

자료구조

~ Mini Project : Queue Simulation ~

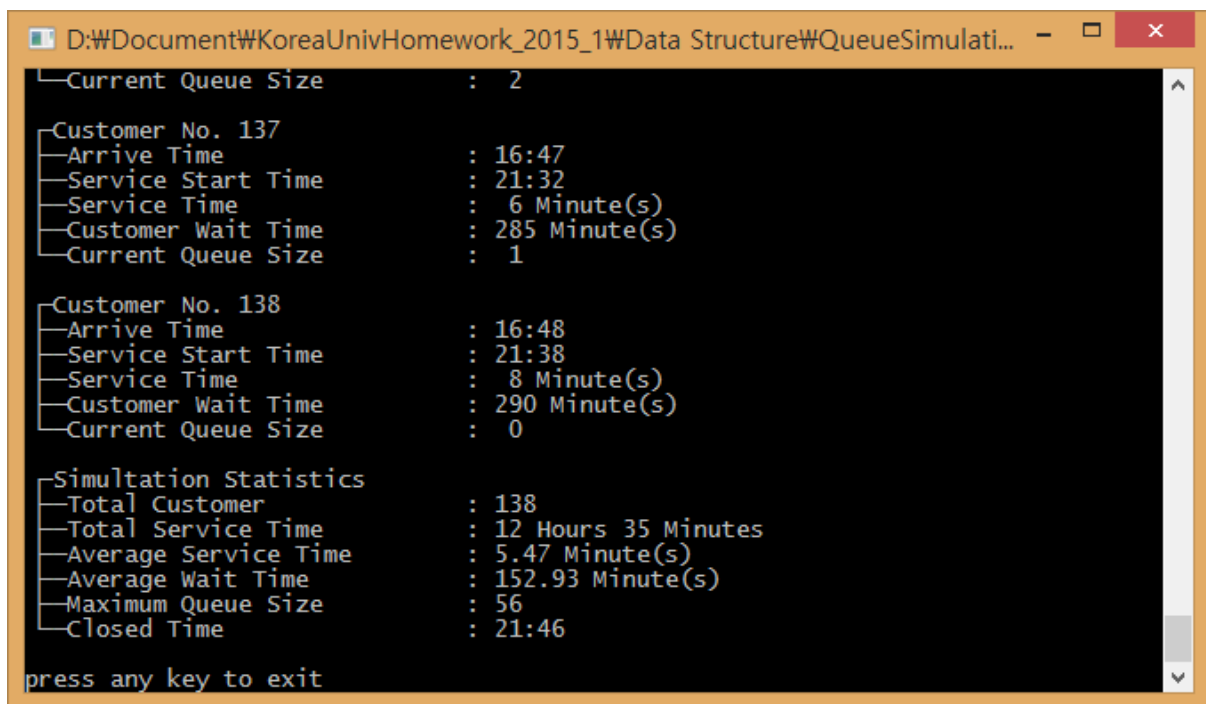
2013210111 남세현

// 개발환경 :: Windows 8.1, Visual Studio 2013 Prof. 버전에서 C언어 //

1. 목표

ADT를 이용하여 Queue를 만들고, 카페에서 1분당 0.25명이 들어오고 서비스하는데 1~10분 사이 걸린다고 했을 때의 시뮬레이션을 만드는 것이 목표입니다.

2. 실행 화면



```
Current Queue Size      : 2
Customer No. 137
  Arrive Time           : 16:47
  Service Start Time    : 21:32
  Service Time          : 6 Minute(s)
  Customer Wait Time    : 285 Minute(s)
  Current Queue Size    : 1
Customer No. 138
  Arrive Time           : 16:48
  Service Start Time    : 21:38
  Service Time          : 8 Minute(s)
  Customer Wait Time    : 290 Minute(s)
  Current Queue Size    : 0
Simulation Statistics
  Total Customer        : 138
  Total Service Time    : 12 Hours 35 Minutes
  Average Service Time  : 5.47 Minute(s)
  Average Wait Time     : 152.93 Minute(s)
  Maximum Queue Size    : 56
  Closed Time           : 21:46
press any key to exit
```

