

Tarea 01: Introducción a Kotlin

Contenido

Objetivo.....	2
Instrucciones	2
Evaluación	2
Rúbrica	2
Fecha de entrega.....	2
Problemas asignados a resolver.....	2
Respuesta del problema 14.....	3
Respuesta del problema 39.....	5
Referencias.....	10

Objetivo

El Objetivo de esta asignación es que el estudiante se enfrente a las bases del lenguaje de programación Kotlin resolviendo problemas en los cuales se sienta seguro y que le permitan entender el funcionamiento de este.

Instrucciones

De la página: [Ninety Nine Kotlin Problems](#), seleccionar 2 problemas 1 fácil y uno medio o difícil. Problemas fáciles: P05 - P28; problemas medios: P30 - P50; problemas difíciles: P55 – 99.

Resolver los problemas dentro de un archivo con extensión “.kts” (nota: Verificar que el archivo no sea “.kt”); “.kts” se refiere a Kotlin Script, mientras que “.kt” se refiere a solo un archivo Kotlin.

Evaluación

Se evaluará que el archivo contenga:

- Nombre y matrícula dentro del archivo.
- Números de problemas a resolver.
- Pseudocódigo o algoritmo de los problemas a resolver.
- Funcionalidad de los problemas a resolver.
- Casos de prueba:
 - Todos los dados por el problema.
 - Al menos 2 realizados por el estudiante.

Rúbrica

La siguiente rúbrica será utilizada para cada uno de los problemas:

Concepto	Porcentaje
Pseudocódigo	30%
Código	40%
Funcionamiento	30%

Fecha de entrega

27/02/2020 11:59 a través de GitHub Classroom.

Problemas asignados a resolver

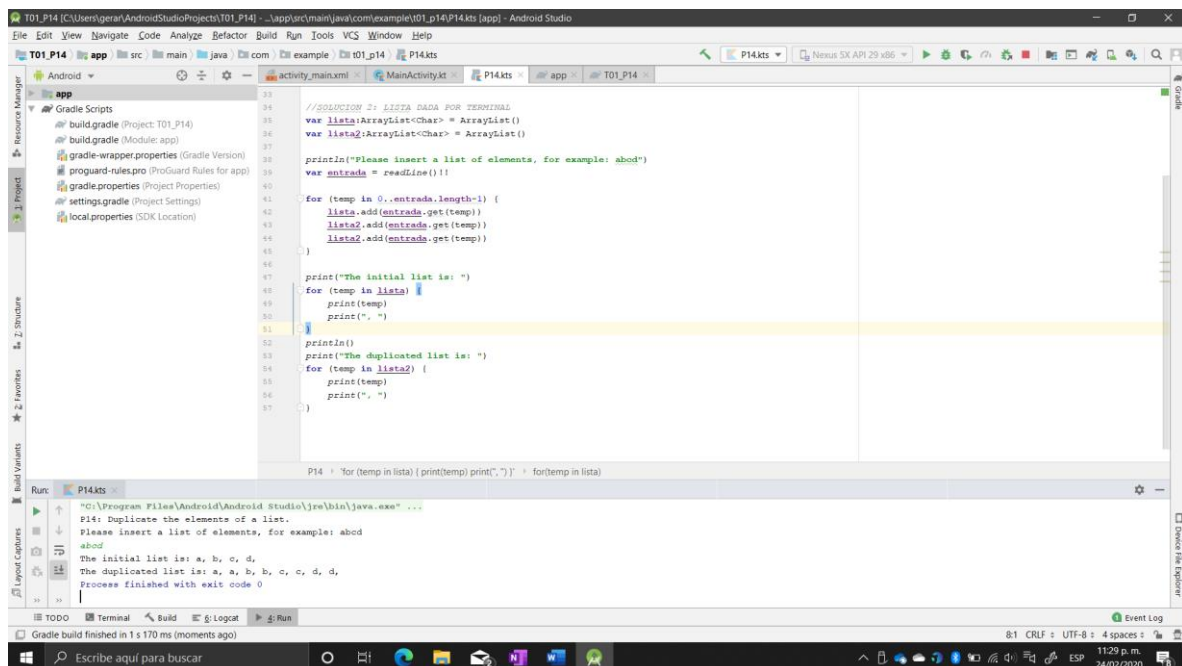
Los problemas por resolver son el 14 y el 39.

