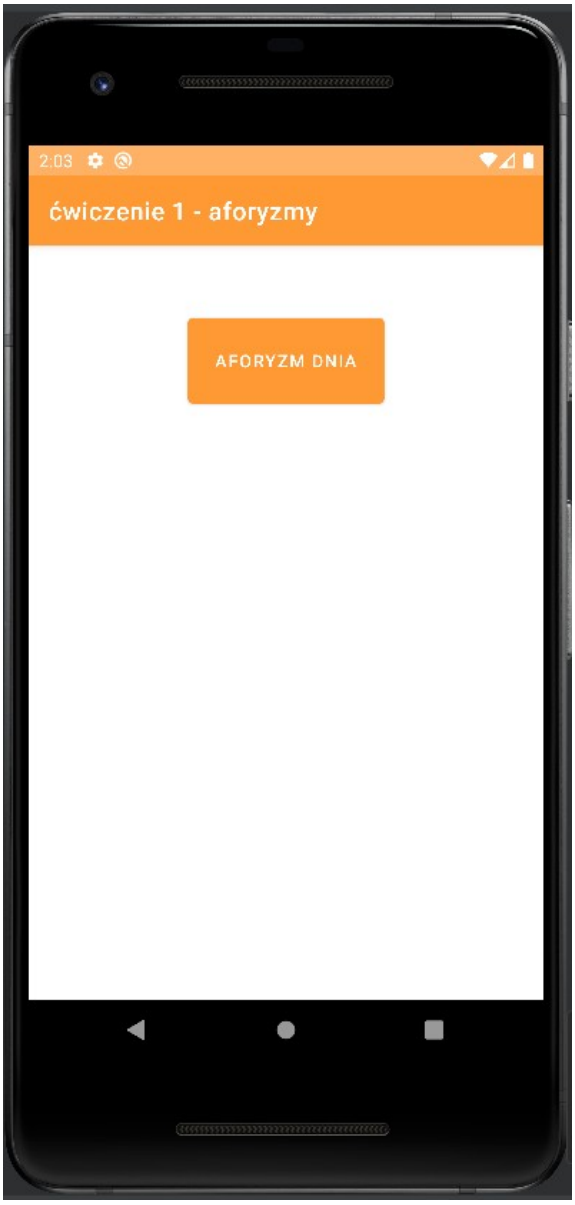
Kurs programowania aplikacji mobilnych
Temat: Aplikacja wyświetlająca aforyzmy oraz aplikacja-kalkulator.

Data: 13/03/2022
Ewa Namysł, Adrianna Pilawa
Informatyka stosowana, III rok

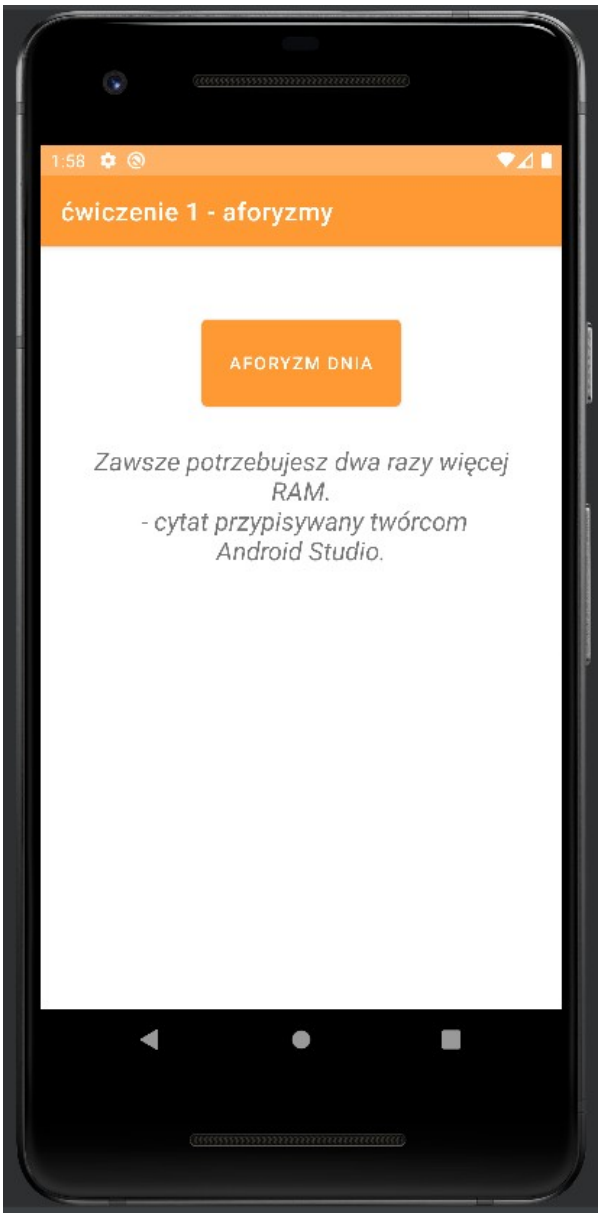
1. Aplikacja wyświetlająca aforyzmy:

