电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

GO 语言与区块链技术
期中设计：计算器

姓名：陈秉辉

学号：2018080901001

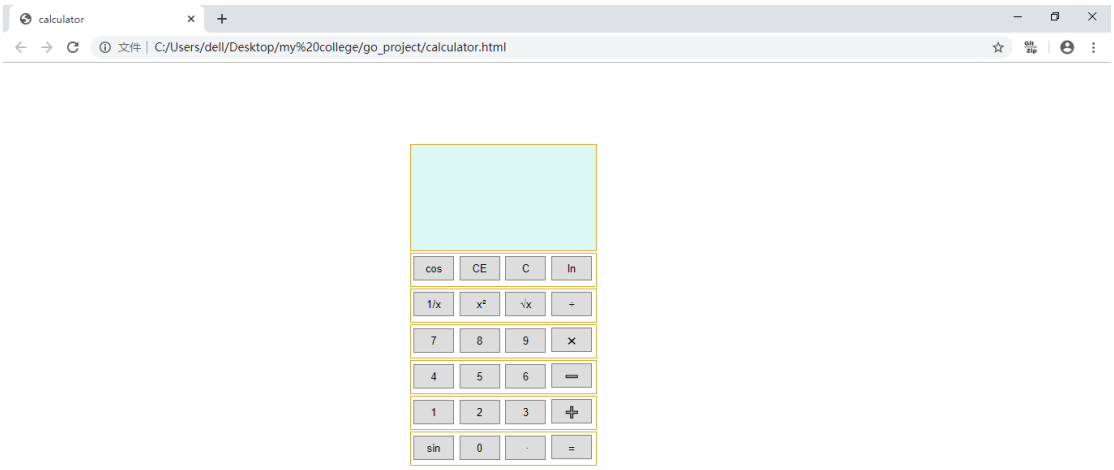
所属学院和班级：计算机学院互联网加班

时间：2019 年 12 月 12 日星期四

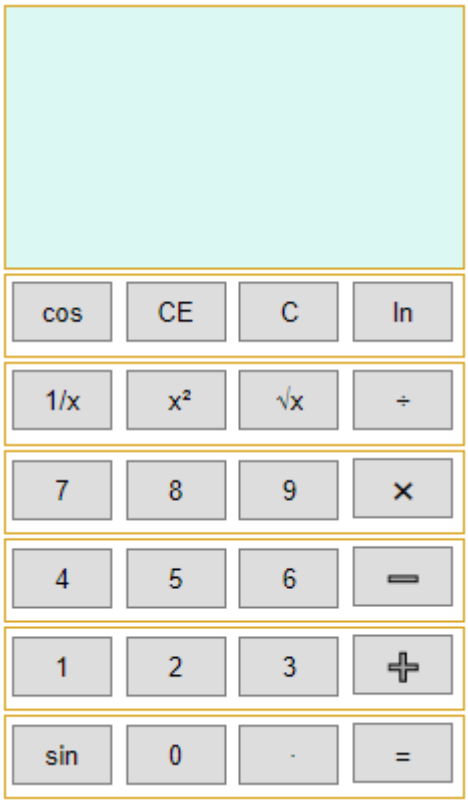
目录

一．结果展示

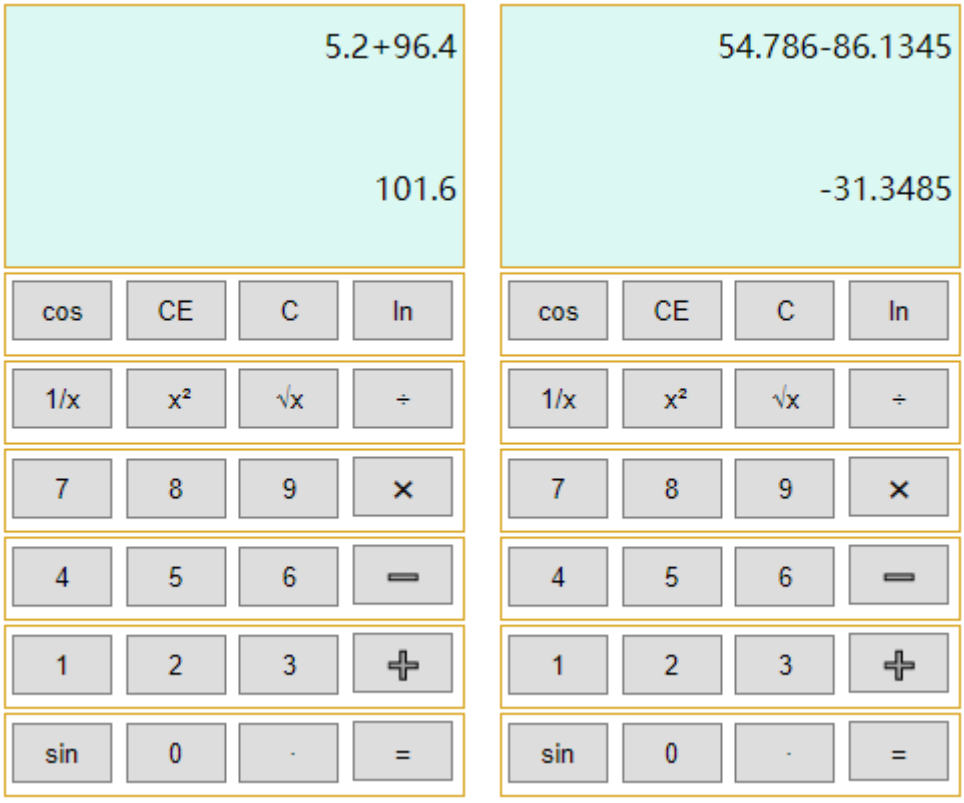
