КУРСОВая Работа

Создание игры «Tetris» под Android на языке   
Kotlin с использованием функционального   
подхода к программированию

Выполнил

студент гр.3530904/80004 < > Мащенко Б.Б

Руководитель <> Лукашин А.А

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

# Содержание

[Содержание 2](#_Toc27521899)

[Цель 3](#_Toc27521900)

[Структура программы 4](#_Toc27521901)

[Класс NextPiece 4](#_Toc27521902)

[Класс Piece 5](#_Toc27521903)

[Класс PieceI 6](#_Toc27521904)

[Класс PieceJ 7](#_Toc27521905)

[Класс PieceL 8](#_Toc27521906)

[Класс PieceO 9](#_Toc27521907)

[Класс PieceS 10](#_Toc27521908)

[Класс PieceT 11](#_Toc27521909)

[Класс PieceZ 12](#_Toc27521910)

[Класс CubeAdapter 13](#_Toc27521911)

[Класс NextBlockAdapter 14](#_Toc27521912)

[Скриншоты программы 15](#_Toc27521913)

[Заключение 17](#_Toc27521914)

# **Цель**

1. Разработать программу «Tetris» на языке Kotlin под Android с использованием знаний о функциональном программировании, полученных за семестр.

# **Структура программы**

Вся программа состоит из 13 файлов с исходным кодом и файлов-ресурсов, нужных для отрисовки и представления программы. Рассмотрим используемые в программе классы.

## Класс NextPiece

* constructor(nextPiece: ArrayList<Piece>, block: Int);

+ constructor(axe: Int, block: String);

NextPiece

## Класс Piece

Piece

+ axe: Int  
+ block: String  
+ rotation: Int  
+ cube1: Int  
+ cube2: Int  
+ cube3: Int  
+ cube4: Int  
+ cube5: Int  
+ cube6: Int  
+ cube7: Int  
+ cube8: Int

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ moveDown(pieceList: ArrayList<Piece>)  
  
+ moveRight(pieceList: ArrayList<Piece>)  
  
+ moveLeft(pieceList: ArrayList<Piece>)  
  
+ checkRight(pieceList: ArrayList<Piece>) : Boolean  
  
+ checkLeft(pieceList: ArrayList<Piece>) : Boolean  
  
+ checkDown(pieceList: ArrayList<Piece>) : Boolean  
  
+ detectWall(pieceList: ArrayList<Piece>) : Boolean   
  
+ removeBlock(pieceList: ArrayList<Piece>)

+ detectBottom(pieceList: ArrayList<Piece>) : Boolean

## Класс PieceI

PieceI

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceJ

PieceJ

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceL

PieceL

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceO

PieceO

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceS

PieceS

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceT

PieceT

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс PieceZ

PieceT

+ block: String

+ constructor(axe: Int)

+ constructor(pieceList: ArrayList<Piece>, axe: Int)

+ rotation(pieceList: ArrayList<Piece>) : Boolean

+ detectWall(pieceList: ArrayList<Piece>) : Boolean

+ checkRight(pieceList: ArrayList<Piece>) : Boolean

+ checkLeft(pieceList: ArrayList<Piece>) : Boolean

+ checkDown(pieceList: ArrayList<Piece>) : Boolean

+ moveRight(pieceList: ArrayList<Piece>)

+ moveLeft(pieceList: ArrayList<Piece>)

+ moveDown(pieceList: ArrayList<Piece>)

## Класс CubeAdapter

CubeAdapter

- mInflater: LayoutInflater  
- mBlock: String  
- mAxe: Int  
- items: ArrayList<Piece>  
- test: Int

+ constructor(Context, ArrayList<Piece>)

+ getView(position: Int, convertView: View?, parent: ViewGroup): View  
  
+ getCount(): Int

+ getItemId(position: Int): Long

+ getItem(position: Int): Any?

- square(position: Int, view: View)

- zi(position: Int, view: View)

- el(position: Int, view: View)

- ji(position: Int, view: View)

- es(position: Int, view: View)

## Класс NextBlockAdapter

+ getView(position: Int, convertView: View?, parent: ViewGroup): View

+ getCount(): Int

+ getItemId(position: Int): Long

+ getItem(position: Int): Any?

