Universidad Tecnológica de León
Ser, saber, hacer

UNIVERSIDAD TECNOLÓGICA DE LEÓN

INGENIERIA EN DESARROLLO Y GESTIÓN DE SOFTWARE

DESARROLLO PARA DISPOSITIVOS INTELIGENTES

EXAMEN SEGUNDO PARCIAL

IDGS 904

P R E S E N T A

22001638 LUIS ANDRÉS CISNEROS MENDOZA

LEÓN, GUANAJUATO, 04/07/2025

CODIGO

MainActivity.kt

```
package com.example.zoodiacochino

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import com.example.zoodiacochino.ui.theme.ZoodiacochinoTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            ZoodiacochinoTheme {
                // Inicia la aplicación
                AppZodiacoChino()
            }
        }
    }
}
```

AppZodiacoChino.kt

```
package com.example.zoodiacochino

import androidx.compose.runtime.Composable
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember

/**
 * Aplicación principal que gestiona la navegación entre pantallas.
 * - Pantallas: "usuario", "examen", "respuestas", "resultado".
 */
@Composable
fun AppZodiacoChino() {
    // Estado para controlar la pantalla actual
    val pantallaActual = remember { mutableStateOf("usuario") }
```

```
// Datos del usuario ingresados en la primera pantalla
val usuario = remember { mutableStateOf(Usuario()) }

// Respuestas del examen
val respuestas = remember { mutableStateOf(listOf<Int>()) }

// ViewModel que contiene las preguntas del examen
val viewModel = ExamenViewModel()

// Lista de imágenes de los signos del zodiaco chino (deben estar en res/drawable)
val imagenesSignos = listOf(
    R.drawable.rata, R.drawable.buey, R.drawable.tigre,
    R.drawable.conejo, R.drawable.dragon, R.drawable.serpiente,
    R.drawable.caballo, R.drawable.cabra, R.drawable.mono,
    R.drawable.gallo, R.drawable.perro, R.drawable.cerdo
)

// Navegación entre pantallas
when (pantallaActual.value) {
    "usuario" -> UsuarioScreen { user ->
        usuario.value = user
        pantallaActual.value = "examen"
    }
    "examen" -> ExamenScreen(
        preguntas = viewModel.preguntas,
        onTerminar = { respuestasUsuario ->
            respuestas.value = respuestasUsuario
            pantallaActual.value = "respuestas"
        }
    )
    "respuestas" -> RespuestasScreen(
        preguntas = viewModel.preguntas,
        respuestasUsuario = respuestas.value,
        onVerResultado = { pantallaActual.value = "resultado" }
    )
    "resultado" -> ResultadoScreen(
        usuario = usuario.value,
        preguntas = viewModel.preguntas,
        respuestasUsuario = respuestas.value,
```

```
            imagenesSignos = imagenesSignos,
            onVolver = { pantallaActual.value = "usuario" }
        )
    }
}
```

Usuario.kt

```
package com.example.zoodiacochino

/**
 * Clase de datos que representa la información del usuario.
 */
data class Usuario(
    var nombre: String = "",
    var apellidoPaterno: String = "",
    var apellidoMaterno: String = "",
    var diaNacimiento: String = "",
    var mesNacimiento: String = "",
    var anioNacimiento: String = "",
    var sexo: String = ""
)
```

Pregunta.kt

```
package com.example.zoodiacochino

/**
 * Clase de datos que representa una pregunta del examen.
 * @param respuestaCorrecta Índice de la respuesta correcta (0 a 3).
 */
data class Pregunta(
    val texto: String,
    val opciones: List<String>,
    val respuestaCorrecta: Int
)
```

