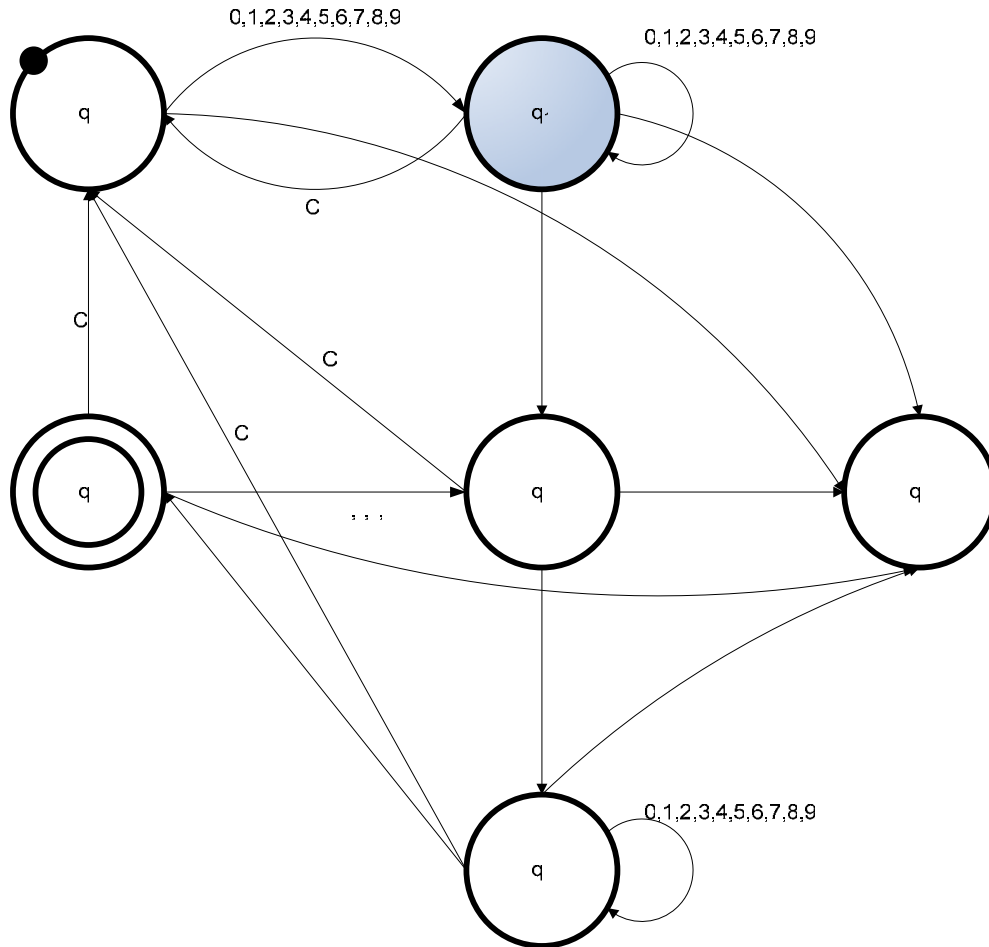


Aufgabe 8: Implementierung eines Ereignis-Automaten

- a) Modellieren Sie einen Ereignis-Automaten für den in der Abbildung dargestellten und in Calculator.java bereits teilweise implementierten Taschenrechner.



$$A = \{\Sigma, Q, \delta, q_0, F\}$$

$$\Sigma = \{0,1,2,3,4,5,6,7,8,9,+,-,*,/,C,X\}$$

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$$

$$F = q_4$$

Die Übergangsfunktion δ wird durch den oben abgebildeten Zustandsgraphen beschrieben.

Erläuterung der Zustände:

q_0 : Initialisierter Zustand, bei dem noch keine Eingaben gemacht wurden und sich alle Variablen im Ursprungszustand befinden

q_1 : Zustand nach der Eingabe einer Zahl(enfolge)

q_2 : Zustand, nachdem ein Operator (+, -, *, /) ausgewählt wurde

q_3 : Zustand nach der Eingabe einer zweiten Zahl(enfolge)

- q₄: Endzustand mit dem berechneten Ergebnis
q₅: Das Programm wurde durch den Schließen-Button („X“) beendet

Wird von q₄ in q₂ gewechselt, so ist das berechnete Ergebnis aus q₄ die erste Zahl der neuen Berechnung (vgl. Aufgabenstellung).