本次作业完成了计算器的部分功能，包括基本的加减乘除，平方，倒数，开方，以及额外增加的两个三角函数 cos，sin 和以自然对数为底的对数函数 ln，并且可以对一个正确的长表达式求值，以及清空和退格等，完成的计算器如下图所示：



具体的计算器界面如下:



加法和减法：

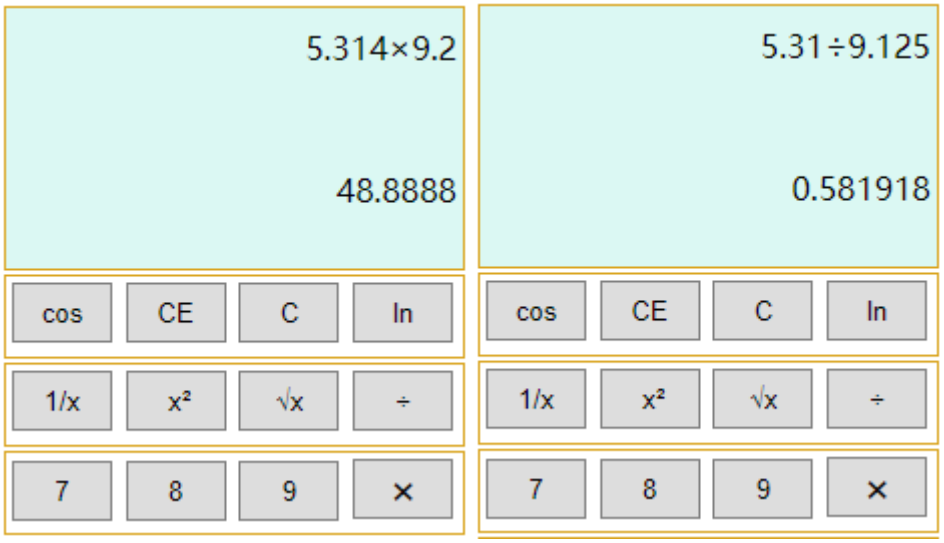



```
后台服务开启...
接收到来自前端的信息：5.2+96.4
转为后缀表达式：5.2 96.4 +
发送信息到前端：101.6
接收到来自前端的信息：54.786-86.1345
转为后缀表达式：54.786 86.1345 -
发送信息到前端：-31.3485
```