ExamenViewModel.kt

```
package com.example.zoodiacochino

import androidx.lifecycle.ViewModel

class ExamenViewModel : ViewModel() {

    // Lista de preguntas
    private val _preguntas = listOf(
        Pregunta("¿Cuál es la suma de 2 + 2?", listOf("8", "6", "4", "3"), 2),
        Pregunta("¿Capital de Francia?", listOf("Roma", "París", "Madrid", "Londres"), 1),
        Pregunta("¿Cuál es el resultado de 5 x 3?", listOf("15", "10", "8", "13"), 0),
        Pregunta("¿Color del cielo despejado?", listOf("Rojo", "Azul", "Verde", "Gris"), 1),
        Pregunta("¿Cuántos días tiene una semana?", listOf("5", "6", "7", "8"), 2),
        Pregunta("¿Cuál es el planeta rojo?", listOf("Tierra", "Venus", "Marte", "Saturno"), 2)
    )

    // Lo ponemos públicamente ya que se necesita acceder desde otras pantallas
    val preguntas: List<Pregunta> = _preguntas
}
```

UsuarioScreen.kt

```
package com.example.zoodiacochino

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.unit.dp

/**
 * Pantalla para ingresar los datos del usuario.
 * @param onSiguiente Callback que se ejecuta al presionar "Siguiente", enviando los datos del usuario.
```

```
 */
@Composable
fun UsuarioScreen(onSiguiente: (Usuario) -> Unit) {
    // Estados para los campos del formulario
    var nombre by remember { mutableStateOf("") }
    var apellidoP by remember { mutableStateOf("") }
    var apellidoM by remember { mutableStateOf("") }
    var dia by remember { mutableStateOf("") }
    var mes by remember { mutableStateOf("") }
    var año by remember { mutableStateOf("") }
    var sexo by remember { mutableStateOf("Masculino") }
    var showError by remember { mutableStateOf(false) }

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
            .verticalScroll(rememberScrollState())
    ) {
        Text(
            "Datos Personales",
            style = MaterialTheme.typography.titleLarge,
            modifier = Modifier
                .padding(bottom = 16.dp)
                .align(Alignment.CenterHorizontally)
        )

        // Campo para el nombre
        OutlinedTextField(
            value = nombre,
            onValueChange = { nombre = it },
            label = { Text("Nombre") },
            modifier = Modifier
                .fillMaxWidth()
                .padding(bottom = 8.dp),
            isError = showError && nombre.isBlank()
        )

        // Campos para apellidos
        OutlinedTextField(
```

```
    value = apellidoP,
    onValueChange = { apellidoP = it },
    label = { Text("Apellido Paterno") },
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 8.dp)
)

OutlinedTextField(
    value = apellidoM,
    onValueChange = { apellidoM = it },
    label = { Text("Apellido Materno") },
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 8.dp)
)

// Campos para fecha de nacimiento
Row(Modifier.fillMaxWidth()) {
    OutlinedTextField(
        value = dia,
        onValueChange = { dia = it },
        label = { Text("Día") },
        modifier = Modifier.weight(1f).padding(end = 8.dp)
    )
    OutlinedTextField(
        value = mes,
        onValueChange = { mes = it },
        label = { Text("Mes") },
        modifier = Modifier.weight(1f).padding(end = 8.dp)
    )
    OutlinedTextField(
        value = año,
        onValueChange = { año = it },
        label = { Text("Año") },
        modifier = Modifier.weight(1f),
        isError = showError && año.isBlank()
    )
}
```

```
// Selección de sexo
Text("Sexo:", modifier = Modifier.padding(top = 8.dp))
Row(verticalAlignment = Alignment.CenterVertically) {
    RadioButton(
        selected = sexo == "Masculino",
        onClick = { sexo = "Masculino" }
    )
    Text("Masculino")
    RadioButton(
        selected = sexo == "Femenino",
        onClick = { sexo = "Femenino" },
        modifier = Modifier.padding(start = 16.dp)
    )
    Text("Femenino")
}

// Mostrar error si los campos obligatorios están vacíos
if (showError && (nombre.isBlank() || año.isBlank())) {
    Text(
        "Por favor complete todos los campos obligatorios",
        color = Color.Red,
        modifier = Modifier.padding(top = 8.dp)
    )
}

// Botón para continuar
Button(
    onClick = {
        if (nombre.isNotBlank() && año.isNotBlank()) {
            val user = Usuario(
                nombre, apellidoP, apellidoM,
                dia, mes, año, sexo
            )
            onSiguiente(user)
        } else {
            showError = true
        }
    },
    modifier = Modifier
        .fillMaxWidth()
```

```
            .padding(top = 16.dp)
        ) {
            Text("Siguiente")
        }
    }
}
```

