자료구조 실습과제 07-데큐/큐 응용

- 솔루션 및 프로젝트 명칭 : Proj_07_이름이니셜

- 제출방법 : 아래 문제를 해결하기 위한 프로그램을 구현한 후 컴파일 및 실행한 후, 오류가 없으면 메뉴에서 **솔루션 정리를 수행한 후 윈도우 탐색기에서 솔루션 폴더를 찾아 압축하여** E-class에 올림

각 문제를 테스트하기 위하여 아래와 같이 구현하시오.

```
#define _CRT_SECURE_NO_WARNINGS

#define PROB 1 // 각각의 문제를 구현하고 해당 문제 번호를 변경하여 테스트

#if PROB == 1
// 1번 문제를 해결하기 위한 프로그램 작성

#elif PROB == 2
// 2번 문제를 해결하기 위한 프로그램 작성

#endif
```

문제1) 배열 기반 원형 덱(deque)을 이용하여 수강 선호 교과목을 관리하는 프로그램을 작성하고자 한다.

수강 교과목에는 교과목 명, 담당 교수, 수업시간이 명시된다. 수업시간은 1시간 단위로 시수를 표시하며, 수업이 편성된 요일과 교시가 입력된다. 교재에서 소개된 deque의 예제 프로그램을 활용하여 선호 수강 교과목 리스트를 관리하는 프로그램을 작성하시오. add_front에는 우선 순위가 높은 교과목을, add_rear에는 우선 순위나 낮은 교과목을 추가한다. 아래 main 함수의 예제와 main 함수를 수행한 실행 예를 참고하여 deque를 관리하는 프로그램을 작성하시오.

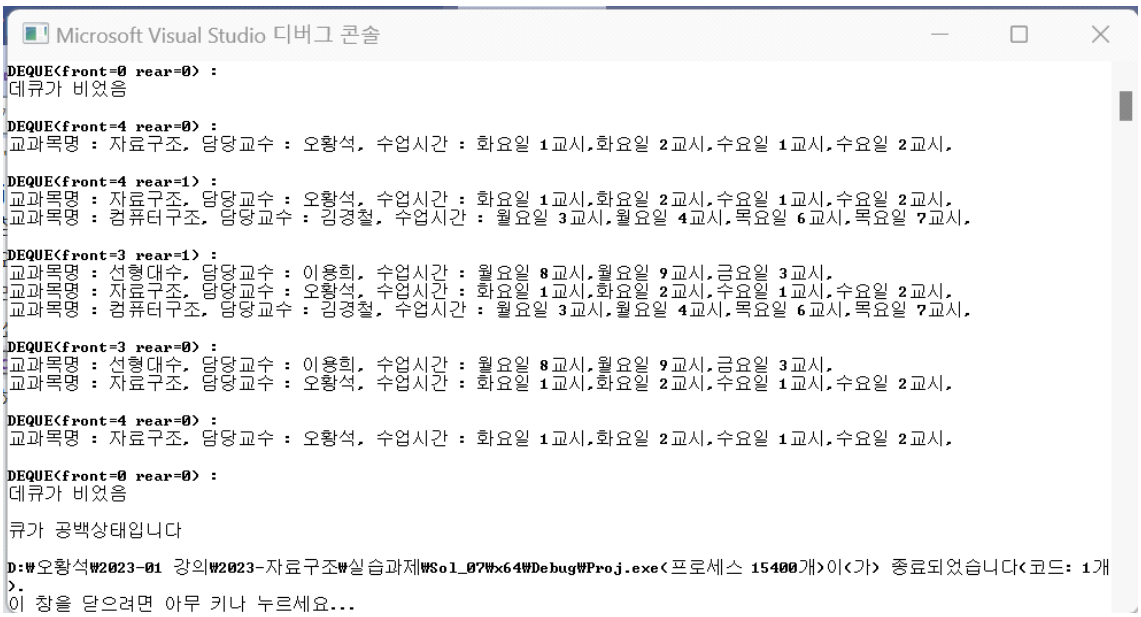
```
int main(void)
{
        DequeType queue;
        element data1 = {"자료구조","오황석",{4, {"화", 1, "화", 2, "수", 1, "수", 2}}};
        element data2 = {"컴퓨터구조","김경철",{4, {"월", 3, "월", 4, "목", 6, "목", 7}}};
        element data3 = {"선형대수","이용희",{3, {"월", 8, "월", 9, "금", 3}}};

        init_deque(&queue);
        deque_print(&queue);
        add_front(&queue, data1);
        deque_print(&queue);
        add_rear(&queue, data2);
        deque_print(&queue);
        add_front(&queue, data3);
        deque_print(&queue);

        delete_rear(&queue);
        deque_print(&queue);
        delete_front(&queue);
        deque_print(&queue);
        delete_front(&queue);
        deque_print(&queue);
        delete_rear(&queue);
        deque_print(&queue);

        return 0;
}
```