Aplikacja po kliknięciu przycisku *Aforyzm dnia* ma za zadanie wyświetlić jeden losowy aforyzm z puli cytatów.

Po włączeniu aplikacji wyświetlony jest ww. guzik. Po kliknięciu przycisku aplikacja wyświetla poniżej wylosowany tekst.



Po włączeniu aplikacji



Po kliknięciu przycisku

Paleta kolorów okna została zmodyfikowana w pliku XML *themes*, a kolory zdefiniowano w *colors.xml*.

```
<color name="orange">#FF9933</color>
<color name="lightorange">#FFB266</color>
```

Fragment pliku colors.xml

```
<item name="colorPrimary">@color/orange</item>
<item name="colorPrimaryVariant">@color/lightorange</item>
<item name="colorOnPrimary">@color/white</item>
```

Fragment pliku themes.xml

Ponadto wyśrodkowano tekst wyświetlany w polu tekstowym, zmieniono jego rozmiar oraz użyto kursywy.

Teksty aforyzmów, napis na przycisku oraz nazwa aplikacji zostały zapisane w pliku *strings.xml*.

```
<?xml version="1.0" encoding="utf-8"?>

<resources>
    <string name="app_name">ćwiczenie 1 - aforyzmy</string>

    <string name="button">Aforyzm dnia</string>

    <string name="cytat1">Ognia nie gasi się ogniem.</string>
    <string name="cytat2">Amicus certus in re incerta cernitur.</string>
    <string name="cytat3">Człowiekiem jestem i nic, co ludzkie, nie jest mi obce.</string>
    <string name="cytat4">Bad art is more tragically beautiful than good art,
                          because it documents human failure.</string>
    <string name="cytat5">If Hell is other people, thought Shadow, then Purgatory is airports.</string>
    <string name="cytat6">If you don\'t know anything about computers,
                          just remember that they are machines that do exactly what you tell them
                          but often surprise you in the result.</string>
    <string name="cytat7">The only legitimate use of a computer is to play games.</string>
    <string name="cytat8">Zawsze potrzebujesz dwa razy więcej RAM.\n -
                          cytat przypisywany twórcom Android Studio.</string>
    <string name="cytat9">The greatest failure is not to try.</string>
    <string name="cytat10">To err is human, but being right is nice too.</string>
    <string name="cytat11">Lex retro not agit.</string>
    <string name="cytat12">If you want to change the way people respond to you,
                           change the way you respond to people.</string>
```

strings.xml

Następnie cytaty zostały przypisane do tablicy w pliku *arrays.xml*.

```
<?xml version="1.0" encoding="utf-8"?>

<resources>
    <string-array name="tab_aforyzmy">
        <item>@string/cytat1</item>
        <item>@string/cytat2</item>
        <item>@string/cytat3</item>
        <item>@string/cytat4</item>
        <item>@string/cytat5</item>
        <item>@string/cytat6</item>
        <item>@string/cytat7</item>
        <item>@string/cytat8</item>
        <item>@string/cytat9</item>
        <item>@string/cytat10</item>
        <item>@string/cytat11</item>
        <item>@string/cytat12</item>
    </string-array>
</resources>
```

arrays.xml

```java
package com.example.myfirstapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    int previous = 0; //poprzedni wylosowany numer

    private TextView tv1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        tv1 = findViewById(R.id.textView1);
    }

    public void sendMessage(View view) {
        java.util.Random gen = new java.util.Random();

        String[] tab = getResources().getStringArray(R.array.tab_aforyzmy);

        int i = gen.nextInt(tab.length);

        if(i == previous){ //jesli wylosowano to samo, co poprzednio
            i = i % 2;
        }
        previous = i;

        String message = tab[i];
        tv1.setText(message); //wyswietl cytat

    }
}
```