- b) In diesem Aufgabenteil soll der in Calculator.java bereits teilweise implementierte Taschenrechner um die Implementierung eines Ereignis-Automaten (entsprechend dem bereits in Aufgabenteil 9a) modellierten) erweitert werden. Bei der Implementierung ist darauf zu achten, dass alle Buttons, die in den einzelnen Zuständen nicht benötigt werden, deaktiviert sind. Bitte beachten Sie bei Ihren Tests, dass die Integer Werte in Java nur eine Größe von 32 Bit besitzen. Sie brauchen den Automaten nicht auf Überlaufbehandlung erweitern.

Vorbemerkung: Aus offensichtlichen Gründen werden in dem hier vorliegenden Programm nur die Zustände 0 bis 4 abgebildet. Ein Wechsel auf Zustand 5 entspricht der Terminierung des Programms.

Hinweis: Einige Zeilen mussten für ein ansprechendes Druckergebnis umgebrochen werden. Dieser Umbruch wurde durch einen Unterstrich („_“) dargestellt und gehört nicht zum eigentlichen Quelltext.

```
import java.awt.*;
import java.awt.event.*;

public class Calculator extends Frame
{
    //Variablendeklaration

    // Textfeld erzeugen
    TextField textField = new TextField();
    // Button Beschriftung
    String[] buttonLabelArray = { "+", "-", "*", "/", "1", "2", "3", "4", "5", "_",
        "6", "7", "8", "9", "0", "=", "C" };
    // 16 Buttons des Taschenrechners
    Button[] buttonArray = new Button [16];
    //derzeitiger Status
    int state;
    //Zahlen und Operator für die Berechnungen
    int zahl1 =0;
    int zahl2 = 0;
    char operator=' ';

    // Taschenrechner Konstruktor
    public Calculator()
    {
        this.setTitle("Calc"); //Windows Titel setzen
        // Listener fuer das Fenster
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0); //Fenster schließen (somit Zustands-
                //wechsel auf q5)
            }
        });
    }
}
```

```

    }
});

// Fenstergroesse festlegen
setSize(150,200);

// Textfeld
textField.setText("");
textField.setVisible(true);
textField.setEnabled(false);

// Panel für die Button erzeugen
Panel centerPanel = new Panel();
centerPanel.setLayout(new GridLayout(4,4));

// Layout festlegen und Komponeten hinzufuegen
setLayout(new BorderLayout());
this.add(centerPanel, BorderLayout.CENTER);
this.add(textField, BorderLayout.NORTH);

// Buttons mit Labeln und ActionListener generieren
for (int i = 0; i<16; i++)
{
    buttonArray[i] = new Button(buttonLabelArray[i]);
    centerPanel.add(buttonArray[i]);
    buttonArray[i].addActionListener(new buttonActionListener());
}
setState(0);
}

// Listener fuer die Buttons
class buttonActionListener implements ActionListener
{
    public void actionPerformed(ActionEvent event)
    {
        // ermitteln welcher Button gedruickt wurde
        char Caption = event.getActionCommand().charAt(0);
        if (("1234567890").indexOf(Caption)>=0)
            //Es wurde ein Ziffernbutton gedrückt
            {
                textField.setText(textField.getText() + Caption);
                //Darstellung im Textfeld anpassen
                if(getState()==0)
                    //Initalisierungszustand => erste Zifferneingabe
                    setState(1); //Neuer Zustand: Möglichkeit der
                    //Eingabe von Operatoren
                if(getState()==2) //Weitere Zifferneingabe (xte Zahl)
                    setState(3); //Neuer Zustand: Möglichkeit der
                    //Berechnung des Endergebnisses
            }
        else if (("+-*/").indexOf(Caption)>=0)
            //Es wurde ein Operatorenbutton gedrückt
            {
                operator = Caption; //Der Operator wird gespeichert
                setState(2); //Neuer Zustand: Möglichkeit der
                //weiteren Zifferneingabe
            }
    }
}

