```
1 //
 2 //
       ContentView.swift
 3 //
       Shared
 4 //
 5 // Created by Elijah Allen on 9/2/21.
 6 //
 7 import SwiftUI
 8
9 struct ContentView: View {
10
11
       func fibonacci(text: String) -> String {
12
           let num = Int32(text) ?? 0
13
14
15
           if num <= 0 {return "0"}</pre>
           if num == 1 {return "1"}
16
17
           if num > 50 {return "That number is way to big"}
18
19
           var output: UInt = 0
20
           var last: UInt
21
           var lastlast: UInt = 0
22
           for _ in 1..<num {
23
               output = last + lastlast
24
               lastlast = last
25
               last = output
26
27
           return "\(output)"
28
29
       }
30
       func factorial(text: String) -> String {
31
32
           let num = Int64(text) ?? 0
33
34
           if num == 0 {return "1.0"}
35
36
           var ans: Double = 1
37
           for x in 1...num{
38
               ans *= Double(x)
39
           }
           return "\(ans)"
40
41
       }
42
43
       func sum(text: String, other_text: String) -> String {
44
           let num = Int64(text) ?? 0
45
           let num_other = Int64(other_text) ?? 0
46
47
           let big = num < num_other ? num_other : num</pre>
           let small = num > num_other ? num_other : num
48
49
50
           var output: Int64 = 0
51
52
           for x in small...big{
53
               output += x
54
           }
55
56
           return "\(output)"
57
       }
58
```

```
59
        func coinConvert(text: String) -> String {
            func removeCents(cents: Double, remove: Double)-> Double {
 60
                return cents - remove
 61
 62
            func buildString(str: String, type: String, value: Double) -> String{
 63
                if(value <= 0){
 64
                    // dont print this
 65
                    return str
 66
 67
                }else{
 68
                    let fmt value = String(format: "%g", value)
 69
                    if(type == "penney"){
                         return "\(str) \(fmt value) \(value > 1 ? "pennies" : type)"
 70
 71
                           "\(str) \(fmt value) \(type)\(value > 1 ? "s": "")"
 72
 73
                }
 74
            }
 75
 76
            let num: Double = Double(text) ?? 0
            var str = ""
 77
 78
            var cents = num * 100
 79
            let quarter = floor(cents/25)
            cents = removeCents(cents: cents, remove: (quarter*25))
80
            str = buildString(str: str, type: "quarter", value: quarter)
81
 82
 83
            let dimes = floor(cents/10)
 84
            cents = removeCents(cents: cents, remove: (dimes*10))
85
            str = buildString(str: str, type: "dime", value: dimes)
 86
            let nickles = floor(cents/5)
 87
88
            cents = removeCents(cents: cents, remove: (nickles*5))
            str = buildString(str: str, type: "nickle", value: nickles)
 89
 90
 91
            let pennies = floor(cents/1)
92
            str = buildString(str: str, type: "penney", value: pennies)
93
94
            return str
95
        }
 96
97
        var body: some View {
98
            VStack() {
99
                ZStack {
100
                    Text("Liddle Rock")
101
                        // .padding(.vertical)
102
                         .font(.largeTitle)
103
104
                         .foregroundColor(.purple)
105
                         .bold()
                }.frame( height: 100 ).clipped()
106
107
108
109
110
111
                HStack {
112
                    Text("Testing how to use Swift UI")
113
114
                HW2UI(title: "Compute the ith Fibonacci", callback: fibonacci)
115
                HW2UI(title: "Compute n! for integer n ≥ 0", callback: factorial)
116
```

```
117
                DoubleInput(title: "Compute the sum of all integers between two given
    integers (inclusive)", callback: sum)
                HW2UI(title: "Dollars to Cents", callback: coinConvert)
118
119
120
            .padding()
121
122
123
        }
124 }
125
126 struct HW2UI : View {
127
        @State var text: String = ""
128
        @State var output: String = ""
129
        var title: String = ""
130
        var callback: (String)->String
131
132
133
        var body: some View {
            VStack {
134
                HStack {
135
136
                     Text(self.title)
137
                         .font(.title2)
138
                     Spacer()
139
                }
                HStack {
140
                     TextField("",text: $text)
141
142
                         .border(.purple)
                         .keyboardType(.decimalPad)
143
                     Spacer()
144
145
                     Button(action: {
146
                         // Closure will be called once user taps your button
                         self.output = callback(text)
147
                     }) {
148
                         Text("Submit")
149
150
                     }.foregroundColor(.purple).padding(.horizontal)
151
                }
                HStack {
152
                     Text(output).foregroundColor(.purple).bold()
153
154
155
                 }
156
            }.padding()
157
        }
158 }
159
160 struct DoubleInput : View{
        @State var input_a: String = ""
161
        @State var input_b: String = ""
162
        @State var output: String = ""
163
        var title: String = ""
164
165
        var callback: (String, String)->String
166
        var body : some View{
167
            VStack{
168
                Text(self.title)
169
170
                     .font(.title2)
171
                HStack{
172
                     TextField("",text: $input_a)
173
                         .border(.purple)
174
```

```
.keyboardType(.decimalPad).frame(width: 60)
175
                    Text("--")
176
                    TextField("",text: $input b)
177
                         .border(.purple)
178
179
                         .keyboardType(.decimalPad).frame(width: 60)
180
181
                    Button(action: {
182
                        // Closure will be called once user taps your button
183
184
                        self.output = callback(input_a, input_b)
185
                    }) {
                        Text("Submit")
186
                    }.foregroundColor(.purple).padding(.horizontal)
187
188
                Text(output).foregroundColor(.purple).bold()
189
            }
190
        }
191
192 }
193
194 struct ContentView_Previews: PreviewProvider {
        static var previews: some View {
195
            ContentView()
196
197
198
        }
199 }
```