MainActivity.java

Metoda *sendMessage* odpowiada za losowanie aforyzmu (używając do tego modułu Random*)* oraz wyświetlenie go w polu tekstowym *tv1*.

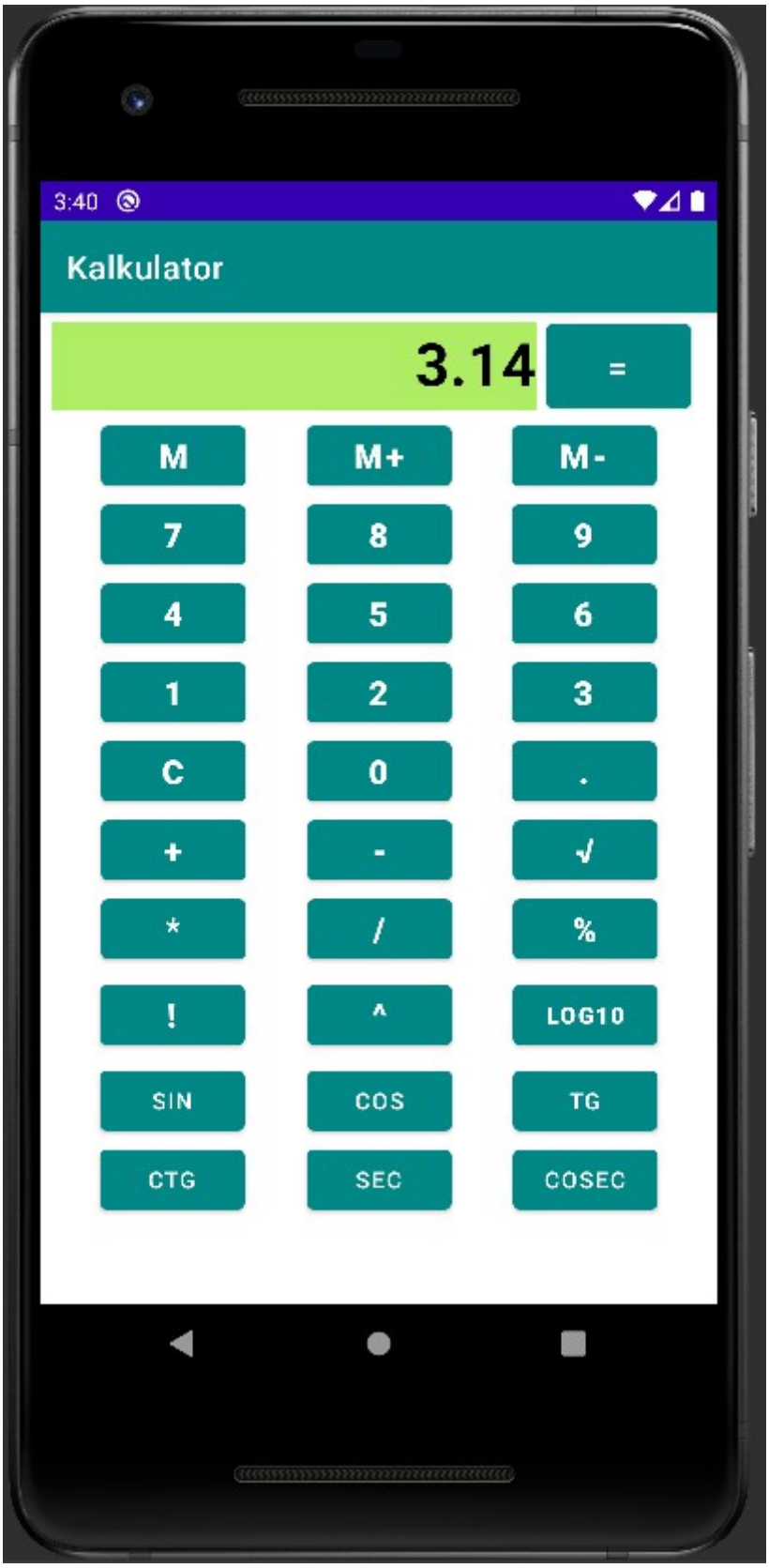
W celu ograniczenia powtórzeń tych samych cytatów pod rząd, zdefiniowano zmienną *previous*, która zachowuje wcześniej wylosowaną liczbę. Jeśli zostanie wylosowana ta sama liczba, co poprzednim razem, wykonywana jest wówczas operacja modulo 2.