- square(position: Int, view: View)

- zi(position: Int, view: View)

- ti(position: Int, view: View)

- line(position: Int, view: View)

- el(position: Int, view: View)

- ji(position: Int, view: View)

- es(position: Int, view: View)

- mInflater: LayoutInflater

- mBlock: String  
- mAxe: Int  
- items: ArrayList<Piece>   
- test: Int

NextBlockAdapter

# **Скриншоты программы**

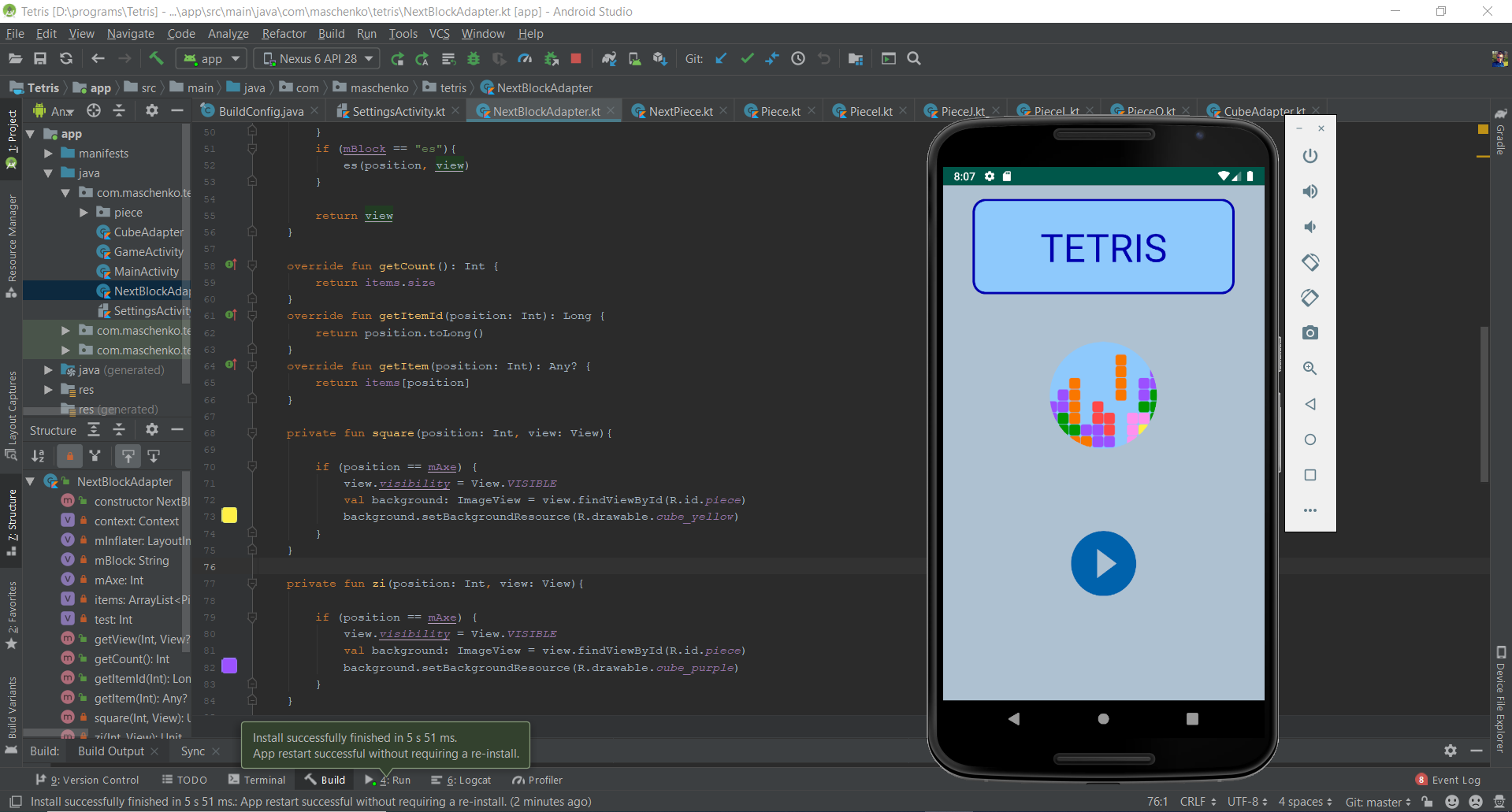


рис 1. Запуск приложения

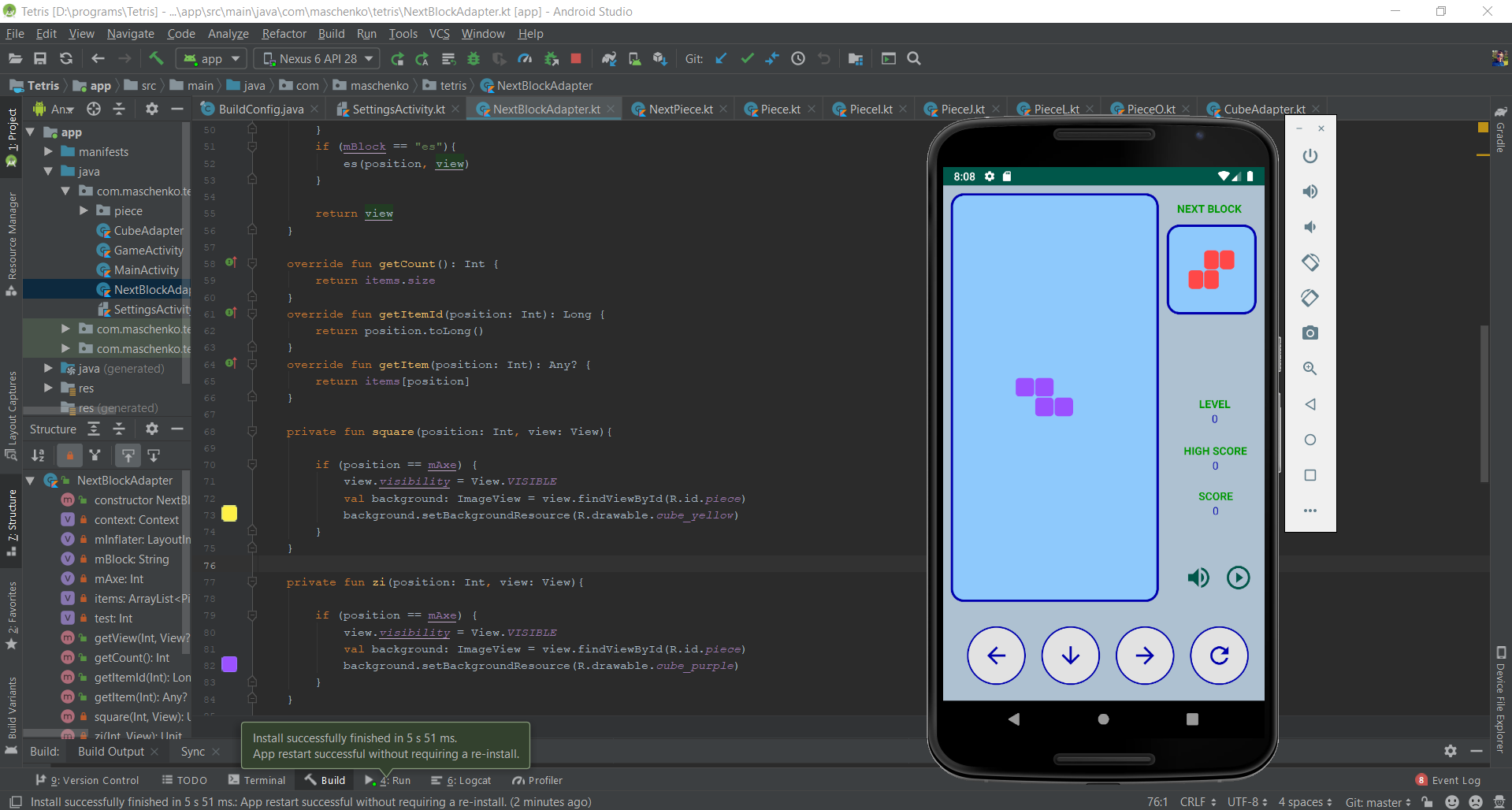


