МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики. Кафедра програмного забезпечення комп'ютерних систем

ЗВІТ
з лабораторної роботи № 2
«Створення обробників для елементів управління у android-застосунках»

Виконав:
студент 3-го курсу, групи КП-83,
спеціальності 121 – Інженерія
програмного забезпечення
Коваль Андрій Олександрович

Перевірив:
к. т. н, старший викладач
Хайдуров Владислав
Володимирович

Київ – 2020

ВСТУП

Метою даної лабораторної роботи є ознайомлення із основними принципами та методами створення обробників для елементів управління в Android-застосунках. За цю роботу створено декілька додатків, що допомагає значно швидше розібратися з розробкою мобільних застосунків.

Завдання до лабораторної роботи

1. Ознайомитись із усіма теоретичними відомостями до лабораторної роботи.
2. Розробити функціональну частину для простого калькулятора, інтерфейс якого був розроблений у попередній лабораторній роботі. Для кожної кнопки створити власний обробник подій.
3. Створити копію проекту, який повністю реалізовано у пункті 2 замінити обробник подій для кожної кнопки одним обробником для масиву кнопок, що мають текстові надписи 0, 1, 2, ..., 9. Обробник виділити окремим методом (функцією). У звіт додати програмний код та результати роботи Android-програми.
4. Створити аналогічний калькулятор, що має два текстові поля, у які користувач самостійно вводить два числа. Для вибору операцій (+, -, *, /) а головному вікні (Main_Activity) створити Spinner. Для виконання обрахунків додати до інтерфейсу кнопку (Button). Інтерфейс має бути подібний до такого:

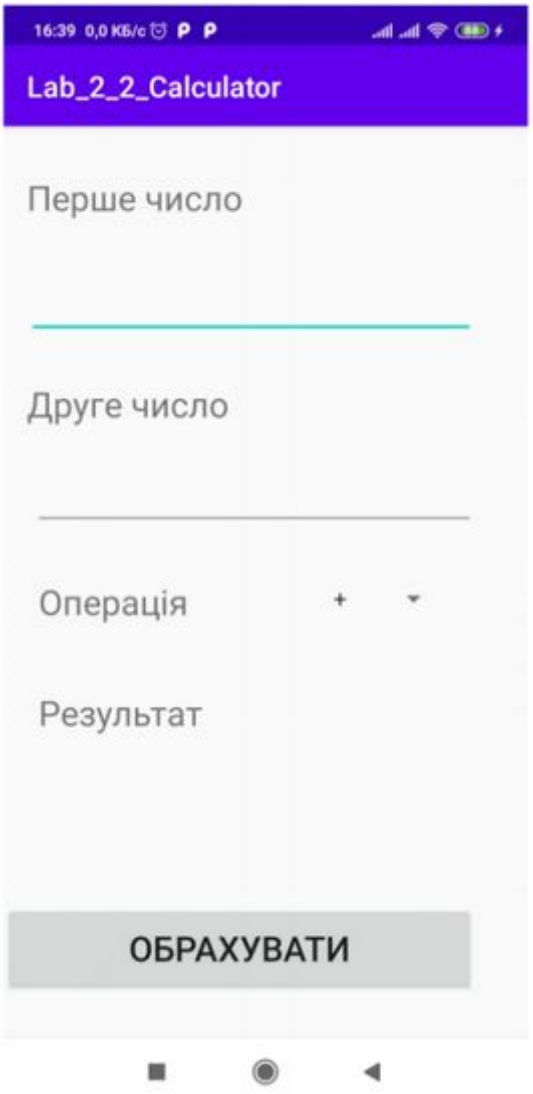

Рис. 1, Приклад інтерфейсу завдання 4

5. За аналогією до пункту 2 створити простий проект гру хрестики-нулики для поля розміром 4x4. Кожна клітинка повинна мати список Spinner. У звіт додати програмний код та результати роботи Android-програми.
6. Оформити звіт до даної лабораторної роботи.

Короткі теоретичні відомості

Існує 3 способи створення обробників подій:

- атрибут onClick
- метод setOnClickListener()
- реалізація інтерфейсу View.OnClickListener

Найбільш часто використовуються перший та другий способи оскільки вони є найшвидшими та найпростішими в реалізації. У даній роботі ми будемо використовувати перший та другий спосіб. Третій частково зв’язаний з другим, а саме випливає з нього, але не будемо його враховувати а будемо використовувати як частинний випадок перших двох.