后台输出情况

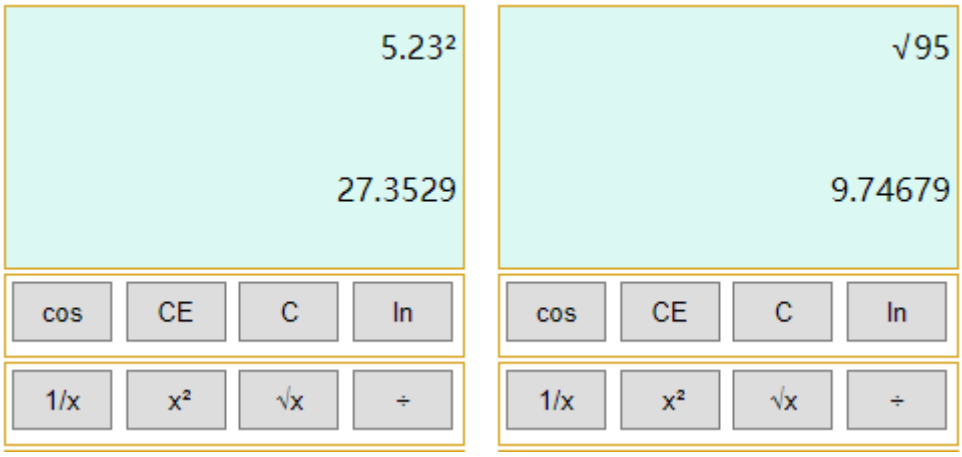
为了节省篇幅，下面开始只截部分

乘法和除法：



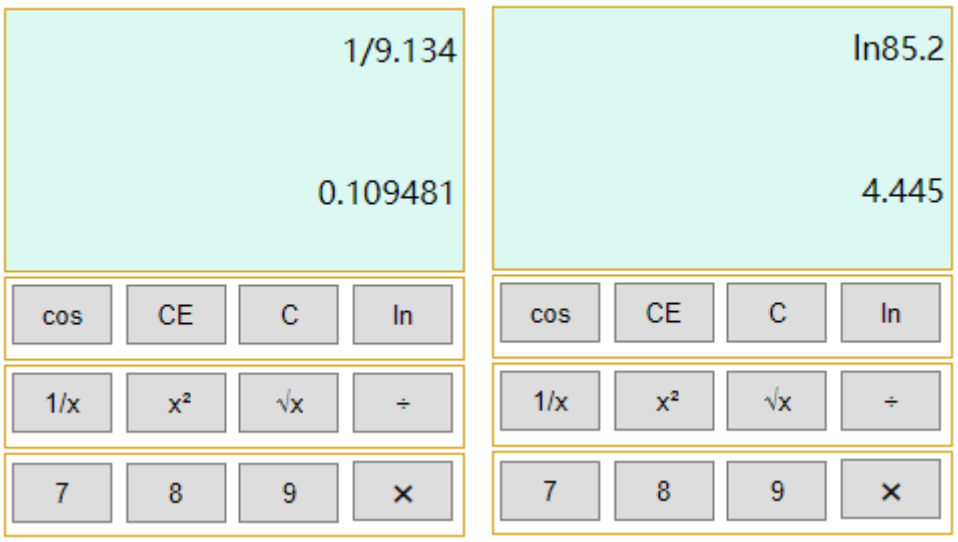

```
接收到来自前端的信息: 5.314m9.2
转为后缀表达式: 5.314 9.2 m
发送信息到前端: 48.8888
接收到来自前端的信息: 5.31d9.125
转为后缀表达式: 5.31 9.125 d
发送信息到前端: 0.581918
```