Po wylosowaniu liczby, odpowiedni aforyzm z tablicy wyświetlany jest w polu tekstowym.

2. Kalkulator:

Aplikacja posiada podstawowe operacje arytmetyczne, możliwość obliczenia pierwiastka, potęgi, silni, logarytmu, procentu z liczby oraz wartości funkcji trygonometrycznych.

Ponadto zaimplementowano pamięć.



Podobnie jak w poprzedniej aplikacji, dostosowano kolory GUI w *colors.xml* oraz *themes.xml*. W pliku *strings.xml* zdefiniowano napisy wyświetlane na przyciskach kalkulatora.

```
<string name="pamiec_pokaz">M</string>
<string name="pamiec_dodaj">M+</string>
<string name="pamiec_usun">M-</string>
```

Fragment pliku strings.xml

Zdefiniowano zmienne typu String *liczba1_s*, *operacja*, *pamiec* oraz zmienną typu boolean *jest_przecinek*.

```
package com.example.kalkulator;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import static java.lang.Math.*;

public class MainActivity extends AppCompatActivity {
    String liczba1_s;
    String operacja;
    String pamiec;
    boolean jest_przecinek = false;

    private TextView obj;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        obj = findViewById(R.id.textView);
    }
```

Fragment MainActivity.java

W zmiennej *liczba1_s* przechowywana jest liczba wprowadzona przed operacją. Rodzaj operacji zapisywany jest w *operacja*. Jeśli chcemy zachować jakąś liczbę, możemy użyć M+, wówczas zostanie ona zapisana w zmiennej *pamiec*.

Dzięki zmiennej *jest_przecinek* program jest w stanie sprawdzić, czy przy wprowadzaniu liczby pojawił się już przecinek.

Jak wynika z nazwy metody *aktualizuj_pole*, metoda ta pozwala na kontrolowanie znaków wyświetlanych w polu tekstowym *obj*.
Metoda ta sprawdza co zostało do niej przesłane w stringu *new_text* i postępuje w zależności od tego, czy przesłany został przecinek, operacja, cyfra lub też wybrano czyszczenie pola tekstowego.

```
public void aktualizuj_pole(String new_text){
    //aktualizuje pole tekstowe kalkulatora
    CharSequence old_text;
    String text;

    if (new_text.equals("")){
        //czyszczenie ekranu (C)
        obj.setText(new_text);
    }
    else if(new_text.equals(".")){
        //przecinek
        if(!jest_przecinek){
            old_text = obj.getText();

            if(old_text.equals("")){
                old_text = "0";  //najpierw wprowadzono kropke, dodaj zero przed przecinkiem
            }
            jest_przecinek = true;

            text = old_text + "."; //zaktualizuj o kropke
            obj.setText(text);
        }
    }
    else{
        //wprowadzono operacje lub cyfre

        if(!isNumeric(new_text)){
            //wprowadzono operacje
            liczba1_s = obj.getText().toString(); //zapisz liczbe
            operacja = new_text; //zapisz operacje
            obj.setText(new_text); //wyswietl operacje
            jest_przecinek = false; //zresetuj przecinek
        }
        else{
            //wprowadzono cyfre
            old_text = obj.getText();
            String old_text_s = old_text.toString();

            if(!isNumeric(old_text_s)){
                //poprzednia byla operacja
                obj.setText(new_text); //wyswietl nowa cyfre
            }

            else {
                //poprzednia byla cyfra
                text = old_text + new_text; //dopisz wprowadzona cyfre
                obj.setText(text);
            }
        }
    }
}
```