실행 예:


Microsoft Visual Studio 디버그 콘솔
DEQUE<front=0 rear=0> :
데큐가 비었음
DEQUE<front=4 rear=0> :
교과목명 : 자료구조, 담당교수 : 오황석, 수업시간 : 화요일 1교시,화요일 2교시,수요일 1교시,수요일 2교시,
DEQUE<front=4 rear=1> :
교과목명 : 자료구조, 담당교수 : 오황석, 수업시간 : 화요일 1교시,화요일 2교시,수요일 1교시,수요일 2교시,
교과목명 : 컴퓨터구조, 담당교수 : 김경철, 수업시간 : 월요일 3교시,월요일 4교시,목요일 6교시,목요일 7교시,
DEQUE<front=3 rear=1> :
교과목명 : 선형대수, 담당교수 : 이용희, 수업시간 : 월요일 8교시,월요일 9교시,금요일 3교시,
교과목명 : 자료구조, 담당교수 : 오황석, 수업시간 : 화요일 1교시,화요일 2교시,수요일 1교시,수요일 2교시,
교과목명 : 컴퓨터구조, 담당교수 : 김경철, 수업시간 : 월요일 3교시,월요일 4교시,목요일 6교시,목요일 7교시,
DEQUE<front=3 rear=0> :
교과목명 : 선형대수, 담당교수 : 이용희, 수업시간 : 월요일 8교시,월요일 9교시,금요일 3교시,
교과목명 : 자료구조, 담당교수 : 오황석, 수업시간 : 화요일 1교시,화요일 2교시,수요일 1교시,수요일 2교시,
DEQUE<front=4 rear=0> :
교과목명 : 자료구조, 담당교수 : 오황석, 수업시간 : 화요일 1교시,화요일 2교시,수요일 1교시,수요일 2교시,
DEQUE<front=0 rear=0> :
데큐가 비었음
큐가 공백상태입니다
D:₩오황석₩2023-01 강의₩2023-자료구조₩실습과제₩Sol_07₩x64₩Debug₩Proj.exe(프로세스 15400개)이(가) 종료되었습니다(코드: 1개).
이 창을 닫으려면 아무 키나 누르세요...

문제 2) 큐 자료구조를 이용하여 큐잉이론에 따라 문제 특성을 분석하여 시뮬레이션하여 분석 결과를 제시하는 프로그램을 작성하시오.

문제 상황 : 햄버거 가게에서 두 사람이 햄버거를 만들어 서비스 제공한다. 판매한 햄버거 종류별 개수와 해당 서비스를 하는 데 총 걸린 시간, 햄버거별 서비스를 제공하는데 걸린 평균시간, 주문이 취소된 건 수를 주어진 상황에서 분석하는 프로그램을 작성하시오.

- 주문은 매 7 시간마다 손님이 방문하여 주문을 한다. 주문할 수 있는 내용은 아래 표와 같다.

메뉴	소요소간
MENU 0	5
MENU 1	10
MENU 2	15
MENU 3	20
MENU 4	25

- 손님은 rand() % 5로 메뉴를 결정하고, 주문한다. 주문된 내용은 Queue에 저장된다. 주문 즉시 햄버거가 만들어지지 않고, 주문이 Queue에 들어가기 때문에 햄버거를 만드는 시간 보다 손님은 더 기다릴 수 있다.
- Queue가 가득 찬 경우에 손님이 주문할 경우 주문을 넣을 수 없어 해당 손님은 주문을 하지 않고 떠난다.
- 주문을 처리하는 두 사람(server1,server2)은 현재 처리하는 작업이 없으면 바로 Queue로부터 다음 주문을 추출하여 처리한다.
- 하나의 서비스가 완료되면, 손님이 주문을 한 시간부터 서비스가 완료되기까지의 시간을 구하여 대기시간으로 한다.
- 이러한 서비스를 WORK-TIME 10,000 동안 수행할 경우, 메뉴 선택 패턴이 동일하게 하고, 큐의 크기(주문을 저장할 수 있는 개수)를 10, 20, 30, 40, 50일 때 메뉴별 서비스한 개수, 누적대기시간, 평균서비스시간과 취소된 주문의 건 수, server1과 server2가 놀고 있는 시간을 분석하시오.

