**Homework 9**

|  |
| --- |
| 그림입니다. 원본 그림의 이름: YU_UI_RGB-10.png 원본 그림의 크기: 가로 2256pixel, 세로 3047pixel 프로그램 이름 : Adobe ImageReady |

|  |  |
| --- | --- |
| 과목명 | 객체지향프로그래밍과자료구조 |
| 교수님 | 김영탁 교수님 |
| 이 름 | 김주환 |
| 학 번 | 21812158 |
| 일 자 | 2021.11.10.수 |

|  |
| --- |
| **(1) Critical section (임계구역) 설정이 필요한 이유에 대하여 구체적으로 설명하고, mutex 사용하여 어떻게 임계 구역을 관리하는지에 대하여 예를 들어 설명하라.**  다중 스레드 사용을 지원하는 운영체제는 프로그램 실행 중에 스레드 또는 프로세스 간에 교체가 일어날 수 있게 하여, 다수의 스레드/프로세스가 병렬로 처리될 수 있도록 관리하여야 한다. Context switching이 일어나면, 현재 실행 중이던 스레드/프로세스의 중간 상태가 임시 저장되고, 다른 스레드/프로세스가 실행된다. 프로그램 실행 중에 특정 구역은 실행이 종료될 때까지 스레드/프로세서 교체가 일어나지 않도록 관리하여야 하는 경우가 있다.  은행 입금 및 출금 스레드 예시를 통해 critical section으로 보호하여야 하는 구역을 알아보겠다.    위의 경우는 입금과 출금 중간에 context switching이 발생한 경우다.  공유자원을 스레드 A에서 사용하는 중에 스레드 B에서 사용했기 때문에 문제가 발생했다.  이런 문제를 해결하기 위해서 critical section을 설정해주겠다.    mutex를 통해 critical section을 지정해서 각 스레드가 공유자원을 사용하는 중에 context switching이 일어나지 않도록 구현했다.  아래의 함수를 통해 세마포를 선언하고, critical section설정을 시작하고, 종료한다. |
| **(2) Multi-thread에 파라메터를 전달하는 방법에 대하여 구체적으로 설명하라. 특히, Event의 생성에서 처리까지 경과된 시간을 측정하기 위하여 각 스레드에 어떤 파라메터를 전달하고, 각 스레드가 어떤 처리를 하여야 하는지에 대하여 설명하라.**  Multi-thread에 파라메터를 전달하기 위해 필요에 따라 파라메터 항목들을 포함하는 구조체를 정의한다. 기본적으로는 mutex에 관련된 정보, 공유되는 큐의 정보, 파일 입출력에 관련된 정보를 포함한다.    특히 Event의 생성에서 처리까지 경과된 시간을 측정하기 위해서는 각 스레드에 특정 파라메터를 전달하여야 한다. 이에 대해 알아보겠다.  Event class는 아래의 시간 정보들을 가진다.    생성 스레드에서 이벤트 생성 시간을 저장한다.    처리 스레드에서 이벤트 처리 시간을 저장하고, 이전에 저장했던 생성시간과의 차이를 이용하여 생성으로부터 처리까지 걸린 시간을 구합니다. |
| **(3) Multi-thread의 동작 상태를 monitoring하여, 주기적으로 상태를 출력하는 함수를 구상하고, 필요한 파라메터 전달, 출력 포맷에 따른 주기적인 출력 방법에 대하여 설명하라.**    스레드의 진행 상황을 파악하기 위한 ThreadStatusMonitor구조체를 정의한다.  main() 함수에서 ThreadStatusMonitor 구조체 변수를 생성하고, 그 구조체 변수의 주소를 스레드 생성의 인수로 전달한다.  각 스레드의 진행상황(예 : 이벤트 생성, 이벤트 처리 등)을 ThreadStatusMonitor 구조체 변수에 기록한다.  main() 함수에서 주기적으로 threadStatusMonitor 구조체 변수의 내용을 파악 및 분석한다.  전체 스레드의 진행 상황을 한 눈에 쉽게 볼 수 있도록 현황판 형식으로 콘솔에 출력한다.  main문 내부의 반복문에서 각 라운드마다 주기적으로 상태를 출력합니다. 이 때 매 반복마다 ConsoleDisplay파일의 cls()로 실행창을 초기화하고 gotoxy()로 (0, 0)에서부터 다시 출력합니다. |
| **(4) 동적으로 할당된 배열을 기반으로 구현된 Circular Queue의 기본 기능인 enQueue()와 deQueue() 함수의 기본 구조 및 동작 원리에 대하여 그림으로 나타내어 설명하라.**  Circular Queue는 배열의 시작과 끝을 연결함으로 구현된다.   * enQueue()   1. Queue에 공간이 있는지 확인하고 없다면 NULL을 반환한다.  2. 그렇지 않다면 Event를 back에 집어넣는다.  3. back을 1만큼 증가시키고 capacity를 초과한다면 %연산으로 위치를 조정해준다.  4. num\_element를 1만큼 증가시킨다.  삽입한 element의 포인터를 반환한다.   * deQueue()   1. Queue가 비어있는지 확인하고 비어있다면 NULL을 반환한다.  2. 그렇지 않다면 front의 Event의 포인터를 저장받는다.  3. front를 1만큼 증가시키고 num\_element를 1만큼 감소시킨다.  4. front가 capacity를 초과한다면 %연산으로 위치를 조정해준다.  5. ‘2.’에서 저장받은 element의 포인터를 반환한다. |
| **(5) 우선 순위를 고려한 Event처리를 위하여 사용되는 Priority Queue에서 우선 순위가 높은 event가 우선적으로 처리될 수 있는 구조와 동작 원리에 대하여 설명하라.**  구조   * Heap은 각 node에 Key를 저장하면서 아래의 특성을 만족하는 Complete Binary Tree다. * Heap-Order : root를 제외한 모든 node는 부모보다는 큰 Key값을 가지고, 자식보다는 작은 Key값을 가진다. * CBT : h를 heap의 높이로 한다. I = 0, …… , h - 1에서 각 depth : i에는 2^i개의 node가 존재한다. * heap의 last node는 maximum depth의 rightmost node다.   동작 원리   * up-heap bubbling(Insert) 1. Event 생성 및 last node에 삽입 2. last node와 부모 node의 우선 순위 비교 3-1. child node의 우선 순위가 높을 경우 swap  2 ~ 3-1 반복 4. 삽입한 node가 root에 도달한 경우 break 3-2. 삽입한 node의 우선 순위가 낮을 경우 break      * down-heap bubbling(Remove) 1. root node remove 2. last node를 root로 지정 3. root부터 아래로 우선순위 비교 4-1. 부모의 우선순위가 높을 경우 break 4-2. 자식의 우선순위가 높을 경우, 높은 우선순위를 가지는 child과 swap  3 ~ 4 반복 5. 최초의 last node가 다시 last node에 위치한다면 break |