Для будь-якого вибору у Android можуть використовуватися різні варіанти. Один з них є найпростіший список, що випадає схожий на <select> з веб програмування. Ми будемо використовувати саме його для вибору необхідних математичних операцій.

Програмна реалізація задачі

Завдання 2

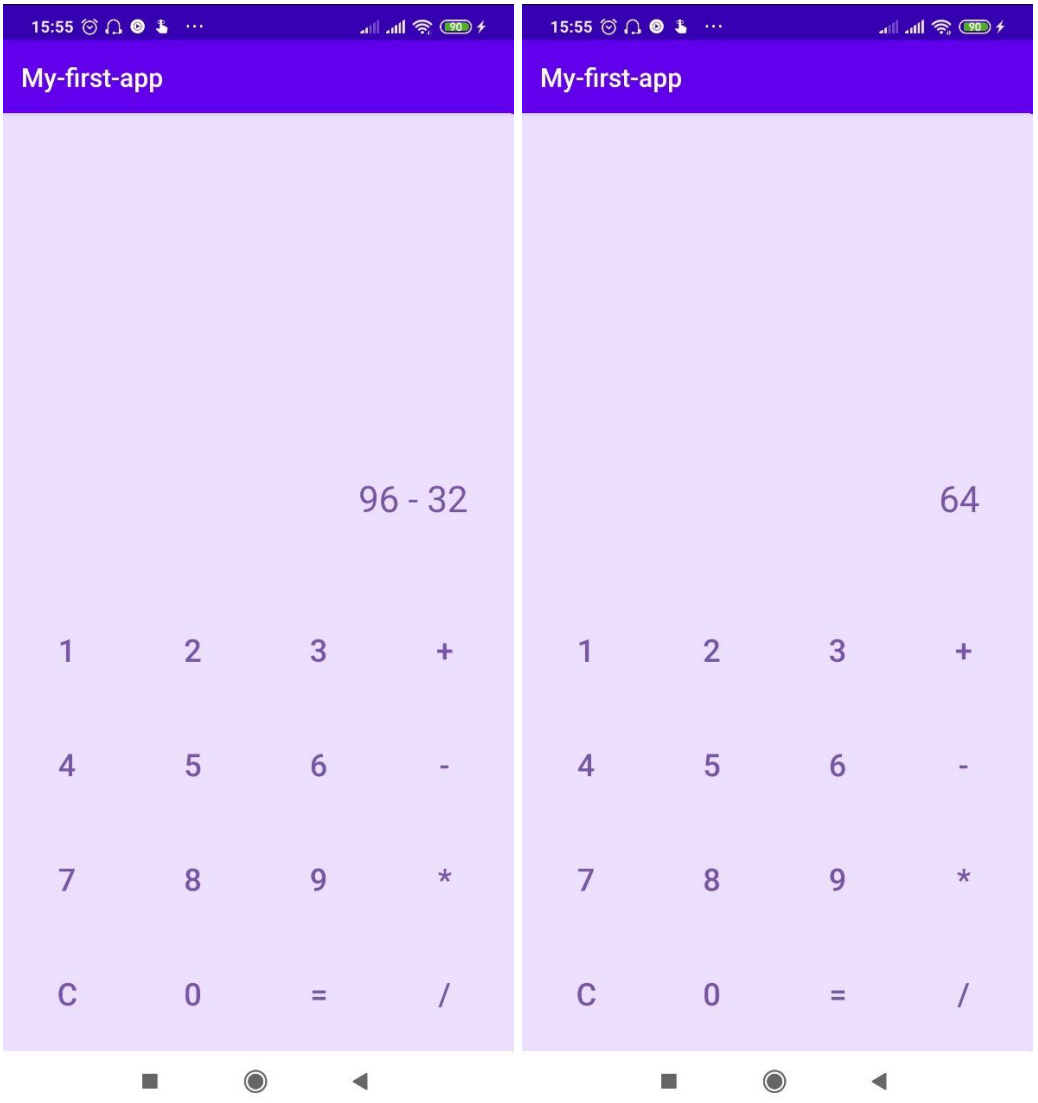

Рис. 2.1, 2.2, Реалізований функціонал калькулятора

MainActivity.java

```
package com.example.first_app;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
```

```
public class MainActivity extends AppCompatActivity {

    private final String operatorPlus = "+";
    private final String operatorMinus = "-";
    private final String operatorMultiply = "*";
    private final String operatorDivide = "/";
    private final String operatorEquals = "=";
    private String operator = "";
    private String firstNumber = "";
    private String secondNumber = "";
    private TextView resultTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.resultTextView = findViewById(R.id.calcValueView);
    }

    private void onIncomeNumber(String num) {
        if (this.operator.isEmpty() && !(this.firstNumber.isEmpty() && num.equals("0"))) {
            this.firstNumber += num;
        } else if (!(this.secondNumber.isEmpty() && num.equals("0"))) {
            this.secondNumber += num;
        }
        this.updateCalcText();
    }

    private void onIncomeOperator(String op) {
        this.operator = op;
        this.updateCalcText();
    }

    private void updateCalcText() {
        String operatorString = "";
        if (this.operator.length() > 0) {
            operatorString = " " + this.operator + " ";
        }
        this.resultTextView.setText(this.firstNumber + operatorString + this.secondNumber);
    }

    public void onClick0(View view) {
        onIncomeNumber(((Button)view).getText().toString());
    }

    public void onClick1(View view) {
        onIncomeNumber(((Button)view).getText().toString());
    }

   …..

    public void onClick9(View view) {
        onIncomeNumber(((Button)view).getText().toString());
    }

    public void onPlusClick(View view) {
```

```
    this.onIncomeOperator(((Button)view).getText().toString());
  }

  public void onMinusClick(View view) {
    this.onIncomeOperator(((Button)view).getText().toString());
  }

  public void onMultClick(View view) {
    this.onIncomeOperator(((Button)view).getText().toString());
  }

  public void onDivideCLick(View view) {
    this.onIncomeOperator(((Button)view).getText().toString());
  }

  public void onEqualsClick(View view) {
    try {
      float firstNum = Float.parseFloat(this.firstNumber);
      float secondNum = Float.parseFloat(this.secondNumber);
      float result = performOperation(firstNum, secondNum, this.operator);
      this.clearText();
      String resultString = String.valueOf(result);
      this.resultTextView.setText(resultString);
      this.firstNumber = resultString;
    } catch (Exception e) {
      // ignore parsing error
    }
  }

  private void clearText() {
    this.firstNumber = "";
    this.secondNumber = "";
    this.operator = "";
    this.updateCalcText();
  }

  public void onClearClick(View view) {
    this.clearText();
  }

  private float performOperation(float a, float b, String operation) {
    switch (operation) {
      case operatorPlus: return (a + b);
      case operatorMinus: return (a - b);
      case operatorDivide: return (a / b);
      case operatorMultiply: return (a * b);
//        default: throw new error, ignore for now
    }
    return 0.0f;
  }


}
```

Кожен з вищенаведених обробників приписуємо до необхідних кнопок через атрибут android:onClick

Завдання 3

Функціонал не змінено, але змінено логіку обробки натискань.

