Blocking Probability Table 這邊是用.csv 檔來呈現，已經附了，如要了解各個詳細的機率請參考該表。
而這邊來簡單講解一下程式流程，一開始先把要放的容器都先準備好，把一些隨機的變數也先設定好，然後開始進入迴圈中，首先最外層的是 table 的數量，這邊會有兩個 table，所以會跑兩次，而再裡面那層的迴圈是 sever 數量，分別是 1,5,10，讓其執行可以得到那三個 server 數的寫法，再下去的兩層迴圈分別是 λ 跟 μ ，讓其跑出指定的數值。接下來裡面是模擬的流程，首先把 arrival 數跟 block 數先歸零，然後建立一個模擬十次的迴圈，裡面再開始模擬，一開始先把每個 arrival 的時間記錄下來，利用 c++內建的函式來生成，並且把它 push 進去 queue 裡面，直到 100000 秒，並且記錄有幾個 arrival，並且在 departure 的時間設定一個很大的時間，讓如果沒有 departure 的時候還有東西可以進行比較，接下來是分成了兩個部分，一個是 queue = 0 的情況，一個是 queue = S 的情況。先講 queue = 0 的情況，外面包一個 while 使得每一個 arrival 都順利進去，並且在裡面比較在目前的情況中，接下來看是要 arrival 還是 departure，如果是 arrival 發生的話，先看在 server 裡面的有沒有滿，如果沒滿的話就把這次的 arrival 加進去 sever 裡面並且產生那個 arrival 的 departure time，並且把其放入 minheap 中，這樣子就可以保證都是最小的先偵測到，做完之後把 in_server 的數量加一個，如果 sever 本來就滿的話就 block 掉那個 arrival 並且把 block 數+1，如果接下來是要做 departure 的話就只要把 minheap 最上面的拿出並且使 in_server 的數量減一個就好。再來講 queue = S 的情況，一樣是包一個 while 使得每一個 arrival 都順利進去，並且看接下來是 arrival 還是 departure，如果是 arrival 的話先看是哪一種情況，如果是 server 還沒滿的情況的話就把該次的 arrival 加進去 server 裡面並且產生那個 arrival 的 departure time，並且放入 minheap 中，如果是 server 滿了但是 queue 還沒滿的話就把其加進去 queue 當中，如果 sever 跟 queue 都滿的話就 block 掉那個 arrival 並且把 block 數+1，如果是 departure 的話先看 queue 裡面有沒有 arrival，如果有的話就把讓出來的位置給 queue 裡面的 arrival，並且記錄下其要 departure 的時間並且加入 minheap 中，做完之後就 pop 出那次的 departure，並且把 queue 的數量減一，如果 queue 裡面沒有 arrival 的話就 pop 出那次的 departure，並且把 in_server 的數量減一。最後利用一個迴圈把 departe 清乾淨，使其可以給下次使用。再來計算出 Blocking Probability 並且先暫存著，之後輸出成.csv 的檔案，裡面就會有 Blocking Probability Table 了。
這次的作業中學習到了很多東西，也比較了解了這整個過程的運作了，也透過這次的練習讓上課所學到的東西有一個實作出來的一個模擬的過程，對這個的過程也有更進一步的理解了。