Respuesta del problema 14

El código utilizado es el siguiente:

```
package com.example.t01_p14
println("P14: Duplicate the elements of a list.")
var lista:ArrayList<Char> = ArrayList()
var lista2:ArrayList<Char> = ArrayList()
println("Please insert a list of elements, for example: abcd")
var entrada = readLine()!!
for (temp in 0..entrada.length-1) {
    lista.add(entrada.get(temp))
    lista2.add(entrada.get(temp))
    lista2.add(entrada.get(temp))
}
print("The initial list is: ")
for (temp in lista) {
    print(temp)
    print(", ")
}
println()
print("The duplicated list is: ")
for (temp in lista2) {
    print(temp)
    print(", ")
}
```

Los casos de prueba dados son el asignado por el problema “a,b,c,d” y los dados por el alumno “Gerardo” y “DesarrolloAplicacionesMoviles”, como se pueden observar en las siguientes capturas de pantalla, a manera de evidencia.



Respuesta del problema 39

El código utilizado es el siguiente:

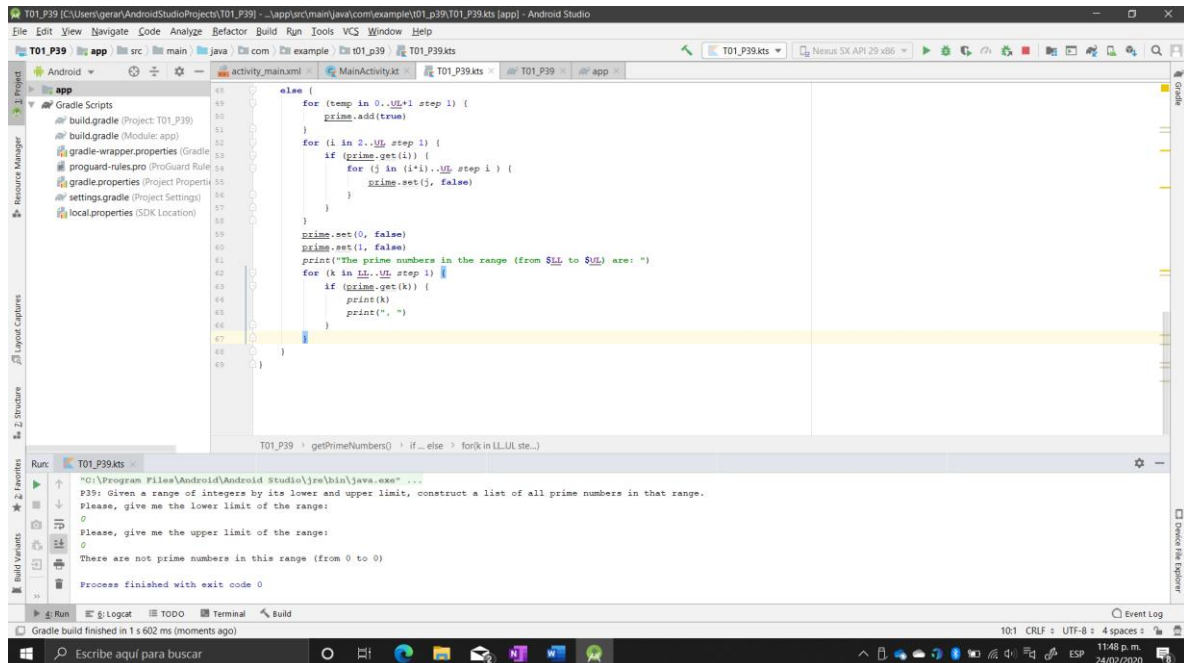
```
package com.example.t01_p39
// Gerardo Daniel Naranjo Gallegos, A01209499
println("P39: Given a range of integers by its lower and upper
limit, construct a list of all prime numbers in that range.")
var repeat = true
var LL = 0
var UL = 0
do {
    println("Please, give me the lower limit of the range: ")
    var lowerLimit = readLine()!!
    try {
        LL = lowerLimit.toInt()
    } catch (e: NumberFormatException) {
        println("Please, give a valid number, like: 0")
        break
    }
    println("Please, give me the upper limit of the range: ")
    var upperLimit = readLine()!!
    try {
        UL = upperLimit.toInt()
    } catch (e: NumberFormatException) {
        println("Please, give a valid number, like: 1")
        break
    }
    if (LL <= UL) {
        repeat = false
        getPrimeNumbers()
    }
    else {
        println("Please, give a valid range.")
    }
} while (repeat)
// Solved using Sieve of Eratosthenes:
https://www.geeksforgeeks.org/sieve-of-eratosthenes/
fun getPrimeNumbers () {
    var prime:ArrayList<Boolean> = ArrayList()
    if (UL < 2) {
        println("There are not prime numbers in this range (from
$LL to $UL)")
        return
    }
    else if (UL == 2) {
        println("The only prime number in this range (from $LL to
$UL) is: 2")
        return
    }
    else {
        for (temp in 0..UL+1 step 1) {
```