MainActivity.java

```
package com.example.first_app;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private final String OPERATOR_PLUS = "+";
    private final String OPERATOR_MINUS = "-";
    private final String OPERATOR_MULTIPLY = "*";
    private final String OPERATOR_DIVIDE = "/";
    private final String OPERATOR_EQUALS = "=";

    private final double EPSILON = 1e-6;

    private String operator = "";
    private String firstNumber = "";
    private String secondNumber = "";
    private TextView resultTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.resultTextView = findViewById(R.id.calcValueView);
    }

    public void onNumberClick(View view) {
        String btnText = ((Button)view).getText().toString();
        onIncomeNumber(btnText);
    }

    public void onOperatorClick(View view) {
        String btnText = ((Button)view).getText().toString();
        this.onIncomeOperator(btnText);
    }

    private void onIncomeNumber(String num) {
        if (this.operator.isEmpty() && !(this.firstNumber.isEmpty() && num.equals("0"))) {
            this.firstNumber += num;
        } else if (!(this.secondNumber.isEmpty() && num.equals("0"))) {
```

```
            this.secondNumber += num;
        }
        this.updateCalcText();
    }

    private void onIncomeOperator(String op) {
        this.operator = op;
        this.updateCalcText();
    }

    private void updateCalcText() {
        if (this.firstNumber.isEmpty()) {
            this.operator = "";
            this.secondNumber = "";
        }

        String operatorString = "";
        if (this.operator.length() > 0) {
            operatorString = " " + this.operator + " ";
        }
        this.resultTextView.setText(String.format("%s%s%s", this.firstNumber, operatorString,
this.secondNumber));
    }

    public void onEqualsClick(View view) {
        try {
            float firstNum = Float.parseFloat(this.firstNumber);
            float secondNum = Float.parseFloat(this.secondNumber);
            float result = performOperation(firstNum, secondNum, this.operator);
            this.clearText();
            String resultString = String.valueOf(result);
            String trimmedResult =  floatIsLikeInteger(result) ?
                    // had to do such a trick because spliting by "." didn't work -_-
                    resultString.replace(".", ",").split(",")[0] :
                    resultString;

            this.firstNumber = trimmedResult;
            this.resultTextView.setText(trimmedResult);
        } catch (Exception e) {
            // ignore float parsing errors
        }
    }

    private void clearText() {
        this.firstNumber = "";
        this.secondNumber = "";
        this.operator = "";
        this.updateCalcText();
    }

    public void onClearClick(View view) {
        this.clearText();
    }

    private float performOperation(float a, float b, String operation) {
        switch (operation) {
            case OPERATOR_PLUS: return (a + b);
```

```
        case OPERATOR_MINUS: return (a - b);
        case OPERATOR_DIVIDE: return (a / b);
        case OPERATOR_MULTIPLY: return (a * b);
//          default: throw error, ignore for now
      }
      return 0.0f;
   }

   private boolean floatIsLikeInteger(float num) {
      return Math.abs(num - Math.round(num)) < EPSILON;
   }
}
```