ExamenScreen.kt

```
package com.example.zoodiacochino

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.runtime.mutableStateListOf
import androidx.compose.runtime.remember
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp

/**
 * Pantalla que muestra el examen con preguntas y opciones de respuesta.
 * @param onTerminar Callback que se ejecuta al finalizar el examen, enviando las respuestas del usuario.
 */
@Composable
fun ExamenScreen(preguntas: List<Pregunta>, onTerminar: (List<Int>) -> Unit) {
    // Estado para almacenar las respuestas seleccionadas por el usuario
    val respuestas = remember { mutableStateListOf<Int>() }

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
            .verticalScroll(rememberScrollState())
    ) {
        Text(
            "Examen de Conocimientos",
```

```
    style = MaterialTheme.typography.titleLarge,
    modifier = Modifier.padding(bottom = 16.dp)
)

// Mostrar cada pregunta con sus opciones
preguntas.forEachIndexed { index, pregunta ->
    Text(
        "${index + 1}. ${pregunta.texto}",
        modifier = Modifier.padding(vertical = 8.dp)
    )

    // Mostrar las opciones de respuesta como RadioButtons
    pregunta.opciones.forEachIndexed { opcionIndex, opcion ->
        Row(
            verticalAlignment = Alignment.CenterVertically,
            modifier = Modifier.padding(horizontal = 16.dp)
        ) {
            RadioButton(
                selected = respuestas.getOrNull(index) == opcionIndex,
                onClick = {
                    if (respuestas.size <= index) {
                        respuestas.add(opcionIndex)
                    } else {
                        respuestas[index] = opcionIndex
                    }
                }
            )
            Text(opcion)
        }
    }
}

// Botón para terminar el examen
Button(
    onClick = { onTerminar(respuestas) },
    modifier = Modifier
        .padding(top = 16.dp)
        .align(Alignment.CenterHorizontally)
) {
    Text("Terminar Examen")
```

```
    }
  }
}
```

RespuestasScreen.kt

```
package com.example.zoodiacochino

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.unit.dp

/**
 * Pantalla que muestra las respuestas del usuario con retroalimentación visual.
 * @param onVerResultado Callback para navegar a la pantalla de resultados.
 */
@Composable
fun RespuestasScreen(
    preguntas: List<Pregunta>,
    respuestasUsuario: List<Int>,
    onVerResultado: () -> Unit
) {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
            .verticalScroll(rememberScrollState())
    ) {
        Text(
            "Tus Respuestas",
            style = MaterialTheme.typography.titleLarge,
            modifier = Modifier.padding(bottom = 16.dp)
        )
```

