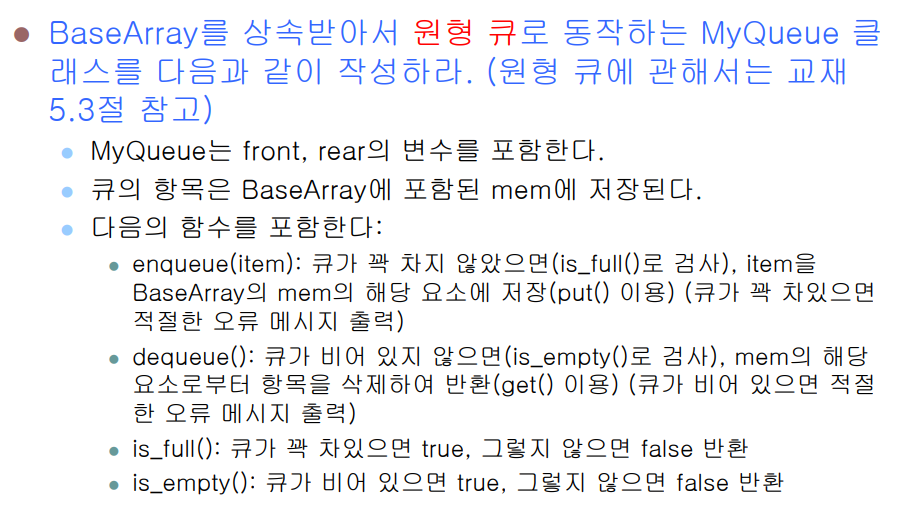


객체지향프로그래밍

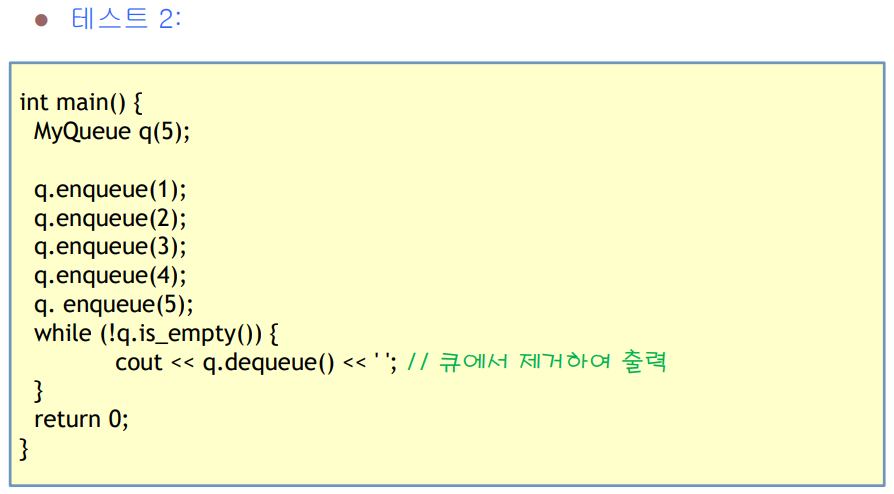
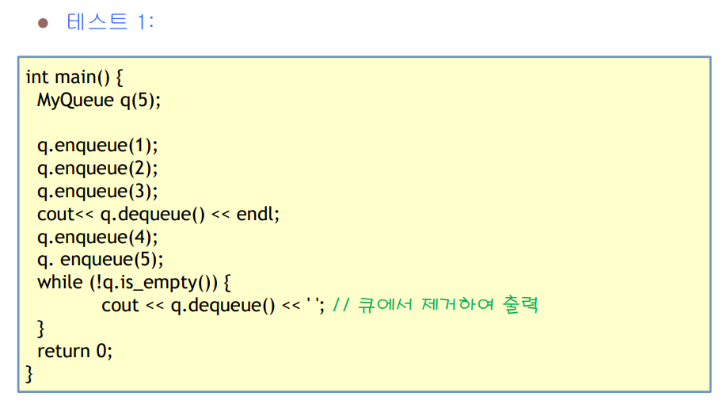
Report #6

|  |  |
| --- | --- |
| 학번 | 20174627 |
| 이름 | 김혜진 |

문제



main() 테스트 예제



전체 코드 및 설명(주석)

|  |
| --- |
| #include<iostream>  using namespace std;  class BaseArray {  private:  int capacity; // 동적 할당된 메모리 용량  int\* mem;  protected:  BaseArray(int capacity) {  this->capacity = capacity; mem = new int[capacity];  }  ~BaseArray() { delete[] mem; }  void put(int index, int val) { mem[index] = val; }  int get(int index) { return mem[index]; }  int getCapacity() { return capacity; }  };  class MyQueue : BaseArray {  private:  //원형큐의 index는 2가지, 삽입하면서 사용할 real 빼면서 사용할 front  int front, real;  public:  //자식생성자 호출시 부모 생성자도 호출 Java의 Super  MyQueue(int capacity) : BaseArray(capacity) {  //원형 큐의 front, real 값 index 0으로 설정  this->front = 0;  this->real = 0;  }  bool is\_full() {  //원형큐의 real+1 % Capacity 가 front 일시 꽉참  if ((real + 1) % getCapacity() == front) {  return true;  }  else {  return false;  }  }  bool is\_empty() {  //front와 real의 index가 같다면 비어있음  if (front == real) {  return true;  }  else {  return false;  }  }  void enqueue(int item) {  //만약 꽉차지 않았다면  if (!is\_full()) {  //real의 값을 +1%Capacity index로 설정, %Capacity를 해야 원형적인 index값을 가질 수 있음  real = (real + 1) % getCapacity();  //부모 클래스의 put호출, index는 real%Capacity, 그리고 item 삽입  put(this->real % getCapacity(), item);  }  else {  cout << "Queue is full" << endl;  }  }  int dequeue() {  //먄약 비어있지 않다면  if (!is\_empty()) {  //뺄 값은 front의 다음번째 index 하지만 원형큐이므로 index값이-> %Capacity 한 값을 추출  int value = get((front + 1) % getCapacity());  //front의 값을 +1 해주고, %Capacity만큼 더해줌  front = (front + 1) % getCapacity();  //값을 Retrun  return value;  }  else {  cout << "Queue is empty" << endl;  }  }  };  // 테스트1  int main() {  MyQueue q(5);  q.enqueue(1);  q.enqueue(2);  q.enqueue(3);  cout << q.dequeue() << endl;  q.enqueue(4);  q.enqueue(5);  while (!q.is\_empty()) {  cout << q.dequeue() << ' '; // 큐에서 제거하여 출력  }  return 0;  }  // 테스트 2  /\*  int main() {  MyQueue q(5);  q.enqueue(1);  q.enqueue(2);  q.enqueue(3);  q.enqueue(4);  q.enqueue(5);  while (!q.is\_empty()) {  cout << q.dequeue() << ' '; // 큐에서 제거하여 출력  }  return 0;  }\*/ |

결과

|  |
| --- |
| [테스트1]    [테스트2] |

느낀 점

원형 큐에 대해 자세하게 공부할 수 있게 되어 좋았습니다.