Завдання 4

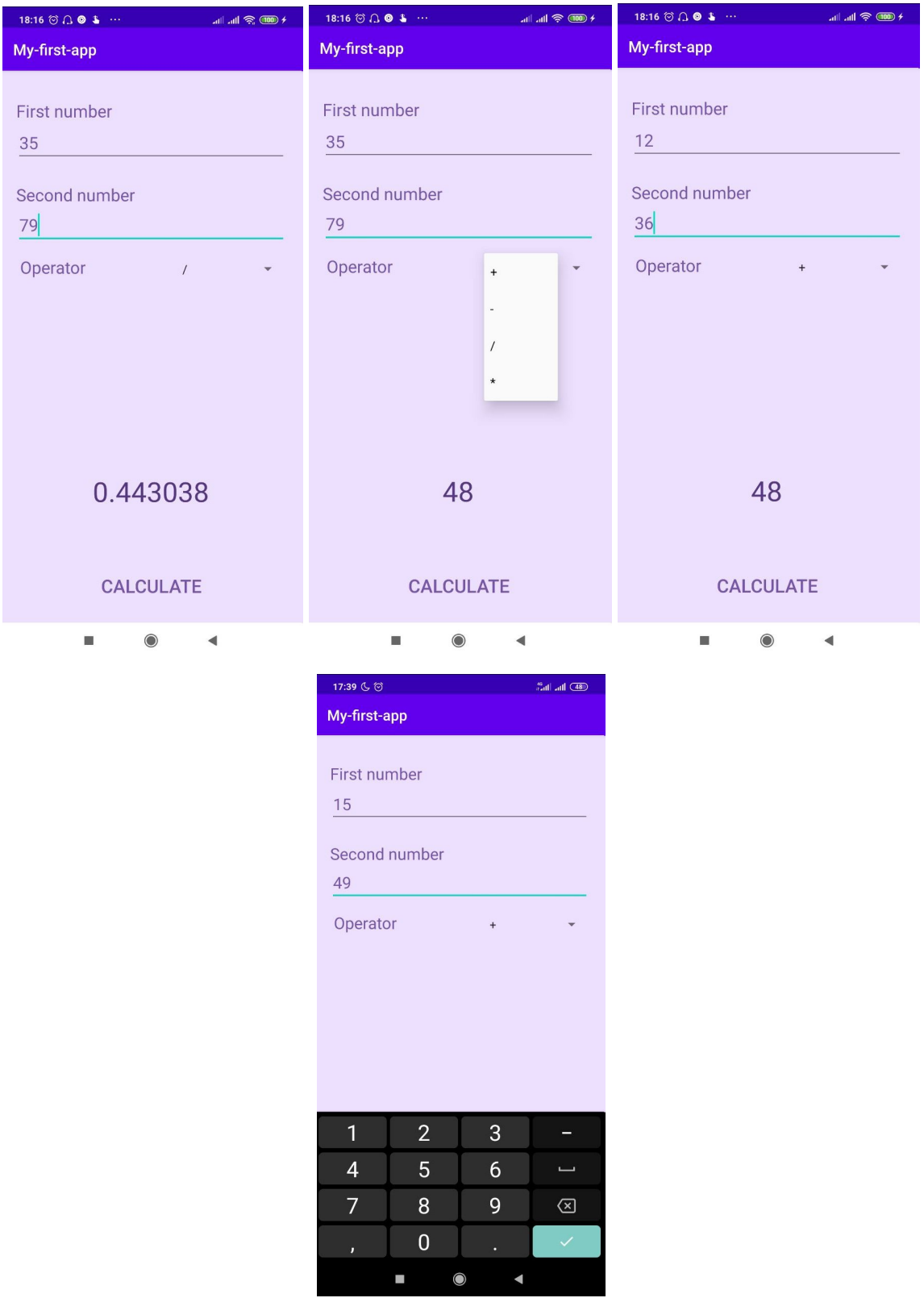

Рис 3.1, 3.2, 3.3, 3.4, Результати роботи нового калькулятора

MainActivity.java

```
package com.example.first_app;

import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import java.lang.reflect.Array;

public class MainActivity extends AppCompatActivity {

   private final String OPERATOR_PLUS = "+";
   private final String OPERATOR_MINUS = "-";
   private final String OPERATOR_MULTIPLY = "*";
   private final String OPERATOR_DIVIDE = "/";
   private final String OPERATOR_EQUALS = "=";

   private final double EPSILON = 1e-6;


   private Spinner spinner;
   private TextView resultText;
   private EditText firstNumInput;
   private EditText secondNumInput;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
//       this.decorateDropdown();
      resultText = (TextView) findViewById(R.id.resultText);
      spinner = (Spinner) findViewById(R.id.operatorSpinner);
      firstNumInput = (EditText) findViewById(R.id.firstNumberInput);
      secondNumInput = (EditText) findViewById(R.id.secondNumberInput);
   }


   public void onCalculateClick(View view) {
      try {
         float firstNumber = Float.parseFloat(firstNumInput.getText().toString());
         float secondNumber = Float.parseFloat(secondNumInput.getText().toString());
         float result = this.performOperation(firstNumber, secondNumber,
spinner.getSelectedItem().toString());
         this.updateResult(result);
      } catch (Exception e) {
         // ignore for now
      }
   }
```

```
    private void updateResult(float res) {
        String resultString = String.valueOf(res);
        String trimmedResult =  floatIsLikeInteger(res) ?
                // had to do such a trick because splitting by "." didn't work -__-
                resultString.replace(".", ",").split(",")[0] :
                resultString;
        this.resultText.setText(trimmedResult);
    }


    private float performOperation(float a, float b, String operation) {
        switch (operation) {
            case OPERATOR_PLUS: return (a + b);
            case OPERATOR_MINUS: return (a - b);
            case OPERATOR_DIVIDE: return (a / b);
            case OPERATOR_MULTIPLY: return (a * b);