실행 예

큐의 크기가 10일 때	========== Results ========== 큐의 갯수 : 10 MENU 0 268개 서비스, 누적대기시간 16694, 평균서비스시간 62.291045 MENU 1 282개 서비스, 누적대기시간 18896, 평균서비스시간 67.007092 MENU 2 259개 서비스, 누적대기시간 18620, 평균서비스시간 71.891892 MENU 3 263개 서비스, 누적대기시간 20223, 평균서비스시간 76.893536 MENU 4 266개 서비스, 누적대기시간 21640, 평균서비스시간 81.353383 취소된 주문 81 개 서버1과 서버2 휴식 시간 : 6, 23 ==============================
큐의 크기가 20일 때	========== Results ========== 큐의 갯수 : 20 MENU 0 263개 서비스, 누적대기시간 33433, 평균서비스시간 127.121673 MENU 1 275개 서비스, 누적대기시간 35904, 평균서비스시간 130.560000 MENU 2 258개 서비스, 누적대기시간 34696, 평균서비스시간 134.480620 MENU 3 267개 서비스, 누적대기시간 38465, 평균서비스시간 144.063670 MENU 4 267개 서비스, 누적대기시간 38869, 평균서비스시간 145.576779 취소된 주문 78 개 서버1과 서버2 휴식 시간 : 6, 23 ==============================
큐의 크기	

가 30일 때	========== Results ========== 큐의 갯수 : 30 MENU 0 270개 서비스, 누적대기시간 49065, 평균서비스시간 181.722222 MENU 1 277개 서비스, 누적대기시간 50379, 평균서비스시간 181.873646 MENU 2 254개 서비스, 누적대기시간 46108, 평균서비스시간 181.527559 MENU 3 262개 서비스, 누적대기시간 52185, 평균서비스시간 199.179389 MENU 4 271개 서비스, 누적대기시간 53913, 평균서비스시간 198.940959 취소된 주문 63 개 서버1과 서버2 휴식 시간 : 6, 23 ================================
큐의 크기 가 40일 때	========== Results ========== 큐의 갯수 : 40 MENU 0 271개 서비스, 누적대기시간 60519, 평균서비스시간 223.317343 MENU 1 280개 서비스, 누적대기시간 62072, 평균서비스시간 221.685714 MENU 2 259개 서비스, 누적대기시간 56669, 평균서비스시간 218.799228 MENU 3 266개 서비스, 누적대기시간 64822, 평균서비스시간 243.691729 MENU 4 264개 서비스, 누적대기시간 62832, 평균서비스시간 238.000000 취소된 주문 49 개 서버1과 서버2 휴식 시간 : 6, 23 ================================
큐의 크기 가 50일 때	========== Results ========== 큐의 갯수 : 50 MENU 0 264개 서비스, 누적대기시간 69361, 평균서비스시간 262.731061 MENU 1 275개 서비스, 누적대기시간 70583, 평균서비스시간 256.665455 MENU 2 256개 서비스, 누적대기시간 64744, 평균서비스시간 252.906250 MENU 3 264개 서비스, 누적대기시간 75548, 평균서비스시간 286.166667 MENU 4 271개 서비스, 누적대기시간 75314, 평균서비스시간 277.911439 취소된 주문 49 개 서버1과 서버2 휴식 시간 : 6, 23 ================================

프로그램의 메인 루프 구조 : 아래 코드 참조

```
while (time < WORK_TIME) { // time을 1씩 증가시키면서
        if (time % 7 == 0) {// 새로운 손님 도착하여 주문
            //주문을 생성하여 큐에 넣기
        }

        if ( !isBusyServer1 ) { //놀고 있다가 새로운 서비스 시작
            …
        }
        else if (isBusyServer1 && server1.processing_time == 0) {//작업 끝남.
            …
        }

        if (!isBusyServer2) { //놀고 있다가 새로운 서비스 시작
             …
        }
        else if (isBusyServer2 && server2.processing_time == 0) {//작업 끝남.
             …
        }

        // 시간 업데이트
        time++;
        // server1의 시간 1 감소
        // server2의 시간 1 감소
}

// 결과 출력
printf("========== Results ============\n");
for (i = 0; i < 5; i++) {
    printf("MENU %d %d개 서비스, 누적대기시간 %d, 평균서비스시간 %f\n", …}
printf("취소된 주문 %d 개\n", q.lost_service);
printf("================================\n");
```