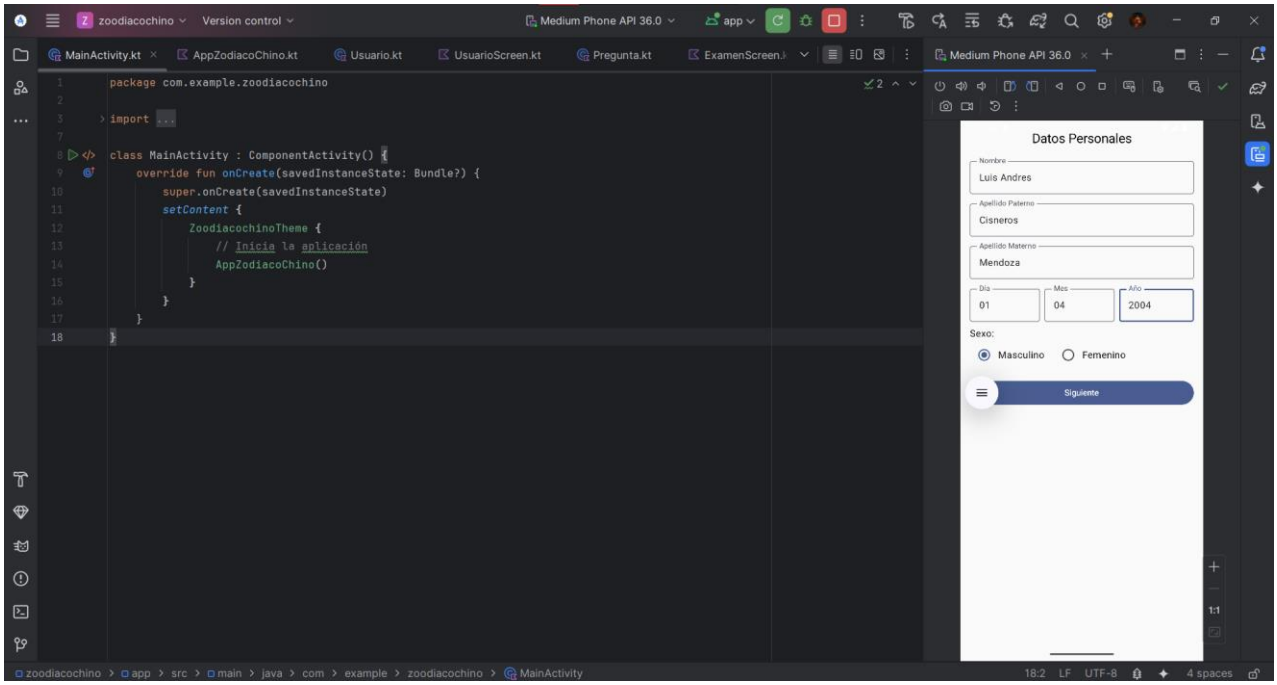
```
        // Mostrar cada pregunta con las respuestas del usuario
        preguntas.forEachIndexed { index, pregunta ->
            Text(
                "Pregunta ${index + 1}: ${pregunta.texto}",
                modifier = Modifier.padding(vertical = 8.dp)
            )

            // Mostrar cada opción con colores según su corrección
            pregunta.opciones.forEachIndexed { opcionIndex, opcion ->
                val esCorrecta = opcionIndex == pregunta.respuestaCorrecta
                val fueSeleccionada = respuestasUsuario.getOrNull(index) == opcionIndex

                val color = when {
                    esCorrecta -> Color.Green
                    fueSeleccionada && !esCorrecta -> Color.Red
                    else -> Color.Black
                }

                Text(
                    text = opcion,
                    color = color,
                    modifier = Modifier.padding(start = 16.dp)
                )
            }
        }

        // Botón para ver el resultado final
        Button(
            onClick = onVerResultado,
            modifier = Modifier
                .padding(top = 16.dp)
                .align(Alignment.CenterHorizontally)
        ) {
            Text("Ver Resultados")
        }
    }
}
```

