# 5COSC023W - Tutorial 3 Exercises Sample Solutions

## Exercises 1–2

```
package uk.ac.westminster.lotterycomposableextendedfullapp

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.width
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.setValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import kotlin.random.Random

class MainActivity : ComponentActivity() {
    var number_of_clicks = 0

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            displayNumbers()
        }
    }

    @Composable
    fun displayNumbers() {
        // make the results list reference and number1-number6 states
        // of the composable
        var results by remember { mutableStateOf(mutableListOf<Int>(0, 0, 0, 0, 0, 0)) }
```

```
var number1 by remember { mutableStateOf(0) }
var number2 by remember { mutableStateOf(0) }
var number3 by remember { mutableStateOf(0) }
var number4 by remember { mutableStateOf(0) }
var number5 by remember { mutableStateOf(0) }
var number6 by remember { mutableStateOf(0) }

Column(
    Modifier.fillMaxSize(),
    horizontalAlignment = Alignment.CenterHorizontally,
    verticalArrangement = Arrangement.Center
) {
    Row {
        Text("Results: ", fontSize = 30.sp)
        Text("" + number1, fontSize = 30.sp)
        Spacer(Modifier.width(10.dp))
        Text("" + number2, fontSize = 30.sp)
        Spacer(Modifier.width(10.dp))
        Text("" + number3, fontSize = 30.sp)
        Spacer(Modifier.width(10.dp))
        Text("" + number4, fontSize = 30.sp)
        Spacer(Modifier.width(10.dp))
        Text("" + number5, fontSize = 30.sp)
        Spacer(Modifier.width(10.dp))
        Text("$number6", fontSize = 30.sp)  // alternative way
    }
    Spacer(Modifier.height(30.dp))
    Row {
        Button(onClick = { results = calculate() }) {
            Text(text = "Generate", fontSize = 24.sp)
        }

        Spacer(Modifier.width(20.dp))

        Button(onClick = { results = sortResults(results) }) {
            Text(text = "Sort", fontSize = 24.sp)
        }
    }

    // buttons for changing individual numbers
    Row {
        Button(onClick = {
            var new_number = 1 + Random.nextInt(59)
            while (new_number in results) {
                new_number = 1 + Random.nextInt(59)
            }
            results[0] = new_number
            number1 = new_number
        }
```

```
) {
    Text(text = "1", fontSize = 24.sp)
}

Button(onClick = {
    var new_number = 1 + Random.nextInt(59)
    while (new_number in results) {
        new_number = 1 + Random.nextInt(59)
    }
    results[1] = new_number
    number2 = new_number
}
) {
    Text(text = "2", fontSize = 24.sp)
}

Button(onClick = {
    var new_number = 1 + Random.nextInt(59)
    while (new_number in results) {
        new_number = 1 + Random.nextInt(59)
    }
    results[2] = new_number
    number3 = new_number
}
) {
    Text(text = "3", fontSize = 24.sp)
}

Button(onClick = {
    var new_number = 1 + Random.nextInt(59)
    while (new_number in results) {
        new_number = 1 + Random.nextInt(59)
    }
    results[3] = new_number
    number4 = new_number
}
) {
    Text(text = "4", fontSize = 24.sp)
}

Button(onClick = {
    var new_number = 1 + Random.nextInt(59)
    while (new_number in results) {
        new_number = 1 + Random.nextInt(59)
    }
    results[4] = new_number
    number5 = new_number
}
) {
    Text(text = "5", fontSize = 24.sp)
```

```
            }

            Button(onClick = {
                var new_number = 1 + Random.nextInt(59)
                while (new_number in results) {
                    new_number = 1 + Random.nextInt(59)
                }
                results[5] = new_number
                number6 = new_number
            }
            ) {
                Text(text = "6", fontSize = 24.sp)
            }
        }


    }

    // modify the individual numbers displayed based on the values from the list
    number1 = results[0]
    number2 = results[1]
    number3 = results[2]
    number4 = results[3]
    number5 = results[4]
    number6 = results[5]
}

/* Sort the numbers in ascending or descending order based
   on whether the number of clicks is even or odd
 */
fun sortResults(results: MutableList<Int>): MutableList<Int> {
    ++number_of_clicks

    var new_results = results.toMutableList()
    if (number_of_clicks % 2 == 1)
        new_results.sort()
    else {
        new_results.sort()
        new_results.reverse()
    }

    return new_results
}


// do the actual calculation of new numbers
fun calculate(): MutableList<Int> {
    /* note the creation of the new list as in this case
       the state of the displayNumbers() will not change otherwise */
    val numbers = mutableListOf<Int>()
```

```
        while (numbers.size < 6) {
            val new_number = 1 + Random.nextInt(59)
            if (new_number !in numbers)
                numbers.add(new_number)
        }

        return numbers
    }
}
```