Metoda aktualizuj_pole

Aktualizuj_pole korzysta również z metody *isNumeric*, która zwraca prawdę, jeśli string ma wartość liczbową, dzięki czemu można sprawdzić czy wprowadzono operację czy liczbę.

```
public static boolean isNumeric(String str) {
    //sprawdza czy podany string jest liczba
    try {
        Double.parseDouble(str);
        return true;
    } catch(NumberFormatException e){
        return false;
    }
}
```

Metoda isNumeric

Metoda *aktualizuj_pole* wywoływana jest przez metody połączone z odpowiednimi przyciskami.

```
public void znak_sec(View view){
    aktualizuj_pole( new_text: "sec");
}

public void znak_cosec(View view){
    aktualizuj_pole( new_text: "cosec");
}

public void znak_mem_plus(View view){
    //dodaje wartosc z ekranu do pamieci
    pamiec = obj.getText().toString();
}

public void znak_mem_minus(View view){
    //usuwa zapisana liczbe z pamieci
    pamiec = "";
}

public void znak_mem(View view){
    //umieszcza liczbe z pamieci w polu tekstowym
    obj.setText(pamiec);
}

public void usun(View view){
    aktualizuj_pole( new_text: "");
    jest_przecinek = false;
}
```

Niektóre z metod wykorzystujących aktualizuj_pole

Aby wykonać działanie należy kliknąć przycisk ze znakiem równości. Wywoływana jest wówczas metoda *rowna_sie*. Sprawdzany jest wówczas rodzaj wprowadzanej operacji.

```
public void rowna_sie(View view){
    //wykonuje dzialania
    double wynik = 0;

    try {
        double liczba1 = Double.parseDouble(liczba1_s);

        if (operacja.equals("+") || operacja.equals("-") || operacja.equals("*") ||
            operacja.equals("/") || operacja.equals("^") || operacja.equals("%")) {
            //operacje dwuliczbowe -> [liczba1] [operacja] [liczba2] =

            //liczba wprowadzona po operacji:
            double liczba2 = Double.parseDouble(obj.getText().toString());

            switch (operacja) {
                case "+":
                    wynik = liczba1 + liczba2;
                    break;
                case "-":
                    wynik = liczba1 - liczba2;
                    break;
                case "*":
                    wynik = liczba1 * liczba2;
                    break;
                case "/":
                    wynik = liczba1 / liczba2;
                    break;
                case "^":
                    wynik = pow(liczba1, liczba2);
                    break;
                case "%":
                    //[liczba1] % z [liczba2]
                    wynik = (liczba1/100) * liczba2;
                    break;
            }
        }
```

Metoda rowna_sie - operacje wymagające dwóch liczb

```
        else {
            //operacje jednoliczbowe -> [liczba1] [operacja] =
            switch (operacja) {
                case "√":
                    wynik = sqrt(liczba1);
                    break;
                case "!":
                    int silnia = 1;
                    for (int i = 1; i <= liczba1; i++) {
                        silnia = silnia * i;
                    }
                    wynik = silnia;
                    break;
                case "log":
                    wynik = log10(liczba1);
                    break;
                //funkcje trygonometryczne: liczba1 w radianach
                case "sin":
                    wynik = sin(liczba1);
                    break;
                case "cos":
                    wynik = cos(liczba1);
                    break;
                case "tg":
                    wynik = tan(liczba1);
                    break;
                case "ctg":
                    wynik = 1 / tan(liczba1);
                    break;
                case "sec":
                    wynik = 1 / cos(liczba1);
                    break;
                case "cosec":
                    wynik = 1 / sin(liczba1);
                    break;
            }
        }
        obj.setText(String.valueOf(wynik)); //wyswietl wynik
    }catch(Exception e){
        e.printStackTrace();
    }
}
```

Metoda rowna_sie – operacje, które wymagają podania jednej liczby