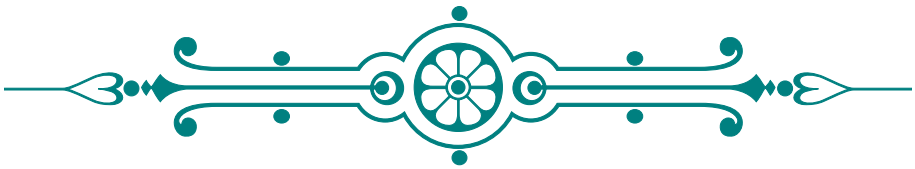


객체 지향 언어 및 실습

학번 : 2016110056

학과 : 불교학부

이름 : 박승원

날짜 : 2017년 4월 21일

문제 1

- 다음 코드에서 오류를 찾고 이를 수정하라. (보고서 기입)

```
public class Bike
{       protected
        private int gear;
        public int speed;
}                                   extends
public class MountainBike expands Bike
{
        public int seatHeight;
        public void MountainBike(int g)
        {
                super();
                gear=g;
        }
}
```

문제 2

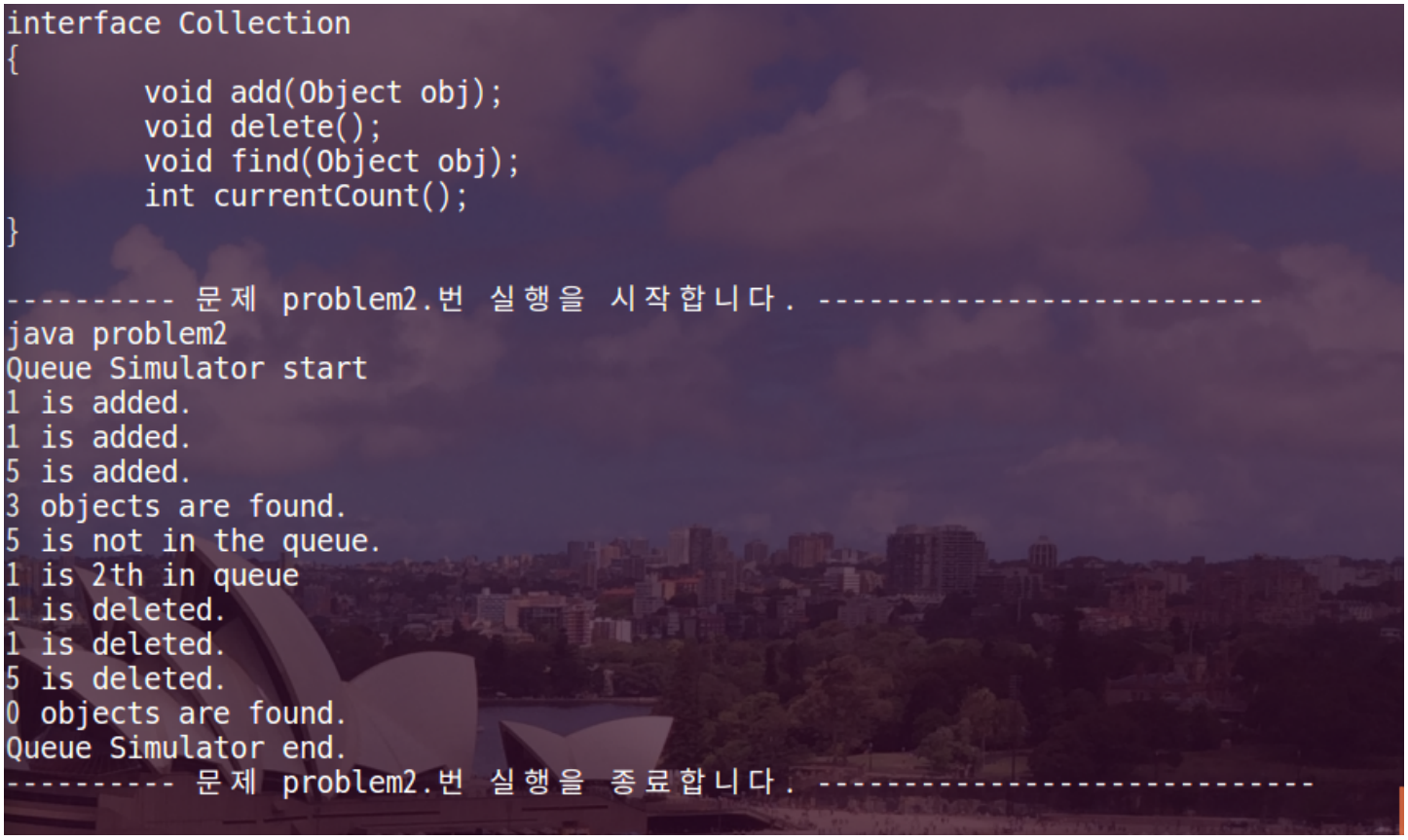
- 빈칸을 채워 큐 클래스의 코드를 완성하시오.

```
class queue implements collection {
    private Link head = null;
    private Link tail = null;
    private Link ptr = null;
    private int count = 0;

    // 빈칸 채워 넣기
    {
    }
```

```
// 2016110056  박승원

interface Collection
{
    void add(Object obj);
    void delete();
    void find(Object obj);
    int currentCount();
}

class Link
{
    Object data;
    Link next;

    Link(Object d, Link n) {
        data = d;
        next = n;
    }
}

class Queue implements Collection
{
    private Link head = null;
    private Link tail = null;
    private Link ptr = null;
    private int count = 0;

    public void add(Object obj) {
        if(count == 0) {
            head = new Link(obj, null);
            tail = head;
        } else {
            tail.next = new Link(obj, null);
            tail = tail.next;
        }
        tail.data = obj;
        count++;
        System.out.println(obj + " is added.");
    }

    public void delete() {
```

