**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

**Лабораторна робота №6**

з дисципліни  
«Об’єктно-орієнтоване програмування»

Виконав: Перевірив:

Студент групи ІМ-22 Порєв В.М  
Тимофеєв Даниіл Костянтинович  
номер у списку групи: 23

Київ 2023

**Мета**: Мета роботи: отримати вміння та навички використовувати засоби

обміну інформацією та запрограмувати взаємодію незалежно працюючих

програмних компонентів.

**Завдання :**

1. Створити у середовищі MS Visual Studio C++ проект Win32 з ім’ям

Lab6.

2. Написати вихідні тексти усіх програм-компонентів згідно варіанту

завдання.

3. Скомпілювати вихідні тексти і отримати виконувані файли програм.

4. Перевірити роботу програми. Налагодити програму.

5. Проаналізувати та прокоментувати результати та вихідний текст

програми.

6. Оформити звіт.

**Варіанти** :

1. Для початку роботи користувач програми вибирає потрібний пункт меню програми Lab6. Далі з’являється вікно діалогу, у якому потрібно ввести параметри згідно варіанту завдання. У вікні діалогу користувач натискує кнопку "Так" (або "Виконати") і на цьому місія користувача закінчується – далі він тільки спостерігає, як програма сама автоматично виконає усе, що потрібно для отримання результату. Виклик інших програм – Object2 та Object3 головна програма Lab6 повинна робити без участі користувача.

2. Обмін повідомленнями та масивами даних між програмами Lab6, Object2 та Object3 повинен відбуватися автоматично, без участі користувача. Програма Lab6 повинна автоматично у певній послідовності знаходити та викликати програми Object2 та Object3.

3. У результаті одного сеансу роботи користувач повинен бачити головні вікна програм Object2 та Object3, у яких відображатимуться потрібні результати відповідно варіанту завдань. Для цього вікна програм повинні автоматично розташуватися так, щоб усі результати було видно. Програма Lab6 повинна залишатися у активному стані, щоб користувач мав можливість повторно виконати роботу.

4. Передбачити варіанти успішної роботи у випадках, коли програми Object2 та Object3 (одна або обидві) до того вже були викликані.

5. По завершенні роботи програми Lab6 повинні автоматично завершуватися і програми Object2 та Object3.

Номер варіанту = Ж mod 4,

де: Ж – номер в списку студентів в журналі.

Номер варіанту: 0

**Програма Lab6 :**

1. Користувач вводить значення параметрів nPoint, xMin, xMax, yMin, yMax у вікні діалогу.

2. Програма викликає програми Object2, 3 і забезпечує обмін повідомленнями для передавання та отримання інформації.

**Програма Object2 :**

1. Створює nPoint пар цілих (int) значень (x,y) в діапазонах xMin – xMax, yMin – yMax.

2. Показує числові значення у власному головному вікні.

3. Записує дані в Clipboard Windows у текстовому форматі.

**Програма Object3 :**

1. Зчитує дані з Clipboard Windows

2. Відображає графік y=f(x) у власному головному вікні. Графік, як в математиці – лінія, що проходить через точки (x,y) в порядку зростання x; осі координат з підписами числових значень x, y.

**Вихідний текст програм** :

**Lab6.kt**

class MainActivity : AppCompatActivity() {

private lateinit var pointsNumber: EditText

private lateinit var minX: EditText

private lateinit var maxX: EditText

private lateinit var performButton: Button

private lateinit var minY: EditText

private lateinit var maxY: EditText

override fun onCreate (savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

pointsNumber = findViewById(R.id.number\_text)

minX = findViewById(R.id.xMin\_text)

maxX = findViewById(R.id.xMax\_text)

performButton = findViewById(R.id.perform\_button)

minY = findViewById(R.id.yMin\_text)

maxY = findViewById(R.id.yMax\_text)

performButton.setOnClickListener {

launchObject2Intent()