3. 구현 방법

Queue의 경우 교과서에도 구현이 나와있지만, 직접 한번 만들어보았습니다.

```

typedef struct Node
{
    int m_CustomerNumber;
    int m_ArriveTime;

    struct Node* next;

    int m_ServiceStartTime;
    int m_ServiceTime;
};

typedef struct Queue
{
    struct Node* m_Front;
    struct Node* m_End;
    int m_Size;
};

struct Queue* CreateQueue();
void PushBack(struct Queue* queue, struct Node* node);
struct Node* PopFront(struct Queue* queue);
int GetCurrentQueueSize(struct Queue* queue);

static void _PushBack(struct Queue* queue, struct Node* node);
static struct Node* _PopFront(struct Queue* queue);

```

큐 안에 들어갈 Node(Customer), 그리고 Queue를 정의하고, 그에 필요한 함수를 아래에 선언했습니다.

_PushBack과 _PopFront는 Private Function으로 동작하기 위해서 static function으로 만들었습니다.

내부 구현은 일반적인 Queue와 다르게 없으므로, PushBack과 PopFront 내부에서 _PushBack, _PopFront를 호출한다는 것 외에는 특별한 설명이 필요없으리라 봅니다.

메인 로직의 경우 저는 순서를 좀 바꾸었습니다.

```

for (nowClock = 9 * 60 + 0; // 09:00
    (nowClock <= 17 * 60 + 0 // 17:00
    || currentServingCustomer != NULL);
    nowClock++)
{
    if (nowClock <= 17 * 60 + 0)
        NewCustomer(myQueue, nowClock);
    OnServeFinished(myQueue, nowClock, &myStatus, &currentServingCustomer);
    ServeStart(myQueue, nowClock, &myStatus, &currentServingCustomer);
}

```

카페 문 닫는 시간 전까지 새 손님을 받는 부분 - NewCustomer - 은 같지만, 서빙을 시작하는 부분 - ServeStart - 와 서빙이 끝나는 부분 - OnServeFinished - 는 순서를 뒤바꿨습니다.

왜 그렇게 했냐면, 9시 45분에 서빙이 끝나면 다음 손님의 서빙이 45분부터 시작되기 때문입니다. 만약 순서를 뒤집지 않는다면, 45분에 서빙이 끝나면 46분부터 다른 사람의 서빙을 시작해야만 할 것입니다.

/*

저는 대체로 변수명이나 함수 명을 줄여쓰지 않는 편입니다.

그래서 별도의 주석 없이 많은 사람들이 이해할 수 있는 코드를 짜려고 노력합니다.

코드를 직접 보시면 아시겠지만, 불필요한 주석 없이 조교님이 읽으실 수 있는 수준으로 작성했으므로 "어!? 주석이 없네~ 감점!"처럼 하시진 않으실거라 믿습니다.

*/

4. 특이점

저는 Average Serving Time이 5.5대에서 머물고 있습니다. 왜냐하면 1부터 10까지의 숫자들의 평균은 5.5이기 때문입니다. 하지만 조교님 PPT에 보면 Average Serving Time이 4분 대로 나와있고, 그에 따라 End of Last Serving Time이 매우 이른 시간입니다. 그에 반하면 저는 9시 그 이상이 되어야 끝나곤 합니다.

Average Serving Time이 4분대로 나온 것은 우연의 일치이겠지만, 이것으로 알 수 있는 것은, 평균 대기시간이 조금만 빨라져도 총 서빙시간이 매우 빨라진다는 것입니다. 큐잉 이론에도 나오는 것으로 알고 있는데, Input Ratio가 증가하고, Queue에 머물고 있는 노드의 갯수가 늘어날수록 Queueing Time이 Exponentially 증가함을 볼 수 있습니다.