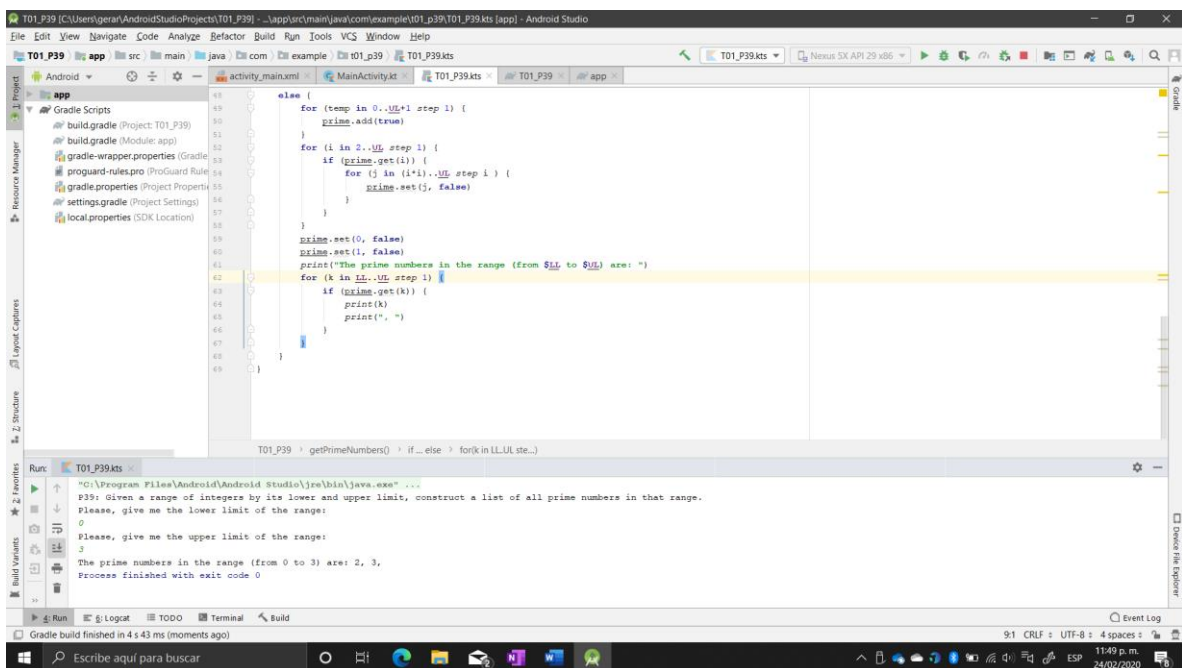
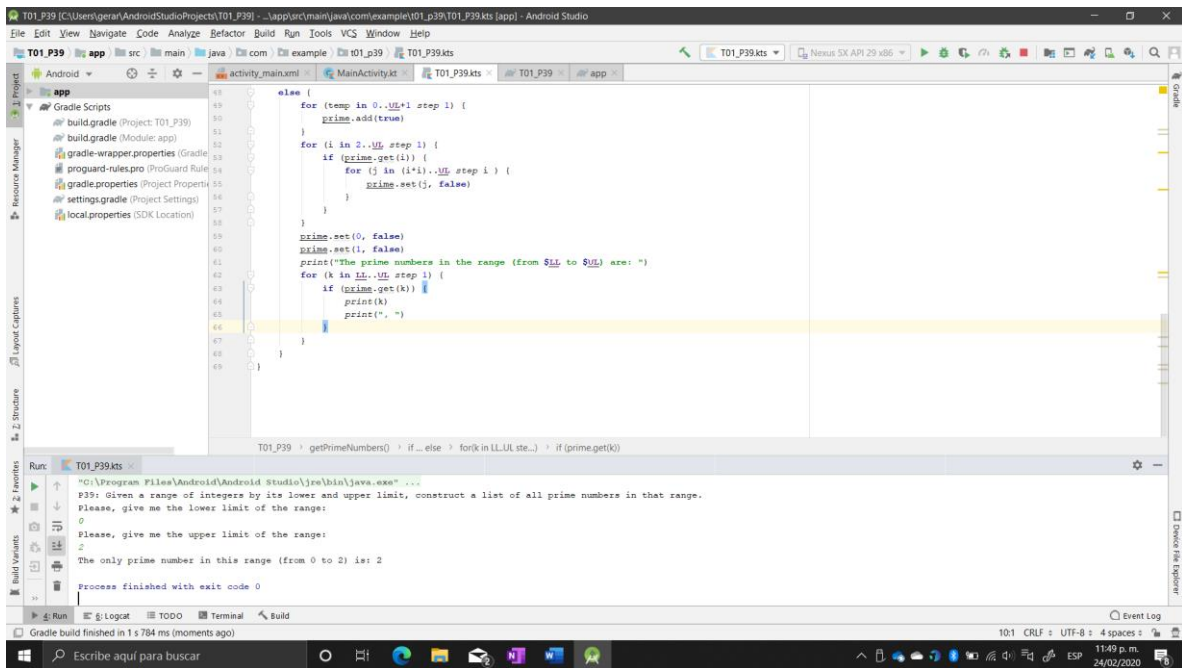
```

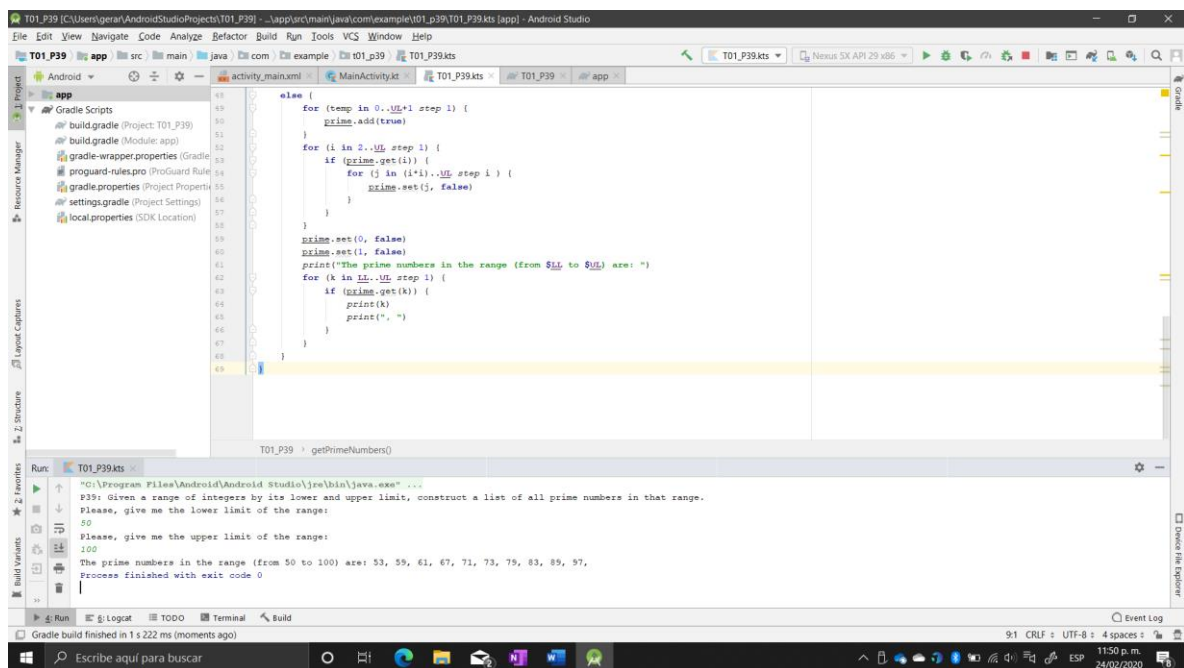
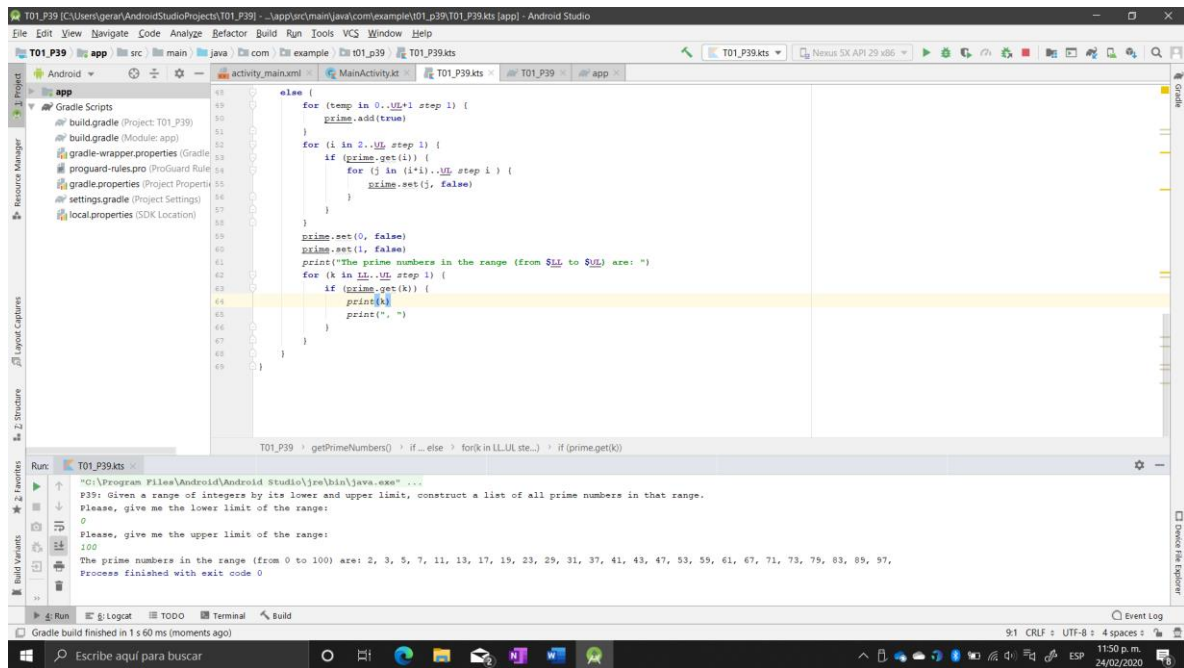