}

}

private fun launchObject2Intent () {

val object2Intent: Intent? = packageManager.getLaunchIntentForPackage("com.example.object2")

object2Intent?.apply {

addFlags(Intent.FLAG\_ACTIVITY\_SINGLE\_TOP)

addFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK)

addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TASK)

putExtra("n", pointsNumber.text.toString().toIntOrNull() ?: 0)

putExtra("minX", minX.text.toString().toIntOrNull() ?: 0)

putExtra("maxX", maxX.text.toString().toIntOrNull() ?: 0)

putExtra("minY", minY.text.toString().toIntOrNull() ?: 0)

putExtra("maxY", maxY.text.toString().toIntOrNull() ?: 0)

startActivity(this)

}

}

}

**Object2.kt**

class MainActivity : AppCompatActivity() {

override fun onCreate (savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

val nPoint = intent.getIntExtra("n", 0)

val xMin = intent.getIntExtra("minX", 0)

val xMax = intent.getIntExtra("maxX", 0)

val yMin = intent.getIntExtra("minY", 0)

val yMax = intent.getIntExtra("maxY", 0)

val points = generatePoints(nPoint, xMin, xMax, yMin, yMax)

val sortedPoints = sortPoints(points)

displayPoints(sortedPoints)

copyToClipboard(sortedPoints, this)

val obj3Intent = packageManager.getLaunchIntentForPackage("com.example.object3")

obj3Intent?.addFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK)

startActivity(obj3Intent)

}

override fun onNewIntent (intent: Intent?) {

super.onNewIntent(intent)

setIntent(intent)

val obj3Intent: Intent? = packageManager.getLaunchIntentForPackage("com.example.object3")

obj3Intent!!.addFlags(Intent.FLAG\_ACTIVITY\_SINGLE\_TOP)

startActivity(obj3Intent)

}

private fun generatePoints (n: Int, xMin: Int, xMax: Int, yMin: Int, yMax: Int): List<Pair<Int, Int>> {

return List(n) {

Pair(Random.nextInt(xMin, xMax), Random.nextInt(yMin, yMax))

}

}

private fun sortPoints (points: List<Pair<Int, Int>>): List<Pair<Int, Int>> {

return points.sortedWith(compareBy({ it.first }, { it.second }))

}

private fun displayPoints (points: List<Pair<Int, Int>>) {

val textView: TextView = findViewById(R.id.textView)

var string = "X\t\t\tY\n"

for (point in points) {

string += "${point.first}\t\t\t${point.second}\n"

}

textView.text = string.trim('\n')

}

private fun copyToClipboard (points: List<Pair<Int, Int>>, context: Context) {

val clipboard = context.getSystemService(CLIPBOARD\_SERVICE) as ClipboardManager

val textToCopy = buildString {

append("X\t\t\tY\n")

append(points.joinToString("\n") { "${it.first}\t\t\t${it.second}" })

}

val clip = ClipData.newPlainText("coords", textToCopy)

clipboard.setPrimaryClip(clip)

}

}

**Object3.kt**

class MainActivity : AppCompatActivity() {

private lateinit var lineGraphView: GraphView

private lateinit var clipboardData: String

private lateinit var xAxisBounds: Pair<Double, Double>

private lateinit var yAxisBounds: Pair<Double, Double>

override fun onCreate (savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

}

override fun onWindowFocusChanged (hasFocus: Boolean) {

super.onWindowFocusChanged(hasFocus)

if (hasFocus) {

initializeGraphView()

processClipboardData()

}

}

private fun initializeGraphView () {

lineGraphView = findViewById(R.id.idGraphView)

lineGraphView.removeAllSeries()

lineGraphView.viewport.isScrollable = true

lineGraphView.viewport.isScalable = true

lineGraphView.viewport.setScalableY(false)

lineGraphView.viewport.setScrollableY(false)

}

private fun processClipboardData () {

val clipboard = getSystemService(CLIPBOARD\_SERVICE) as ClipboardManager

val clipboardText = clipboard.primaryClip?.getItemAt(0)?.text?.toString()

clipboardData = clipboardText.toString()

val graphSeries = createSeriesFromClipboardData(clipboardData)

graphSeries.isDrawDataPoints = true

graphSeries.dataPointsRadius = 10f

graphSeries.color = Color.GREEN

lineGraphView.animate()

setGraphViewMinMax(graphSeries)

lineGraphView.addSeries(graphSeries)

