX + 1.0;
    double temp = reseed();
    if (temp < 0)temp = temp + max;
    return temp/max;
}
int Random::poisson(double mean) {
    double limit = exp(-mean);
    double product = random_real();
    int count = 0;
    while(product > limit) {
        count++;
        product *=random_real();
    }
    return count;
}

```



### (3) plane 类函数:

```
#include <iostream>
#include "Plane.hpp"
using namespace std;

Plane::Plane() {
    flt_num = -1;
    clock_start = -0;
    state = null;
};

Plane::Plane(int flt, int time, Plane_status cur_state) {
    flt_num = flt;
    clock_start = time;
    state = cur_state;
    //以下部分可以优化, 属于UI部分;
    cout << "          Plane number " << flt << " ready to";
    if (state == arriving) {
        cout << " land." << endl;
    } else {
        cout << " take off." << endl;
    }
};

Plane& Plane::operator=(const Plane& other) {
    flt_num = other.flt_num;
    clock_start = other.clock_start;
    state = other.state;
    return (*this);
};

void Plane::refuse() const {
    //以下部分可以优化, 属于UI部分;
    cout << "          Plane number " << flt_num;
    if (state == arriving) {
        cout << " directed to another airport" << endl;
    } else {
        cout << " told to try taking off again later" << endl;
    }
};

void Plane::fly(int time) const {
```

```

    int wait = time - clock_start;
    cout << "At time " << time << ": Plane number " << flt_num << " takes off after "
         << wait << " time unit" << (wait==1?"":"s") << " in the takeoffs queue." << endl;
};

void Plane::land(int time) const {
    int wait = time - clock_start;
    cout << "At time " << time << ": Plane number " << flt_num << " landed after "
         << wait << " time unit" << (wait==1?"":"s") << " in the landings queue." << endl;
};

int Plane::started() const {
    return clock_start;
}; // 用于与runway类沟通达到机场的时间

```

#### (4) runway 类函数:

```

#include <iostream>
#include <iomanip>
#include "Runway.hpp"
#include "Plane.hpp"
using namespace std;

```

```

Runway::Runway(int limit) {
    queue_limit = limit;
    num_land_requests = 0;
    num_takeoff_requests = 0;
    num_land_finished = 0;
    num_takeoff_finished = 0;
    num_land_accepted = 0;
    num_takeoff_accepted = 0;
    num_land_refused = 0;
    num_takeoff_refused = 0;
    land_wait_total = 0;
    takeoff_wait_total = 0;
    idle_time_total = 0;
};

```

```

Error_code Runway::can_land (const Plane &current) {
    Error_code result;
    if (landings.size() < queue_limit)
    {

```

```

        result = success;
        landings.push(current);
    } else {
        result = fail;
    }
    num_land_requests++;
    if (result == success) {
        num_land_accepted++;
    } else {
        num_land_refused++;
    }
    return result;
};

```

```

Error_code Runway::can_depart(const Plane &current) {
    Error_code result;
    if (takeoffs.size() < queue_limit)
    {
        result = success;
        takeoffs.push(current);
    } else {
        result = fail;
    }
    num_takeoff_requests++;
    if (result == success) {
        num_takeoff_accepted++;
    } else {
        num_takeoff_refused++;
    }
    return result;
};

```

```

Runway_activity Runway::activity(int time, Plane &moving) {
    Runway_activity cur_state;
    if (!landings.empty()) {
        cur_state = land;
        moving = landings.front();
        landings.pop();
        num_land_finished++;
        land_wait_total += time - moving.started();
    } else if (!takeoffs.empty()) {
        cur_state = takeoff;
        moving = takeoffs.front();
        takeoffs.pop();
    }
}

```

```

        num_takeoff_finished++;
        takeoff_wait_total += time - moving.started();
    } else {
        cur_state = idle;
        idle_time_total++;
    }
    return cur_state;
};

```

```

void Runway::shut_down(int time) const {

```

```

    // cout << "C:" << fixed << setprecision(3) <<
    (double)num_land_refused/(double)num_land_requests << endl;

```

```

    // cout << "D:" << fixed << setprecision(3) <<
    (double)num_takeoff_refused/(double)num_takeoff_requests << endl << endl;

```

```

// (这里可以优化UI) ----- 可以考虑把输出导入到Excel表格

```

```

    cout << endl;
    cout << "Simulation has concluded after " << time << " time units." << endl

```

```

        << "Total number of planes processed during the Simulation: "
        << num_takeoff_requests+num_land_requests << "." << endl;
    cout << "Total number of planes asking to land: " <<
    num_land_requests << endl;

```

```

    cout << "Average rate of plane wanting to land: "
        << (double)num_land_requests/(double)time << endl;

```

```

    cout << "Total number of planes asking to takeoff: " <<
    num_takeoff_requests << endl;

```

```

    cout << "Average rate of plane wanting to takeoff: "
        << (double)num_takeoff_requests/(double)time << endl << endl;

```

```

    cout << "Total number of planes refused to land: " <<
    num_land_refused << endl;

```

```

    cout << "Average rate of planes refused to land: "
        << (double)num_land_refused/(double)num_land_requests << endl << endl;

```

```

    cout << "Total number of planes refused to takeoff: " <<
    num_takeoff_refused << endl;

```

```

    cout << "Average rate of planes refused to takeoff: "
        << (double)num_takeoff_refused/(double)num_takeoff_requests << endl << endl;

```

```

    cout << "Total number of planes accepted to land: " <<
    num_land_accepted << endl;

```

```

        cout << "Total number of planes accepted to takeoff:      " <<
num_takeoff_accepted << endl;
        cout << "Total number of planes landed successfully:      " <<
num_land_finished << endl;
        cout << "Total number of planes takeoff successfully:      " <<
num_takeoff_finished << endl;

        cout << "Total time all the planes wait(for landing):      " << land_wait_total <<
endl;
        cout << "Average time that every planes wait(for landing):      "
        << (double)land_wait_total/(double)num_land_finished << endl << endl;

        cout << "Total time all the planes wait(for taking off):      " << takeoff_wait_total
<< endl;
        cout << "Average time that every planes wait(for taking off):      "
        << (double)takeoff_wait_total/(double)num_takeoff_finished << endl << endl;

        cout << "Total time the Runway idle:      " <<
idle_time_total << endl;
        cout << "Percentage of time Runway idle:      "
        << (double)idle_time_total/(double)time << endl;

};

```