        prime.add(true)
    }
    for (i in 2..UL step 1) {
        if (prime.get(i)) {
            for (j in (i*i)..UL step i ) {
                prime.set(j, false)
            }
        }
    }
    prime.set(0, false)
    prime.set(1, false)
    print("The prime numbers in the range (from $LL to $UL)
are: ")
    for (k in LL..UL step 1) {
        if (prime.get(k)) {
            print(k)
            print(", ")
        }
    }
}
}
}

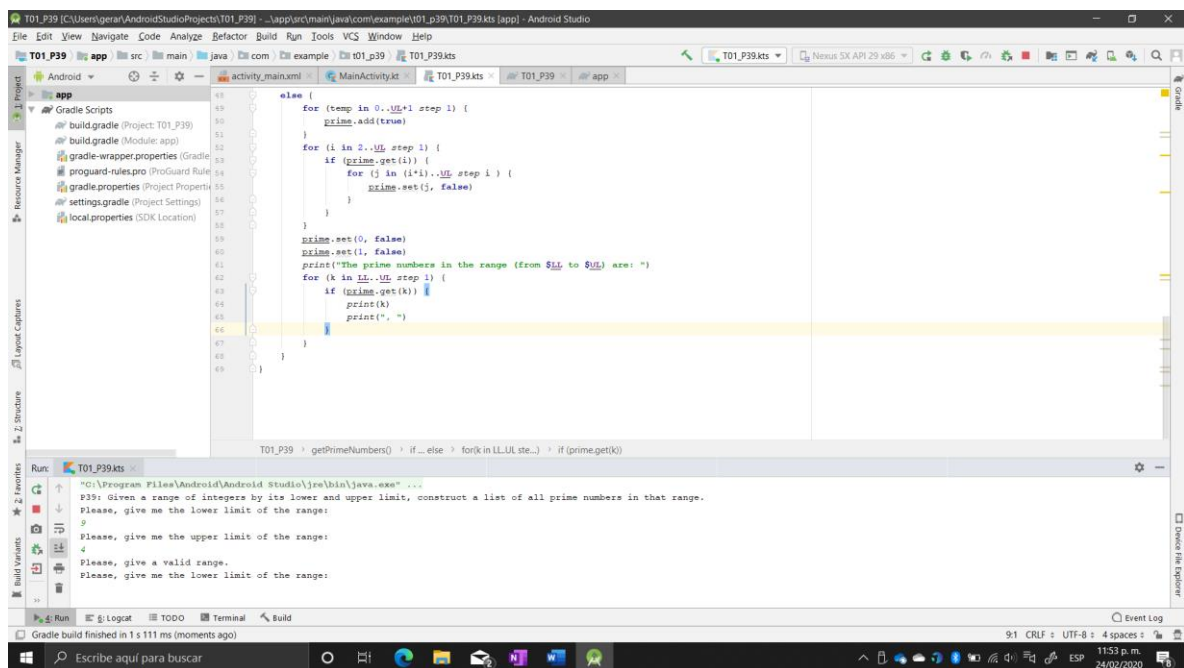
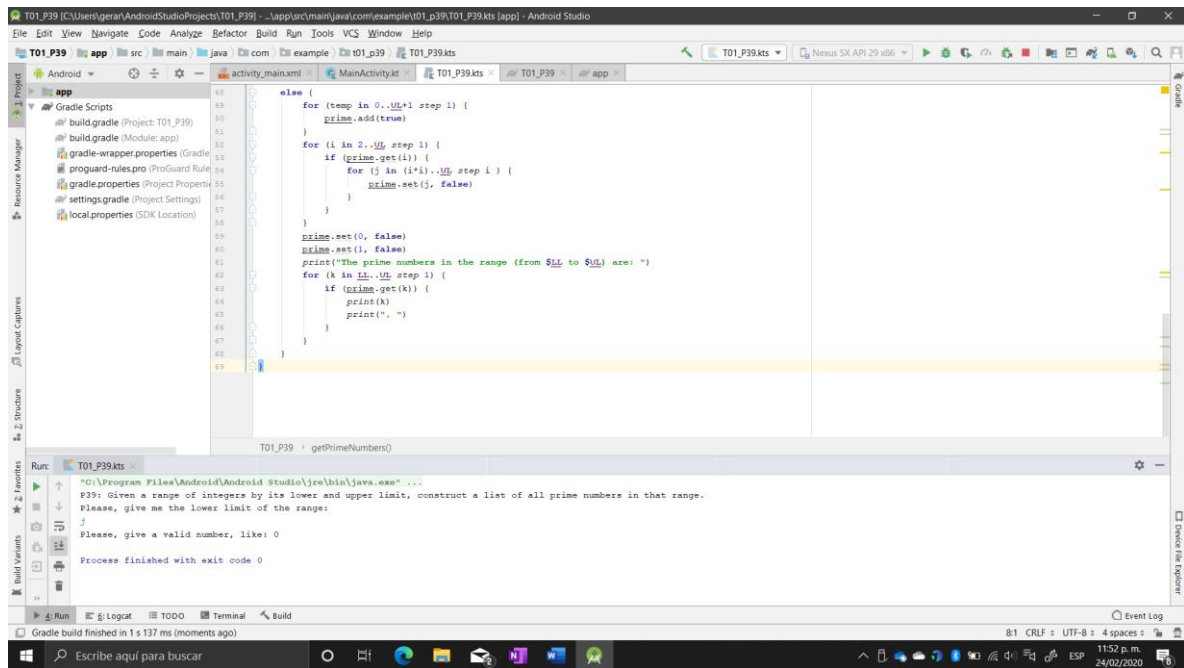
```