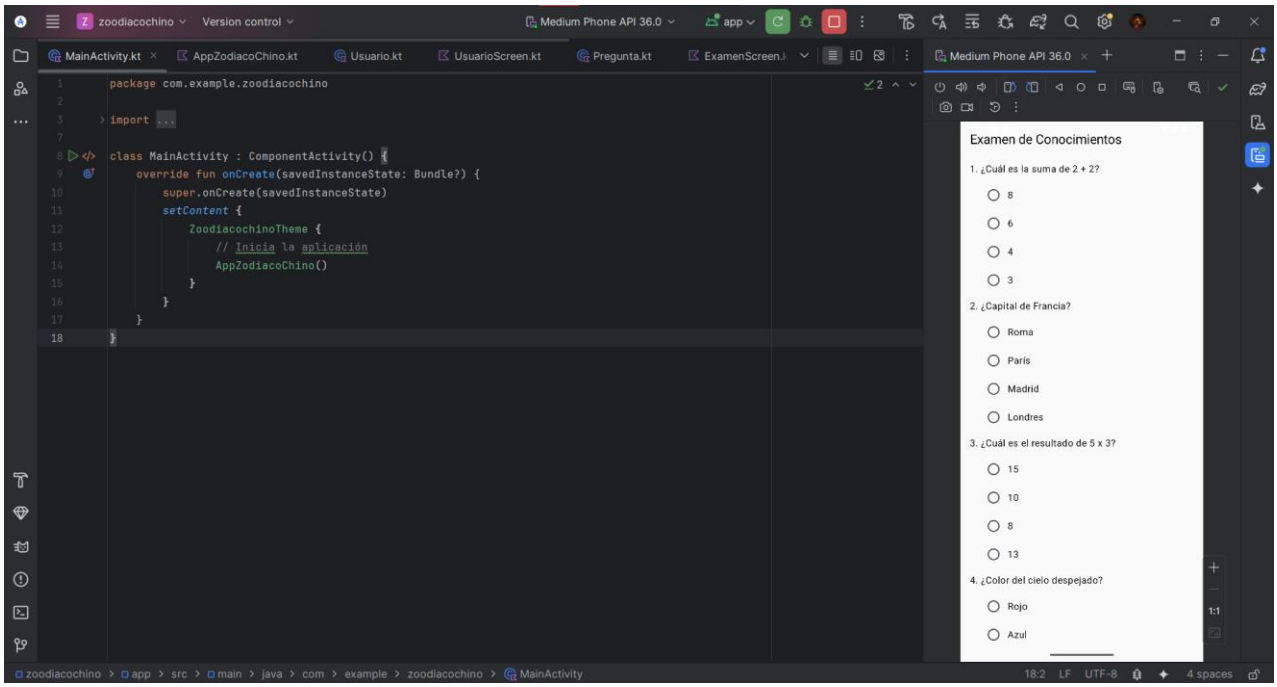
ResultadoScreen.kt

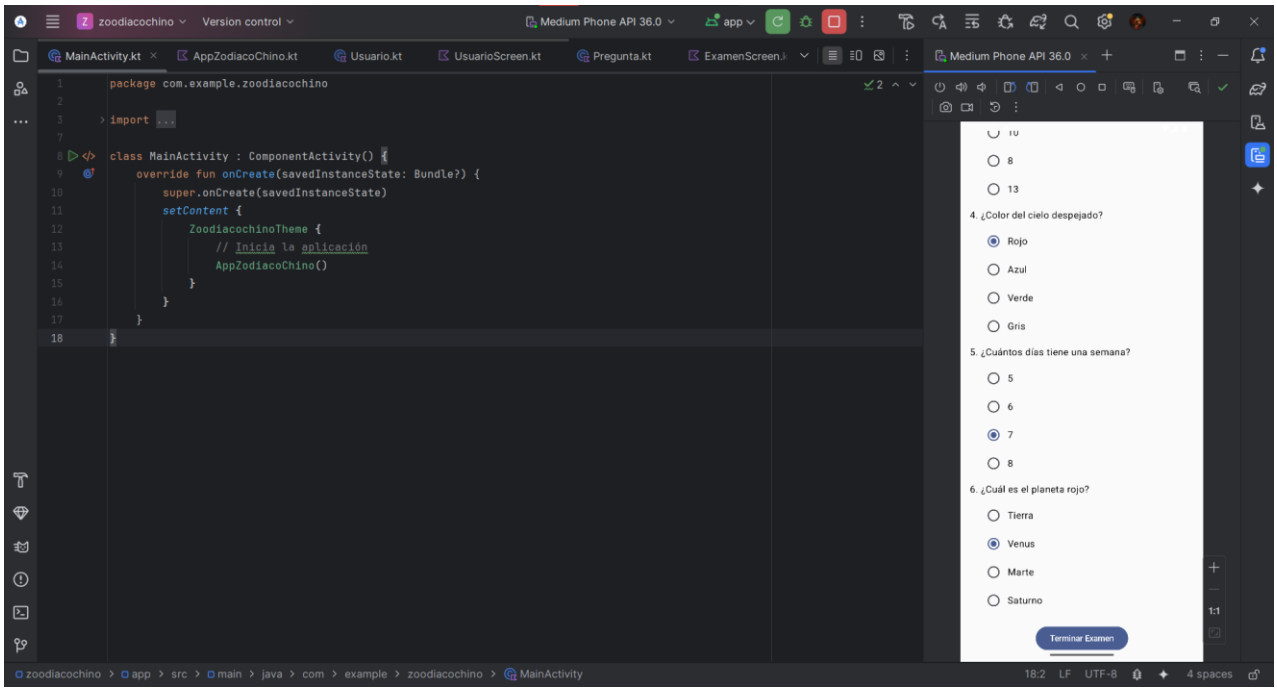
```
package com.example.zoodiacochino

import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.unit.dp
import java.util.Calendar

/**
 * Pantalla que muestra los resultados finales del usuario.
 * @param onVolver Callback para volver al inicio.
 */
@Composable
fun ResultadoScreen(
    usuario: Usuario,
    preguntas: List<Pregunta>,
    respuestasUsuario: List<Int>,
    imagenesSignos: List<Int>,
    onVolver: () -> Unit
) {
    // Cálculo de la edad y el signo chino
    val edad = Calendar.getInstance().get(Calendar.YEAR) - usuario.anioNacimiento.toIntOrNull()!!
    val signos = listOf("Rata", "Buey", "Tigre", "Conejo", "Dragón", "Serpiente",
        "Caballo", "Cabra", "Mono", "Gallo", "Perro", "Cerdo")
    val signoIndex = (usuario.anioNacimiento.toInt() - 1900) % 12
    val signo = signos[signoIndex]
    val correctas = preguntas.indices.count {
        respuestasUsuario.getOrNull(it) == preguntas[it].respuestaCorrecta
    }

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
```