//              default: throw error, ignore for now
        }
        return 0.0f;
    }

// useless effort of styling dropdown =(

    private void decorateDropdown() {
//      this.spinner = (Spinner)findViewById(R.id.operatorSpinner);
//      ArrayAdapter operatorsAdapter = ArrayAdapter.createFromResource(this, R.array.operators,
R.layout.spinner_text);
//      operatorsAdapter.setDropDownViewResource(R.layout.spinner_dropdown_item);
//      this.spinner.setAdapter(operatorsAdapter);
    }

    private boolean floatIsLikeInteger(float num) {
        return Math.abs(num - Math.round(num)) < EPSILON;
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/colorPrimaryLight"
    android:paddingLeft="20sp"
    android:paddingRight="20sp"
    android:paddingBottom="0sp"
    tools:context=".MainActivity">


    <EditText
```

```
    android:id="@+id/secondNumberInput"
    style="@style/heading3"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:ems="10"
    android:importantForAutofill="no"
    android:inputType="numberSigned"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.4"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.264" />

  <TextView
    android:id="@+id/secondNumber"
    style="@style/heading3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/second_number_label"
    app:layout_constraintBottom_toTopOf="@+id/secondNumberInput"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="1.0" />

  <EditText
    android:id="@+id/firstNumberInput"
    style="@style/heading3"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:ems="10"
    android:importantForAutofill="no"
    android:inputType="numberSigned"
    app:layout_constraintBottom_toTopOf="@+id/secondNumber"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.6"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.717" />

  <TextView
    android:id="@+id/firstNumber"
    style="@style/heading3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="2dp"
    android:text="@string/first_number_label"
    app:layout_constraintBottom_toTopOf="@+id/firstNumberInput"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.126"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="1.0" />

