Practice Problem on Event Handling.

Class MathQuiz creates a simple window than should be made to work together with the Problem class to display random problems, allow user to enter their answer, checks correctness.

```java
/** * The class implements a window with GUI components * Shows how to add components to a frame. */

import java.awt.*;  // some helper classes like Color, LayoutManager, etc.
import javax.swing.*;  // library of components

public class MathQuiz extends JFrame {
    // Define instance vars, simultaneously initializing them
    // to let user enter answer
    private JTextField tfAnswer = new JTextField(15);

    //to show if correct or not, and correct answer if wrong
    private JTextArea answers = new JTextArea(2, 20); // rows =2, cols=20

    private JButton question = new JButton("Get question");
    private JButton answer = new JButton("Submit");
    private JButton exit = new JButton("Exit");
    private JRadioButton add = new JRadioButton("Addition", true);
    private JRadioButton subtract = new JRadioButton("Subtraction", false);

    /** Constructor */
    public MathQuiz (){
        // recall: default superclass no-args
        // constructor is called here.
    }
```
// JFrame() creates a frame, initially invisible.

// set a few properties of the window
this.setSize(250, 275);
this.setTitle("Simple Math Quiz");
this.setDefaultCloseOperation(EXIT_ON_CLOSE);

// TO add the components to the container pane of the frame
// tell the pane how to arrange components:
// using FlowLayout
this.setLayout(new FlowLayout());

// add radio buttons
this.add(add);
this.add(subtract);

// Group radio buttons together in a group
// to ensure *single* selection at any time:
ButtonGroup operations = new ButtonGroup();

operations.add(add);
operations.add(subtract);

// add other components
this.add(question);

JLabel lQuestion = new JLabel(" a problem ");
this.add(lQuestion);
lQuestion.setForeground(Color.BLUE);

JLabel label = new JLabel("Enter answer");
this.add(label);
this.add(tfAnswer);
this.add(answer);

answers.setEditable(false);
this.add(answers);
this.add(exit);

this.setVisible(true);
}

public static void main(String[] args) {

    // Create an instance of the Math Quiz
    new MathQuiz();
}

Result: A frame that contains GUI components, but doesn’t do anything in response…
Problem: add functionality to the MathQuiz interface to
1. Have the program exit on ‘Exit’ button.
2. Display a problem (see class Problem below) in place of the blue label ‘a problem’ (by setting the text of that label to the String description of the problem), whenever the ‘Get Question’ button is pressed. The problem should be either an addition or subtraction problem, according to the operation selected with the radio buttons. At the same time clear the text field and bottom text area.
3. Read contents of the JTextField when the ‘Submit’ button is pressed. At the same time, display if answer was correct in the bottom text area.

```java
/* Class Problem. Creates an arithmetical problem with a solution.*/
public class Problem {
    private int op1;
    private int op2;
    private char operator;

    // generate a math problem
    public Problem(char operator){
        this.op1 = 1 + (int)(20 * Math.random()); // between 1 and 20
        this.op2 = 1 + (int)(20 * Math.random()); // between 1 and 20
        this.operator = operator;
    }

    // returns an answer to the problem.
    public int getAnswer(){
        int answer;
        if (this.operator == '+')
            answer = op1 + op2;
        else if (this.operator == '-')
            answer = op1 - op2;
        else
            answer = op1 * op2;
        return answer;
    }

    public String toString(){
        return op1 + " " + operator + " " + op2 + " = ??";
    }

    // main method shows functionality of Problem class.
    public static void main (String[] args){
        for (int i = 0; i<= 5; i++){
            Problem p = new Problem ('+');
            System.out.println(p);
            System.out.println(p.getAnswer());
        }
    }
}
```