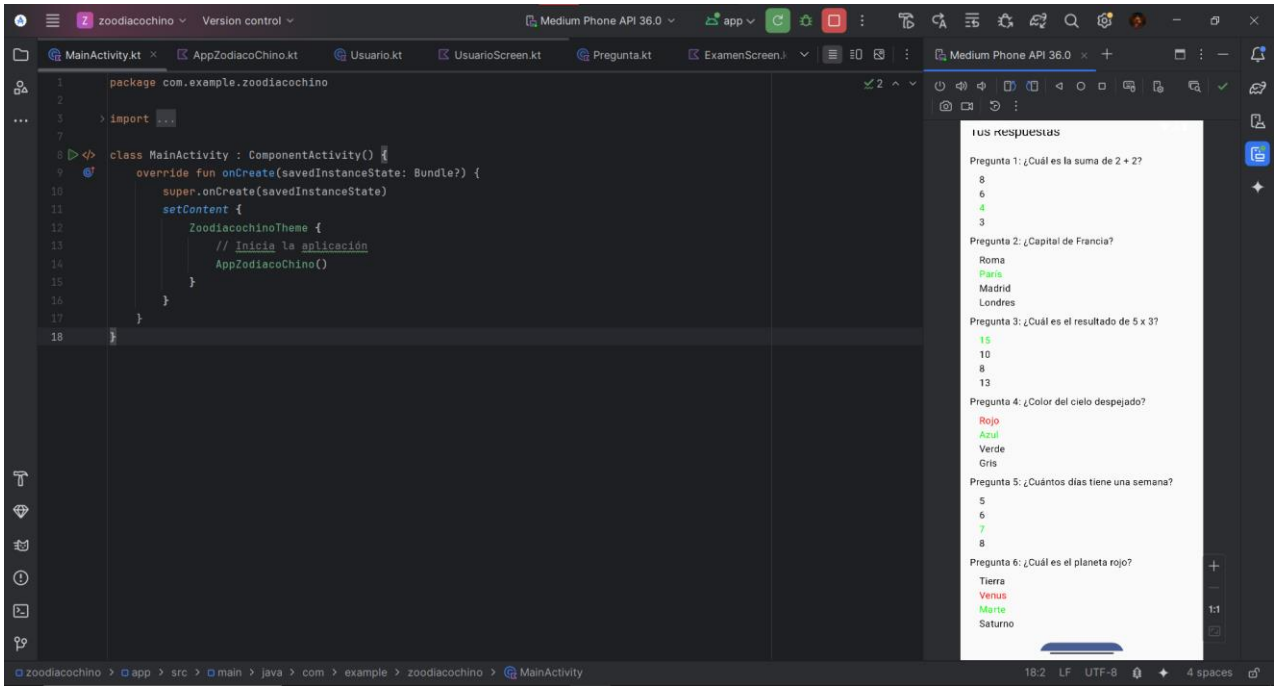
```
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Text(
            "¡Resultados Finales!",
            style = MaterialTheme.typography.titleLarge,
            modifier = Modifier.padding(bottom = 16.dp)
        )

        Text("Hola ${usuario.nombre} ${usuario.apellidoPaterno}")
        Text("Tienes $edad años")
        Text("y tu signo zodiacal es $signo")
        // Imagen del signo chino
        Image(
            painter = painterResource(id = imagenesSignos[signoIndex]),
            contentDescription = signo,
            modifier = Modifier
                .padding(16.dp)
                .size(120.dp)
        )
        Text("Calificacion $correctas")

        // Botón para volver al inicio
        Button(
            onClick = onVolver,
            modifier = Modifier.padding(top = 16.dp)
        ) {
            Text("Volver al Inicio")
        }
    }
}
```