  <LinearLayout
    android:id="@+id/linearLayout"
```

```
    android:layout_width="match_parent"
    android:layout_height="62dp"
    android:orientation="horizontal"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.434"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/secondNumberInput"
    app:layout_constraintVertical_bias="0.029">

    <TextView
      android:id="@+id/operatorLabel"
      style="@style/heading3"
      android:layout_width="209dp"
      android:layout_height="wrap_content"
      android:paddingLeft="5sp"
      android:text="@string/operator_label"
      tools:text="@string/operator_label" />

    <Spinner
      android:id="@+id/operatorSpinner"
      android:layout_width="match_parent"
      android:layout_height="38dp"
      android:entries="@array/operators" />
  </LinearLayout>

  <Button
    android:id="@+id/button2"
    style="@style/calcButton"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="4dp"
    android:text="@string/calculate_button"
    android:onClick="onCalculateClick"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.4"
    app:layout_constraintStart_toStartOf="parent" />

  <TextView
    android:id="@+id/resultText"
    style="@style/heading1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintBottom_toTopOf="@+id/button2"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout"
    app:layout_constraintVertical_bias="0.802" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Завдання 5

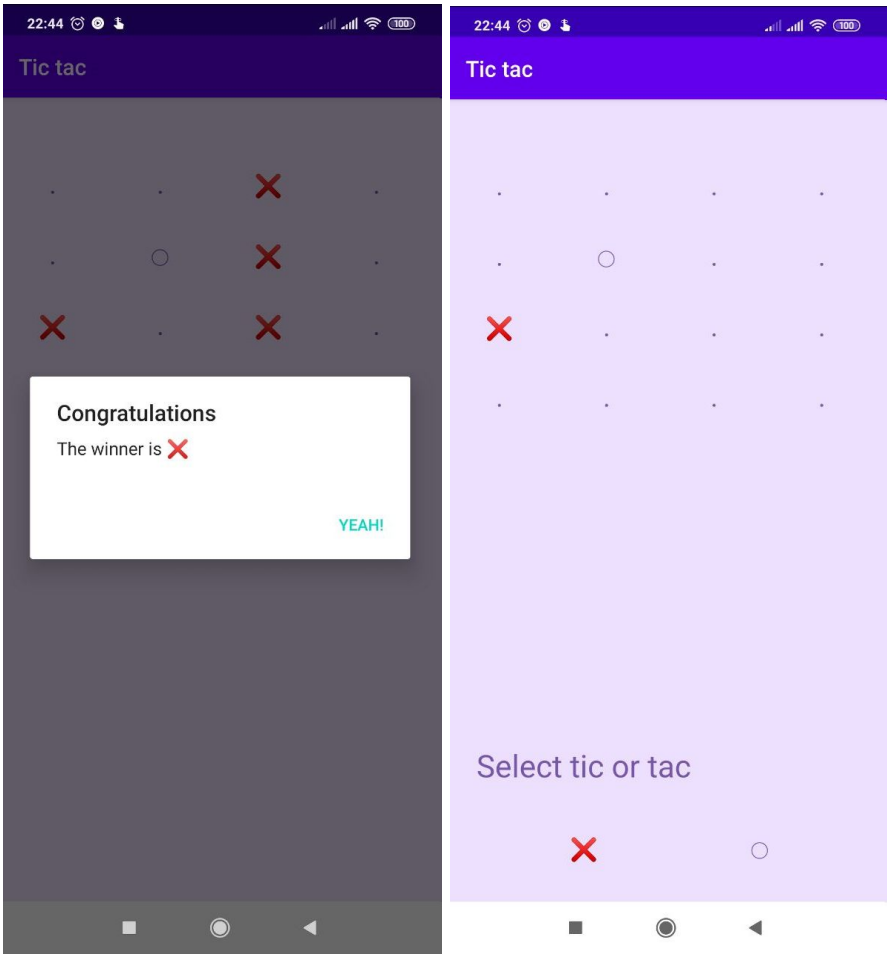

Рис. 4.1, 4.2, Результати роботи гри хрестики-нолики

Реалізація примітивна і проста. Кожен користувач може вибирати брати йому крестик чи нулик, а переможець обирається за звичайними правилами простого варіанту 3 на 3. Алгоритм обрання переможця не залежить від розміру сітки. В планах зробити розмір сітки змінним, весь код, крім розмітки організовано під універсальні розміри сітки

MainActivity.java
package com.example.tic_tak; import androidx.appcompat.app.AlertDialog; import androidx.appcompat.app.AppCompatActivity; import androidx.constraintlayout.widget.ConstraintLayout; import android.content.DialogInterface; import android.os.Bundle; import android.view.View;