рис 2. Старт игры

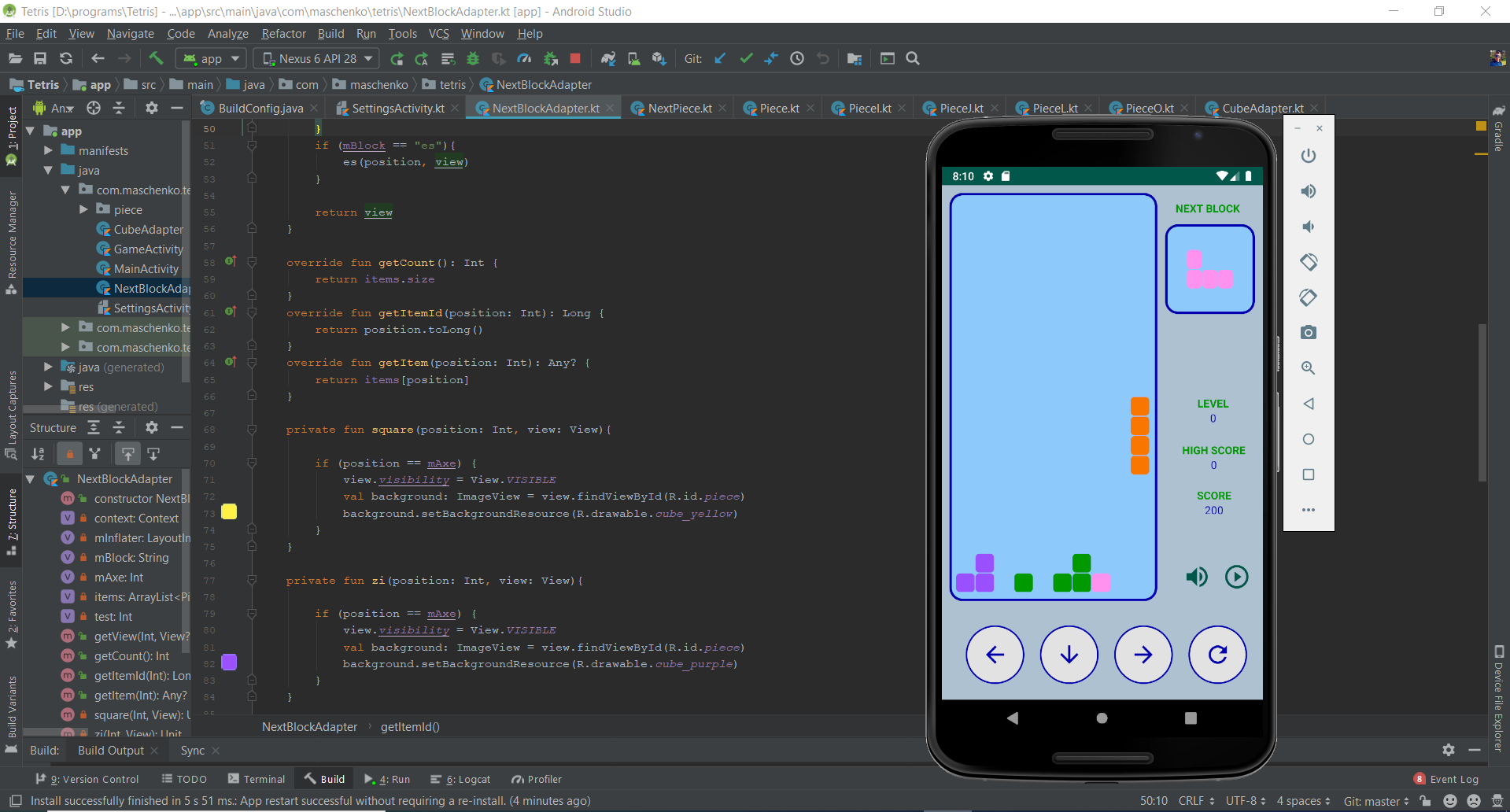


рис 3. Набор очков в игре

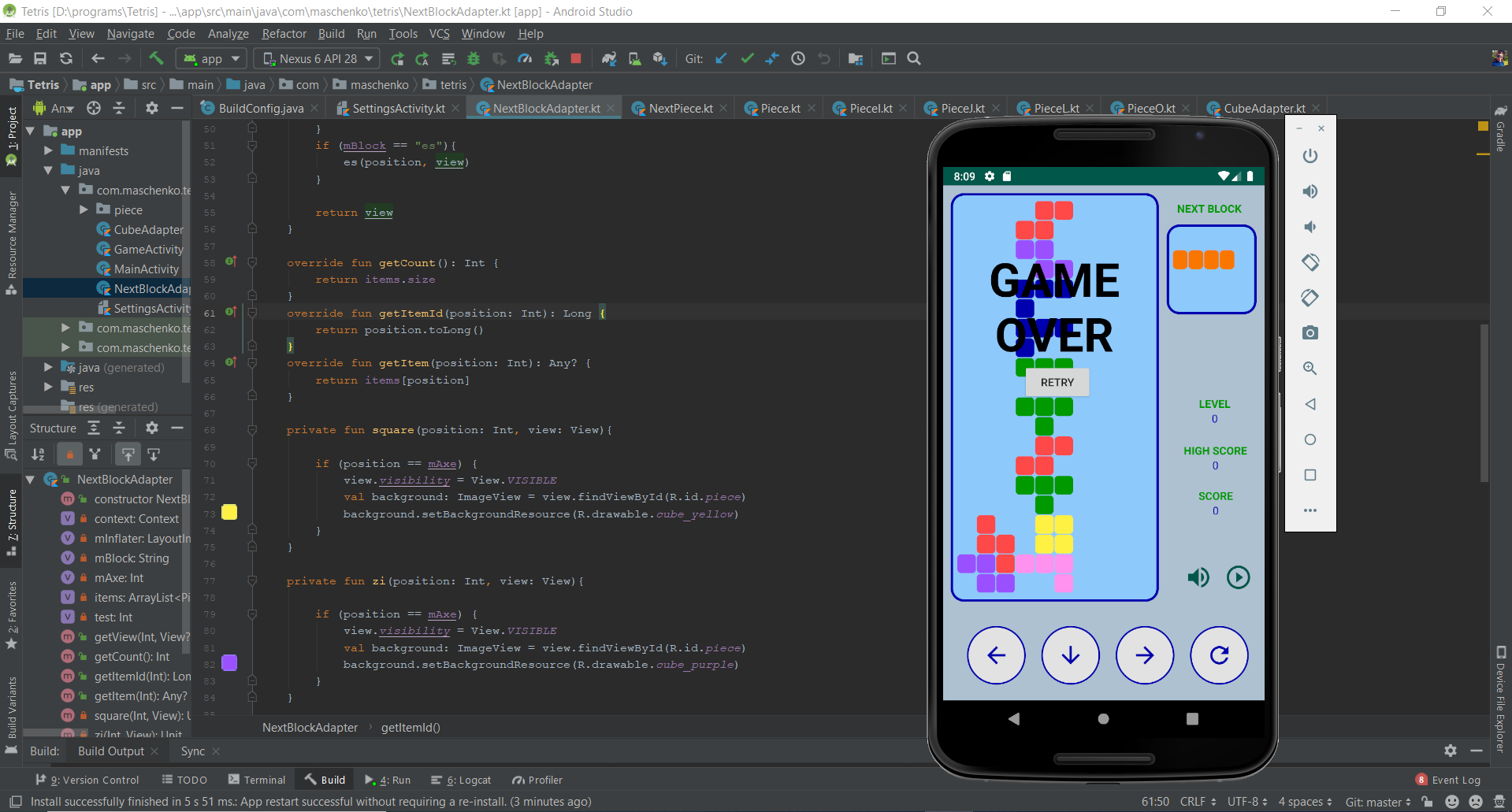


рис 4. Игра проиграна.

# **Заключение**

В данной курсовой работе реализована программа «Tetris» на языке Kotlin на Android.

В результате данной работы были получены знания и навыки в написании программ использующих функциональный подход к программированию, стандартных библиотек для разработки программ на Android, а также лучше усвоены возможности языка Kotlin.