平方和开方：



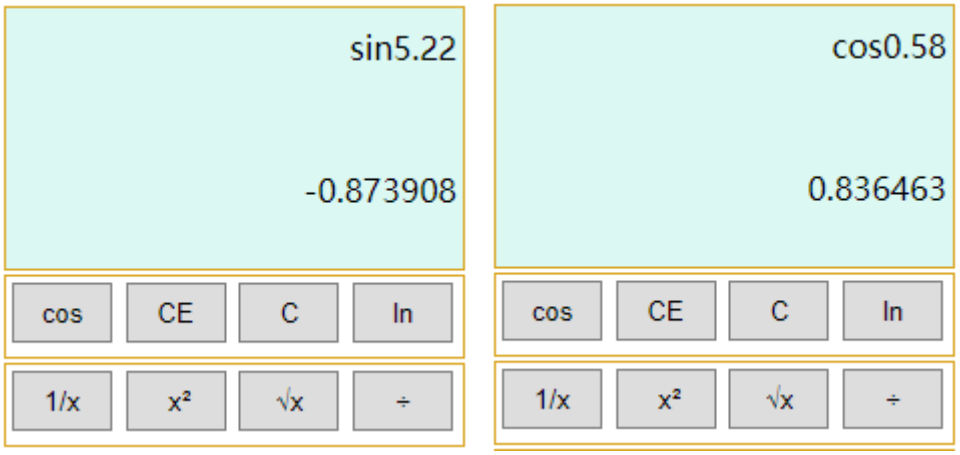

```
接收到来自前端的信息: 5.23p
转为后缀表达式: 5.23 p
发送信息到前端: 27.3529
接收到来自前端的信息: g95
转为后缀表达式: 95 g
发送信息到前端: 9.74679
```

倒数和对数：




```
接收到来自前端的信息: 1/9.134
转为后缀表达式: 1 9.134 /
发送信息到前端: 0.109481
接收到来自前端的信息: ln85.2
转为后缀表达式: 85.2 ln
发送信息到前端: 4.445
```

正弦和余弦：




```
接收到来自前端的信息: sin5.22
转为后缀表达式: 5.22 sin
发送信息到前端: -0.873908
接收到来自前端的信息: cos0.58
转为后缀表达式: 0.58 cos
发送信息到前端: 0.836463
```

长表达式：

由系统自带的计算器可以检验得知计算结果是正确的。

还有清屏和退格功能，不再在此处展示。

二．前端设计

前端是用 html+css+js 写的，主要的思路是：html 里面的计算器有两个显示界面，一个是过程显示界面，即单击等号后将原先输入的表达式在过程显示界面显示出来，在整个计算器的最上方；另外一个是实时显示界面，在用户还在输入表达式的时候实时显示，在用户点击等号之后，显示表达式的结果，除了显示界面，剩下的都是一些按

钮（button 标签），单击某个按钮就会触发单击事件，把按钮的值传给 js 处理，js 会根据不同的按钮在显示界面添加不同的值，在最终输入等号之后，js 会将该完整的表达式传给后台，并等待后台反馈的结果，得到结果后，js 把结果显示在对应的位置上面，css 文件主要是字体、位置、颜色等信息。

三．后台设计

后台主要用到了：正则表达式匹配字符串，计算器的逆波兰算法及其他一些小的数学函数。

主要的思路为：接收来自前端的信息（以字符串形式），然后用正则表达式匹配来将这个字符串分割成一个个操作符及操作数（主要是有些传过来的字符 utf8 编码的，直接对字符串不好操作），然后将这个传过来的中缀表达式转化为逆波兰算法需要的后缀表达式（用栈），具体的转化过程不过多赘述，然后根据这个后缀表达式来计算最终的结果（还是用栈，我在程序里面用数组模拟的栈），再将最终的值以字符串形式发给前端。

四．前端后台的连接

前端后台相连用到的是 websocket（因为这个操作非常简单），只要把他使用文档里面的例程粘下来改一下直接用就好了。

五．总结与心得体会

写这个计算器也花了不少时间吧，之前没学过前端，还花了挺长时间去学习前端的（虽然最后弄出来的界面还是有点丑），然后关于这个前端后台如何连接也花了不少时间，之前考虑过 ajax（但是有点难上手），遂改成了使用 websocket，而对于后台的 go 语言程序，写的也有点磕磕绊绊的（主要还是不够熟悉），但是最后还是完成了这次期中设计，收获也非常多。

六．全部代码

Html：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>calculator</title>
    <link rel="stylesheet" href="calculator.css">
    <script src="calculator.js"></script>
</head>
<body style="margin: 100px 500px;">
    <div id="display">
        <div id="process" value=""></div>
        <div id="result" value=""></div>
    </div>
    <div>
        <form>
            <button type="button" value="cos">cos</button>
            <button type="button" value="ce">CE</button>
            <button type="button" value="c">C</button>
            <button type="button" value="ln"> ln </button>
        </form>
    </div>
    <div>
```