}

private fun createSeriesFromClipboardData (str: String): LineGraphSeries<DataPoint> {

val dataPointStrings = str.trimEnd('\n').split("\n").toMutableList().also { it.removeAt(0) }

val dataPoints = mutableListOf<DataPoint>()

for (i in dataPointStrings.indices) {

val (x, y) = dataPointStrings[i].split("\t\t\t").map { it.toDouble() }

dataPoints.add(DataPoint(x, y))

updateMinMaxValues(x, y, i, dataPointStrings.size)

}

return LineGraphSeries(dataPoints.toTypedArray())

}

private fun setGraphViewMinMax (series: LineGraphSeries<DataPoint>) {

lineGraphView.viewport.setMinX(xAxisBounds.first)

lineGraphView.viewport.setMaxX(xAxisBounds.second)

lineGraphView.viewport.setMinY(yAxisBounds.first)

lineGraphView.viewport.setMaxY(yAxisBounds.second)

}

private fun updateMinMaxValues (x: Double, y: Double, index: Int, size: Int) {

if (index == 0) {

xAxisBounds = Pair(x, x)

yAxisBounds = Pair(y, y)

} else if (index == size - 1) {

xAxisBounds = xAxisBounds.copy(second = x)

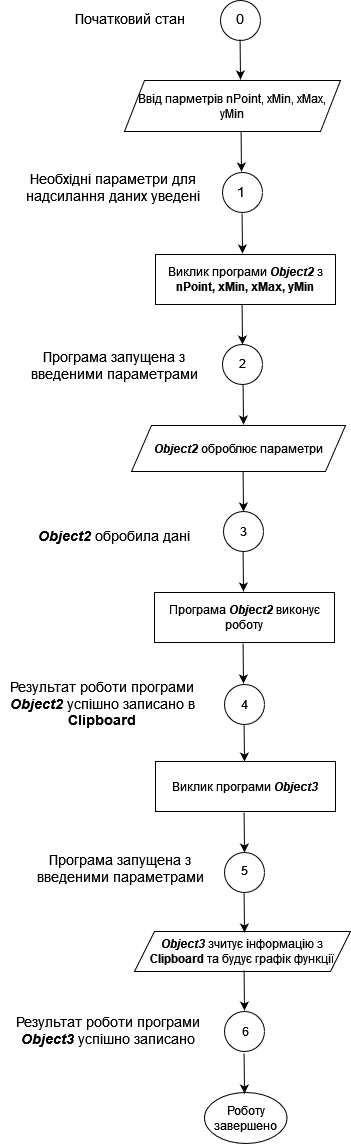
}

yAxisBounds = Pair(yAxisBounds.first.coerceAtMost(y), yAxisBounds.second.coerceAtLeast(y))

