**Task # 1:** Design an interface:

**Result/Output:**

import UIKit

//Chapter 1

var str = "Hello, Rida"

str += "!"

print(str)

//Chapter 2

let numberOfStoplights: Int = 3

var population: Int

population = 1000

let townName: String = "Islamabad"

let townDescription = "\(townName) has a population of \(population) and \(numberOfStoplights) stoplights."

print(townDescription)

//Chapter 3

var message: String

var hasPostOffice: Bool = true

if population < 10000 {

    message = "\(population) is a small town!"

} else if population >= 10000 && population < 50000 {

    message = "\(population) is a medium town!"

} else {

    if population >= 10000 && population < 50000 {

        message = "\(population) is a medium town!"

    } else {

        message = "\(population) is pretty big!"

    }

}

print(message)

if !hasPostOffice {

    print("Where do we buy stamps?")

}

//Chapter 4

print("The maximum Int value is \(Int.max).")

print("The minimum Int value is \(Int.min).")

print("The maximum value for a 32-bit integer is \(Int32.max).")

print("The minimum value for a 32-bit integer is \(Int32.min).")

print("The maximum UInt value is \(UInt.max).")

print("The minimum UInt value is \(UInt.min).")

print("The maximum value for a 32-bit unsigned integer is \(UInt32.max).")

print("The minimum value for a 32-bit unsigned integer is \(UInt32.min).")

let numberOfPeople: UInt = 40

let volumeAdjustment: Int32 = -1000

print(10 + 20)

print(30 - 5)

print(5 \* 6)

print(25 / 5)

print(25 % 5)

var x = 10

x += 10 // Is equivalent to: x = x + 10

print("x has had 10 added to it and is now \(x)")

x -= 5 // Is equivalent to: x = x - 5

print("x has had 5 subtracted from it and is now \(x)")

let d1 = 1.1 // Implicitly Double

let d2: Double = 1.1

let f1: Float = 100.3

print(10.0 + 11.4)

print(11.0 / 3.0)

//Chapter 5

var statusCode: Int = 404

var statusMessage: String = ""

switch statusCode {

case 100, 101:

    statusMessage += " Informational, \(statusCode)."

case 204:

    statusMessage += " Successful but no content, 204."

case 300...307:

    statusMessage += " Redirection, \(statusCode)."

case 400...417:

    statusMessage += " Client error, \(statusCode)."

case 500...505:

    statusMessage += " Server error, \(statusCode)."

case let unknownCode where (unknownCode >= 200 && unknownCode < 300)

    || unknownCode > 505:

    statusMessage = "\(unknownCode) is not a known error code."

default:

    statusMessage = "\(statusCode) is not a known error code."

}

print(statusMessage)

let firstErrorCode = 404

let secondErrorCode = 200

let errorCodes = (firstErrorCode, secondErrorCode)

switch errorCodes {

case (404, 404):

    print("No items found.")

case (404, \_):

    print("First item not found.")

case (\_, 404):

    print("Second item not found.")

default:

    print("All items found.")

}

let point = (x: 1, y: 4)

switch point {

case let q1 where (point.x > 0) && (point.y > 0):

    print("\(q1) is in quadrant 1")

case let q2 where (point.x < 0) && point.y > 0:

    print("\(q2) is in quadrant 2")

case let q3 where (point.x < 0) && point.y < 0:

    print("\(q3) is in quadrant 3")

case let q4 where (point.x > 0) && point.y < 0:

    print("\(q4) is in quadrant 4")

case (\_, 0):

    print("\(point) sits on the x-axis")

case (0, \_):

    print("\(point) sits on the y-axis")

default:

    print("Case not covered.")

}