Respecto a los casos de prueba, se utiliza el dado por el problema (rango de 7 a 31), así como los siguientes rangos aportados por el alumno: 0 a 0, 0 a 2, 0 a 3, 0 a 100 y 50 a 100. Además, dos últimos casos de prueba para comprobar la validación de recibir números válidos.









Referencias

dkandalov. (2020). *Ninety-Nine Kotlin Problems*. Obtenido de GitHub:
<https://github.com/dkandalov/kotlin-99>

Kotlin. (2020). *Basic Syntax*. Obtenido de Kotlin Lang: <https://kotlinlang.org/docs/reference/basic-syntax.html>

Kotlin. (2020). *Control Flow: if, when, for, while*. Obtenido de Kotlin Lang:
<https://kotlinlang.org/docs/reference/control-flow.html>

Kotlin. (2020). *List*. Obtenido de Kotlin Lang:
<https://kotlinlang.org/api/latest/jvm/stdlib/kotlin.collections/-list/index.html>

Kotlin. (2020). *Ranges and Progressions*. Obtenido de Kotlin Lang:
<https://kotlinlang.org/docs/reference/ranges.html>

Mithun Kumar, Radhesh Sarma & Shivam Kumar Nayak. (2020). *Sieve of Eratosthenes*. Obtenido de Geeks for Geeks: <https://www.geeksforgeeks.org/sieve-of-eratosthenes/>

Parewa Labs Pvt. Ltd. (2020). *Kotlin Basic Input/Output*. Obtenido de Programiz:
<https://www.programiz.com/kotlin-programming/input-output>

Parewa Labs Pvt. Ltd. (2020). *Kotlin Program to Check if a String is Numeric*. Obtenido de Programiz: <https://www.programiz.com/kotlin-programming/examples/check-string-numeric>