```
import android.widget.Button;
import android.widget.TableLayout;
import android.widget.TableRow;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

   private ConstraintLayout selectComponent;
   private ArrayList<ArrayList<Button>> cells;
   private Button selectedCell;
   private String cellPlaceholderText;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      this.initMembers();
      this.attachEventListeners();
      this.hideSelect();
   }

   public void onCellClick(View view) {
      this.showSelect();
   }

   private void attachEventListeners() {
      for (ArrayList<Button> row : cells) {
         for (Button cell : row) {
            cell.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                  Button btn = (Button) v;
                  if (selectedCell != null) {
                     return;
                  }
                  showSelect();
                  disableAllButtonsExceptOne(btn);
                  selectedCell = btn;
               }
            });
         }
      }
   }

   private void resetAllCells() {
      for (ArrayList<Button> row : cells) {
         for (Button btn : row) {
            btn.setEnabled(true);
            btn.setText(cellPlaceholderText);
         }
      }
   }

   private void disableAllButtonsExceptOne(Button b) {
      for (ArrayList<Button> row : cells) {
         for (Button btn : row) {
```

```
            if (btn.getId() != b.getId()) {
                btn.setEnabled(false);
            }
        }
    }
}

private void enableAllButtons() {
    for (ArrayList<Button> row : cells) {
        for (Button btn : row) {
            btn.setEnabled(true);
        }
    }
}

private void initMembers() {
    cellPlaceholderText = getResources().getString(R.string.tic_tac_placeholder);
    selectComponent = (ConstraintLayout) findViewById(R.id.select_component);
    cells = new ArrayList();
    TableLayout container = (TableLayout)findViewById(R.id.tableContainer);
    for (int i = 0; i < container.getChildCount(); i++) {
        TableRow row = (TableRow) container.getChildAt(i);
        ArrayList<Button> cellsRow = new ArrayList<Button>();
        for (int j = 0; j < row.getChildCount(); j++) {
            Button cell = (Button) row.getChildAt(j);
            cellsRow.add(cell);
        }
        this.cells.add(cellsRow);
    }
}

public void onTic(View view) {
    setCellTextOnSideResponse(R.string.tic);
}

public void onTac(View view) {
    setCellTextOnSideResponse(R.string.tac);
}

private void setCellTextOnSideResponse(int stringRecourseId) {
    if (selectedCell != null) {
        selectedCell.setText(stringRecourseId);
        this.selectedCell = null;
    }
    hideSelect();
    enableAllButtons();
    String winner = checkWinner();
    if (winner != null) {
        onWinner(winner);
    }
}

private void onWinner(String winner) {
    String message = getResources().getString(R.string.winner_modal_text) + " " + winner;
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle(R.string.winner_modal_title)
            .setMessage(message)
```