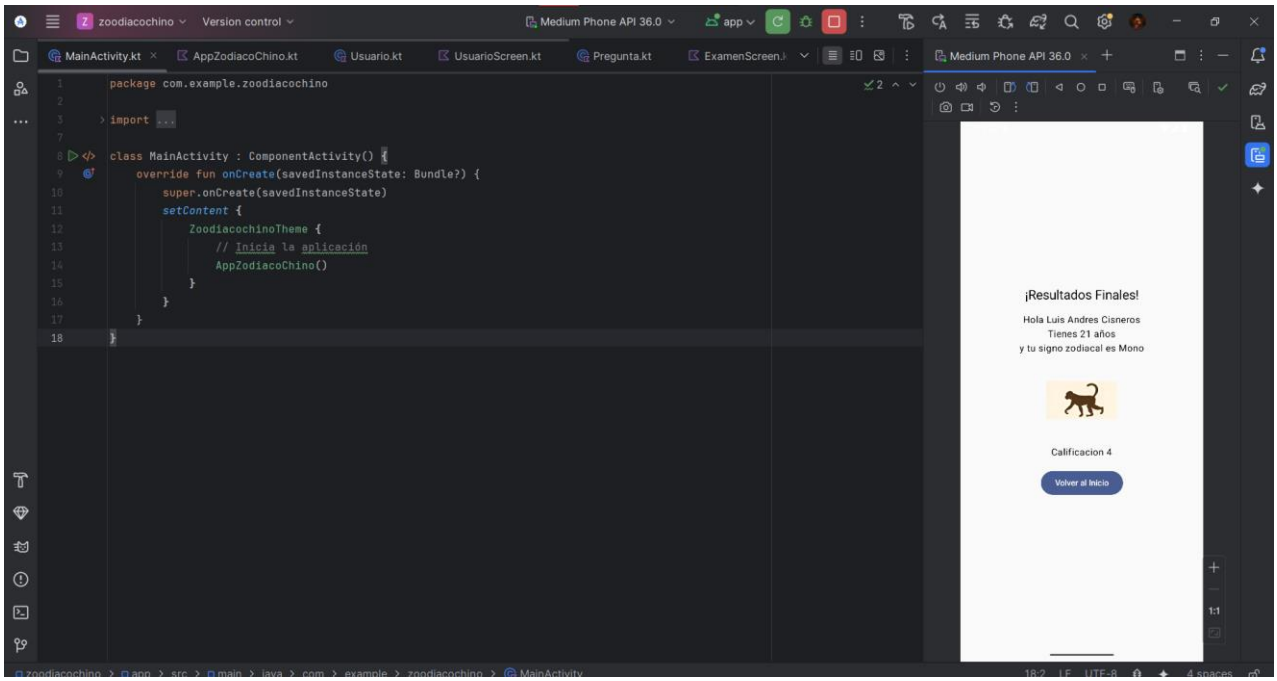
FUNCIONAMIENTO






zoodiacochino
Version control
Medium Phone API 36.0
app
MainActivity.kt
AppZodiacoChino.kt
Usuario.kt
UsuarioScreen.kt
Pregunta.kt
ExamenScreen.kt
package com.example.zoodiacochino
import ...
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContent {
ZoodiacochinoTheme {
// Inicia la aplicación
AppZodiacoChino()
}
}
}
}
Medium Phone API 36.0
8
13
4. ¿Color del cielo despejado?
Rojo
Azul
Verde
Gris
5. ¿Cuántos días tiene una semana?
5
6
7
8
6. ¿Cuál es el planeta rojo?
Tierra
Venus
Marte
Saturno
Terminar Examen
zoodiacochino › app › src › main › java › com › example › zoodiacochino › MainActivity
18:2 LF UTF-8 4 spaces


zoodiacochino
Version control
Medium Phone API 36.0
app
MainActivity.kt
AppZodiacoChino.kt
Usuario.kt
UsuarioScreen.kt
Pregunta.kt
ExamenScreen.kt
package com.example.zoodiacochino
import ...
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContent {
ZoodiacochinoTheme {
// Inicia la aplicación
AppZodiacoChino()
}
}
}
}
Medium Phone API 36.0
Tus respuestas
Pregunta 1: ¿Cuál es la suma de 2 + 2?
8
6
4
3
Pregunta 2: ¿Capital de Francia?
Roma
París
Madrid
Londres
Pregunta 3: ¿Cuál es el resultado de 5 x 3?
15
10
8
13
Pregunta 4: ¿Color del cielo despejado?
Rojo
Azul
Verde
Gris
Pregunta 5: ¿Cuántos días tiene una semana?
5
6
7
8
Pregunta 6: ¿Cuál es el planeta rojo?
Tierra
Venus
Marte
Saturno
zoodiacochino › app › src › main › java › com › example › zoodiacochino › MainActivity
18:2 LF UTF-8 4 spaces


zoodiacochino
Version control
Medium Phone API 36.0
app
MainActivity.kt
AppZodiacoChino.kt
Usuario.kt
UsuarioScreen.kt
Pregunta.kt
ExamenScreen.
Medium Phone API 36.0
package com.example.zoodiacochino
import ...
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContent {
ZoodiacochinoTheme {
// Inicia la aplicación
AppZodiacoChino()
}
}
}
}
¡Resultados Finales!
Hola Luis Andres Cisneros
Tienes 21 años
y tu signo zodiacal es Mono
Calificacion 4
Volver al Inicio
1:1
zoodiacochino > app > src > main > java > com > example > zoodiacochino > MainActivity
18:2
LF
UTF-8
4 spaces