```

```

    }
    else
        switch (Caption)
        {
            case 'C':
                setState(0);//Zustand: Initalisierung
                break;
            case '=':
                setState(4);//Berechnung des Ergebnisses
        }
    }
}

/**
 * Aktiviert bzw. deaktiviert die Steuerelemente, mit denen
 * Operatoren ausgewählt werden können
 * @param enabled    boolean    Gibt an, ob die Steuerelemente
 *                               deaktiviert oder aktiviert sind
 */
public void setOperatorenState(boolean enabled)
{
    for(int i = 0; i<4;i++)
        buttonArray[i].setEnabled(enabled);
}

/**
 * Aktiviert bzw. deaktiviert das Steuerelemente, mit welchem
 * das Ergebnis berechnet werden kann (=)
 * @param enabled    boolean    Gibt an, ob das Steuerelement
 *                               deaktiviert oder aktiviert ist
 */
public void setGleichState(boolean enabled)
{
    buttonArray[14].setEnabled(enabled);
}

/**
 * Aktiviert bzw. deaktiviert die Steuerelemente, mit denen
 * Ziffern ausgewählt werden können
 * @param enabled    boolean    Gibt an, ob die Steuerelemente
 *                               deaktiviert oder aktiviert sind
 */
public void setZahlenState(boolean enabled)
{
    for(int i = 4; i<15;i++)
        buttonArray[i].setEnabled(enabled);
}

/**
 * Ermittelt den aktuellen Zustand, in dem sich das Programm
 * befindet.
 * @return int    Liefert den aktuellen Zustand zurück
 */
public int getState()
{
    return state;
}

```

```

}

/**
 * Methode zum Wechseln der Zustände
 * @param nextState int Der neue Zustand des Programms
 */
public void setState(int nextState)
{
    this.state =nextState; //zuweisen des neuen Zustands
    switch(nextState) //In Abhängigkeit vom neuen Zustand
        //Operationen ausführen
    {
        case 0: //Initialisierungsmodus
            setOperatorenState(false); //Operatorenbuttons
            //deaktivieren
            setGleichState(false); //=-Button deaktivieren
            setZahlenState(true); //Ziffernbutton aktivieren
            textField.setText(""); //Textfeldinhalt löschen
            zahl1=0; //Rücksetzen der Variablen
            zahl2=0;
            operator=' ';
            break;
        case 1: //Es wurden Zahlen eingegeben
            setOperatorenState(true); //Operatorenbuttons aktivieren
            break;
        case 2: //Der erste Operator
            zahl1 = Integer.parseInt(textField.getText()); //Erste
            //Zahl einlesen
            setOperatorenState(false); //Operatorenbtms. deaktivieren
            setZahlenState(true); //Ziffernbuttons aktivieren
            setGleichState(false); //Gleichheitsbutton deaktivieren
            textField.setText(""); //Textfeldinhalt löschen

            //Um eine Division durch Null zu verhindern, wird der
            //0-Button bei der Auswahl des Divisionsoperators
            //gesperrt
            if(operator=='/')
                buttonArray[13].setEnabled(false);
            break;
        case 3: //Zweite Zahl wird eingegeben
            setZahlenState(true); //Ziffernbuttons aktivieren
            //(um evtl. gesperrte 0 wieder frei zu geben)
            setGleichState(true); //Gleichheitsbutton aktivieren
            break;
        case 4: //Ausrechnen
            zahl2 = Integer.parseInt(textField.getText()); //Zweite
            //Zahl einlesen
            int ergebnis = 0; //Ergebnisvariable initialisieren
            switch(operator) //Ausrechnen
            {
                case '+':
                    ergebnis = zahl1+zahl2;
                    break;
                case '-':
                    ergebnis = zahl1-zahl2;
                    break;
            }
    }
}

```

```

        case '/':
            ergebnis = zahl1/zahl2;
        case '*':
            ergebnis = zahl1*zahl2;
            break;
    }
    zahl1=ergebnis; //Das ausgerechnete Ergebnis für
                //weitere Berechnungen zwischenspeichern
    this.textField.setText(String.valueOf(ergebnis));
                //Ergebnis anzeigen
    setOperatorenState(true); //Nur Operatoren als
                //Eingabe zulassen (neben C und X)
    setGleichState(false);
    setZahlenState(false);
    break;
    }
}

//Main Methode erzeugt ein Frame
public static void main(String[] args)
{
    Frame calculatorFrame = new Calculator();
    calculatorFrame.setVisible(true);
}
}

```