```
        .setPositiveButton(R.string.winner_modal_ok_btn, new DialogInterface.OnClickListener() {
          @Override
          public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
            resetAllCells();
          }
        });
    builder.setCancelable(true);
    builder.create().show();
}

/**
 * @returns `null` if no winner
 */
private String checkWinner() {
    String winner = getWinnerInRows();
    if (winner == null) {
        winner = getWinnerInColumns();
    }
    if (winner == null) {
        winner = getWinnerInDiagonals();
    }

    return winner;
}

private String getWinnerInRows() {
    String winner = null;
    for (int i = 0; i < cells.size(); i++) {
        ArrayList<Button> row = cells.get(i);
        boolean rowIsWinner = buttonsHaveSameText(row);
        if (rowIsWinner) {
            winner = row.get(0).getText().toString();
            if (winner.equals(cellPlaceholderText)) {
                winner = null;
            } else {
                break;
            }
        }
    }

    return winner;
}

private String getWinnerInColumns() {
    String winner = null;
    for (int i = 0; i < cells.size(); i++) {
        ArrayList<Button> col = new ArrayList<Button>();
        for (int j = 0; j < cells.size(); j++) {
            col.add(cells.get(j).get(i));
        }
        boolean rowIsWinner = buttonsHaveSameText(col);
        if (rowIsWinner) {
            winner = col.get(0).getText().toString();
            if (winner.equals(cellPlaceholderText)) {
                winner = null;
            } else {
```

```
                break;
            }
        }
    }

    return winner;
  }

  private String getWinnerInDiagonals() {
    ArrayList<Button> primaryDiagonal = new ArrayList<Button>();
    ArrayList<Button> secondaryDiagonal = new ArrayList<Button>();

    for (int i = 0; i < cells.size(); i++) {
      primaryDiagonal.add(cells.get(i).get(i));
      secondaryDiagonal.add(cells.get(i).get(cells.size() - 1 - i));
    }

    String winner = null;
    if (buttonsHaveSameText(primaryDiagonal)) {
      winner = cells.get(0).get(0).getText().toString();
      if (winner == cellPlaceholderText) {
        winner = null;
      }
    }
    if (winner == null && buttonsHaveSameText(secondaryDiagonal)) {
      winner = cells.get(0).get(cells.size() - 1).getText().toString();
      if (winner == cellPlaceholderText) {
        winner = null;
      }
    }

    return winner;
  }

  private boolean buttonsHaveSameText(ArrayList<Button> btns) {
    String possibleSameText = btns.get(0).getText().toString();
    for (int i = 1; i < btns.size(); i++) {
      String currentText = btns.get(i).getText().toString();
      if (!currentText.equals(possibleSameText)) {
        return false;
      }
    }

    return true;
  }

  private void showSelect() {
    selectComponent.setVisibility(View.VISIBLE);
  }

  private void hideSelect() {
    selectComponent.setVisibility(View.INVISIBLE);
  }
}
```

Усі кнопки та тексти винесені в ресурси

strings.xml

```
<resources>
   <string name="app_name">Tic tac</string>
   <string name="tic">❌</string>
   <string name="tac">◯</string>
   <string name="select_side">Select tic or tac</string>
   <string name="tic_tac_placeholder">.</string>
   <string name="winner_modal_title">Congratulations</string>
   <string name="winner_modal_text">The winner is</string>
   <string name="winner_modal_ok_btn">Yeah!</string>
</resources>
```

Контрольні питання

1.
   a. У моїй роботі використовуються EditText, TextView, LinearLayout, TableLayout, TableRow та Button.
   b. Переваги використання одного обробника є очевидними. По-перше, це зменшення кількості коду, по-друге, спрощення для розробників й по-третє це є best practice для розробки, коли не відбувається дублювання коду

2.

Приклад додавання обробника на подію onClick

```
btn.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      // code here
   }
});
```

Висновки

Я ознайомився ознайомлення із основними принципами та методами створення обробників для елементів управління в Android-застосунках. За цю роботу було створено декілька додатків, кожному з яких у майбутньому можна додати набагато більше можливостей. Лабораторна робота навчила правильно обробляти події елементів у Andoid та будувати більш складні інтерфейси та логіку додатків.