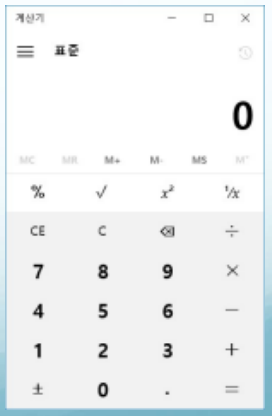
```
        System.out.println(head.data + " is deleted.");
        head = head.next;
        if(head == tail) tail = head;
        count--;
    }

    public void find(Object obj) {
        int i = 0;
        Link l;
        for(l = head; l != null; l = l.next) {
            i++;
            if(l.data == obj) {
                System.out.println(obj + " is " + i + "th in queue");
                break;
            }
        }
        if(l == null) System.out.println(obj + " is not in the queue.");
    }

    public int currentCount() {
        System.out.println(count + " objects are found.");
        return count;
    }
}

public class problem2
{
    static public void main(String[] args) {
        System.out.println("Queue Simulator start");

        Queue q = new Queue();
        Integer a = new Integer(1);

        q.add(new Integer(1));
        q.add(a);
        q.add(new Integer(5));

        q.currentCount();
        q.find(new Integer(5));
        q.find(a);

        q.delete();
```

```
        q. delete () ;
        q. delete () ;

        q.currentCount() ;

        System.out. println ("Queue Simulator end.") ;
    }
}
```



문제 3

- 간단한 계산기를 작성하여 보자. 계산 기능은 나중에 추가하기로 하자. 여기서는 외관만 구현하면 된다. 계산기의 모양은 각자 컴퓨터의 계산기나 평소 사용하는 계산기 모양을 각자 참고하면 된다. oracle을 참고해서 GUI 구현의 다양한 시도를 해보자.
- 이는 이후 과제에 사용 될 예정입니다.
- 반드시 추가해야 할 버튼 목록
  - 입력 숫자 0 ~ 9
  - 덧셈(+), 뺄셈(-), 나눗셈(/), 곱셈(*), 등호(=)
  - . (소숫점)
  - 계산 결과 출력화면
  - 타이틀에 자신의 학번 넣기




```
// 2016110056  박승원
import javax.swing.*;
import java.awt.*;

class Calc extends JFrame {
    final int W = 90, H = 60;
    public Calc() {
        setSize(W * 4, H * 5);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("2016110056  박승원 ");

        JPanel p = new JPanel();
        p.setLayout(null);
        add(p);

        JTextField jt = new JTextField();
        p.add(jt);
        jt.setBounds(0, 0, W*4, H);

        String s = "789+456-123*.0/=";
        JButton[] bts = new JButton[s.length()]; // this just acquired memory space
```

```
        for(int i=0; i<s.length(); i++) //so I need to use constructor again !!!
            bts[i] = new JButton(Character.toString(s.charAt(i)));
        for(int j = 0, y = 0; y < 4 * H; y += H) {
            for(int x = 0; x < 4 * W; x += W) {
                p.add(bts[j]);
                bts[j++].setBounds(x, H + y, W, H);
            }
        }
         setVisible (true);
    }
}

public class problem3
{
    static public void main(String[] args) {
        Calc c = new Calc();
    }
}
```