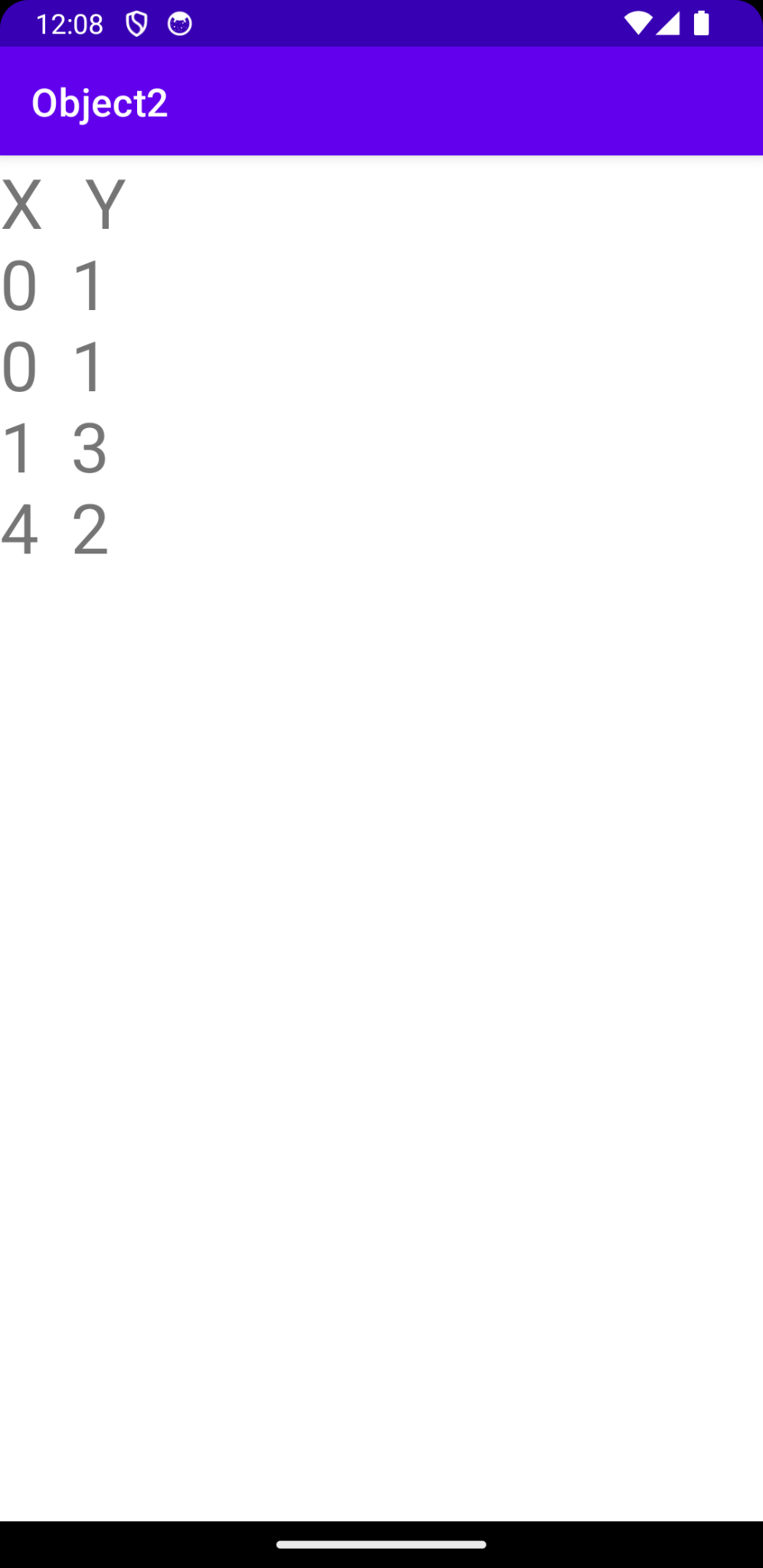
}

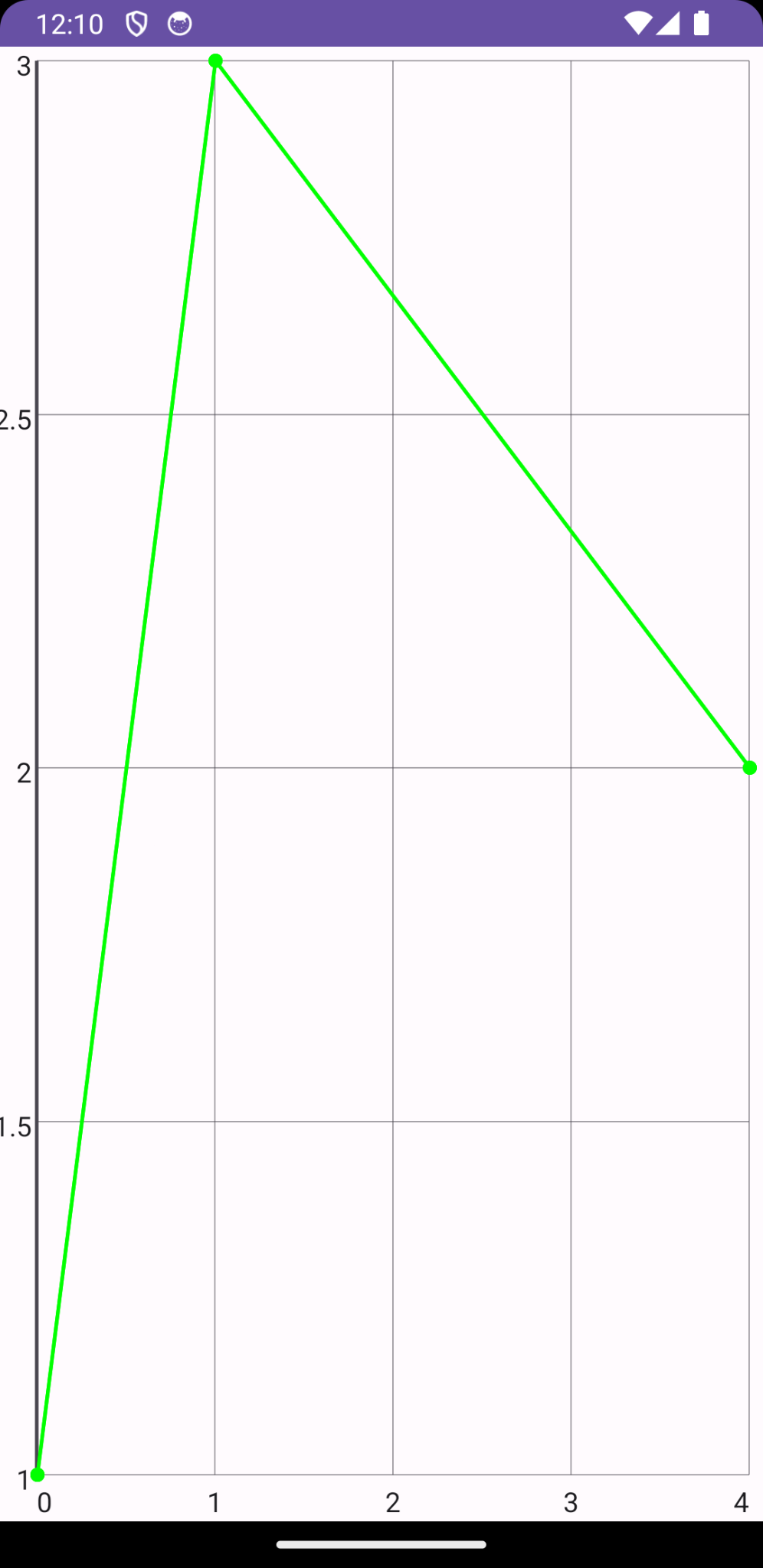
}

**Діаграма класів :**



**Тестування програми (скріншоти)**





**Висновки:** В ході виконання лабораторної роботи № 6 було розроблено взаємодію незалежних програмних компонентів .

У проекті Lab6 і його компонентах Object2 та Object3 використовується механізм передачі даних та обміну повідомленнями між додатками. Основний функціонал представлений так: **Lab6** створює дані (кількість точок, межі значень X та Y). Запускає Object2 через Intent, передаючи ці дані. Обробляє введені користувачем дані та викликає Object2 для подальшої обробки. **Object2**: отримує дані від Lab6 через Intent (кількість точок, межі значень X та Y). Генерує точки на основі отриманих даних. Сортує та відображає точки у вигляді тексту. Копіює відсортовані точки до буфера обміну для подальшого використання. Викликає Object3 для побудови графіка точок. **Object3**: Ініціалізує GraphView для відображення графіка. Отримує дані з буфера обміну, які були згенеровані та відсортовані Object2. Відображає графік точок на GraphView.

Object2 викликає Object3 для побудови графіка, передаючи дані через Intent. Обмін даними між Object2 та Object3 відбувається за допомогою буфера обміну, коли Object2 копіює дані, а Object3 їх отримує. У взаємодії між цими компонентами акцент робиться на передачі та обробці даних, а також на обміні повідомленнями для побудови та відображення графіка точок.