```swift
//
//  ContentView.swift
//  Shared
//
//  Created by Elijah Allen on 9/2/21.
//
import SwiftUI

struct ContentView: View {
    

    func fibonacci(text: String) -> String {
        let num = Int32(text) ?? 0
        
        if num <= 0 {return "0"}
        if num == 1 {return "1"}
        if num > 50 {return "That number is way to big"}
        
        var output: UInt = 0
        var last: UInt   = 1
        var lastlast: UInt = 0
        for _ in 1..<num {
            output = last + lastlast
            lastlast = last
            last = output
            
        }
        return "\(output)"
    }
    
    func factorial(text: String) -> String {
        let num = Int64(text) ?? 0
        
        if num == 0 {return "1.0"}
        
        var ans: Double = 1
        for x in 1...num{
            ans *= Double(x)
        }
        return "\(ans)"
    }
    
    func sum(text: String, other_text: String) -> String {
        let num = Int64(text) ?? 0
        let num_other = Int64(other_text) ?? 0
        
        let big = num < num_other ? num_other : num
        let small = num > num_other ? num_other : num
        
        var output: Int64 = 0
        
        for x in small...big{
            output += x
        }
        
        return "\(output)"
    }
    
    func coinConvert(text: String) -> String {
        func removeCents(cents: Double, remove: Double)-> Double {
            return cents - remove
        }
        func buildString(str: String, type: String, value: Double) -> String{
            if(value <= 0){
                // dont print this
                return str
            }else{
                let fmt_value = String(format: "%g", value)
                if(type == "penney"){
                    return "\(str) \(fmt_value) \(value > 1 ? "pennies" : type)"
                }
                return  "\(str) \(fmt_value) \(type)\(value > 1 ? "s": "")"
            }
        }

        let num: Double = Double(text) ?? 0
        var str = ""
        var cents = num * 100
        let quarter = floor(cents/25)
        cents = removeCents(cents: cents, remove: (quarter*25))
        str = buildString(str: str, type: "quarter", value: quarter)

        let dimes = floor(cents/10)
        cents = removeCents(cents: cents, remove: (dimes*10))
        str = buildString(str: str, type: "dime", value: dimes)

        let nickles = floor(cents/5)
        cents = removeCents(cents: cents, remove: (nickles*5))
        str = buildString(str: str, type: "nickle", value: nickles)

        let pennies = floor(cents/1)
        str = buildString(str: str, type: "penney", value: pennies)

        return str
    }

    var body: some View {
        VStack() {

            ZStack {
                Text("Liddle Rock")
                    // .padding(.vertical)
                    .font(.largeTitle)
                    .foregroundColor(.purple)
                    .bold()
            }.frame( height: 100 ).clipped()




            HStack {
                Text("Testing how to use Swift UI")

            }
            HW2UI(title: "Compute the ith Fibonacci", callback: fibonacci)
            HW2UI(title: "Compute n! for integer n ≥ 0", callback: factorial)
            DoubleInput(title: "Compute the sum of all integers between two given integers (inclusive)", callback: sum)
            HW2UI(title: "Dollars to Cents", callback: coinConvert)
        }
        .padding()


    }
}

struct HW2UI : View {

    @State var text: String = ""
    @State var output: String = ""
    var title: String = ""
    var callback: (String)->String

    var body: some View {
        VStack {
            HStack {
                Text(self.title)
                    .font(.title2)
                Spacer()
            }
            HStack {
                TextField("",text: $text)
                    .border(.purple)
                    .keyboardType(.decimalPad)
                Spacer()
                Button(action: {
                    // Closure will be called once user taps your button
                    self.output = callback(text)
                }) {
                    Text("Submit")
                }.foregroundColor(.purple).padding(.horizontal)
            }
            HStack {
                Text(output).foregroundColor(.purple).bold()

            }
        }.padding()
    }
}

struct DoubleInput : View{
    @State var input_a: String = ""
    @State var input_b: String = ""
    @State var output: String = ""
    var title: String = ""
    var callback: (String, String)->String

    var body : some View{
        VStack{
            Text(self.title)
                .font(.title2)

            HStack{
                TextField("",text: $input_a)
                    .border(.purple)
                    .keyboardType(.decimalPad).frame(width: 60)
                Text("--")
                TextField("",text: $input_b)
                    .border(.purple)
                    .keyboardType(.decimalPad).frame(width: 60)


                Button(action: {
                    // Closure will be called once user taps your button
                    self.output = callback(input_a, input_b)
                }) {
                    Text("Submit")
                }.foregroundColor(.purple).padding(.horizontal)
            }
            Text(output).foregroundColor(.purple).bold()
        }
    }
}

struct ContentView_Previews: PreviewProvider {
    static var previews: some View {
        ContentView()

    }
}
```