```
        <form>
            <button type="button" value="1/">1/x</button>
            <button type="button" value="²"> x&sup2;</button>
            <button type="button" value="√">&radic;x</button>
            <button type="button" value="÷"> ÷&NonBrea
kingSpace;</button>
        </form>
    </div>
    <div>
        <form>
            <button type="button" value="7">7</button>
            <button type="button" value="8">8</button>
            <button type="button" value="9">9</button>
            <button type="button" value="×"> ✖&NonBre
akingSpace;</button>
        </form>
    </div>
    <div>
        <form>
            <button type="button" value="4">4</button>
            <button type="button" value="5">5</button>
            <button type="button" value="6">6</button>
            <button type="button" value="-">➖</button>
        </form>
    </div>
    <div>
        <form>
            <button type="button" value="1">1</button>
            <button type="button" value="2">2</button>
            <button type="button" value="3">3</button>
            <button type="button" value="+">➕</button>
        </form>
    </div>
    <div>
        <form>
            <button type="button" value="sin">sin</button>
            <button type="button" value="0">0</button>
            <button type="button" value="."> ·</button>
            <button type="button" value="=">  =
  </button>
        </form>
    </div>
</body>
</html>
```

Css:

```
div{
    margin:2px;
    border:1px solid goldenrod;
    padding:2px;
    height:36px;
    width:224px;
}
button{
    margin:1px;
    border:1px solid grey;
    height:30px;
    width:50px;
}
#display{
    background: rgb(219, 248, 243);
    margin:2px;
    border:1px solid goldenrod;
    padding:5px;
    height:120px;
    width:218px;
}
#process{
    background: rgb(219, 248, 243);
    margin:none;
    padding:none;
    height:65px;
    width:216px;
    border:none;
    text-align: right;
}
#result{
    background: rgb(219, 248, 243);
    border:none;
    margin:none;
    padding:none;
    height:45px;
    width:216px;
    text-align: right;
}
```

Js：

```
var sock = null;
var wsurl = "ws://127.0.0.1:1234/websocket";
window.onload = function () {
    console.log("onload");
    sock = new WebSocket(wsurl);
    sock.onopen = function () {
        console.log("connected to " + wsurl);
    }
    sock.onclose = function (e) {
        console.log("connection closed (" + e.code + ")");
    }

    sock.onmessage = function (e) {
        console.log("message received: " + e.data);
        result.innerHTML = e.data
    }
    var str = "";
    var btn = document.getElementsByTagName("button");
    var result = document.getElementById("result");
    var process = document.getElementById("process");
    for (var i = 0; i < btn.length; i++) {
        btn[i].onclick = function () {
            if (result.innerHTML == "" && this.value == ".") {
                result.innerHTML = "0.";
                str = "0.";
            }
            else if (!isNaN(this.value) || this.value == ".") {
                result.innerHTML += this.value;
                str += this.value;
            }
            else {
                if (this.value == "=") {
                    sock.send(str)
                    process.innerHTML = result.innerHTML
                }
                else if (this.value == "ce") {
                    result.innerHTML = result.innerHTML.substr(0, resul
t.innerHTML.length - 1);
                    str = str.substr(0, str.length - 1);
                }
                else if (this.value == "c") {
                    result.innerHTML = "";
```

```
                    process.innerHTML = "";
                    str = "";
                }
                else {
                    result.innerHTML += this.value;
                    if (this.value == "²") {
                        str += "p";
                    }
                    else if (this.value == "√") {
                        str += "g";
                    }
                    else if (this.value == "×") {
                        str += "m";
                    }
                    else if (this.value == "÷") {
                        str += "d";
                    }
                    else {
                        str += this.value;
                    }
                }
            }
        }
    }
};
```

Go 语言后台：

```
package main

import (
    "fmt"
    "log"
    "math"
    "net/http"
    "regexp"
    "strconv"
    "strings"

    "golang.org/x/net/websocket"
)

//后台用逆波兰算法
```

```
//用数组模拟栈
const imaxn int = 10000000
const fmaxn float64 = 1000000

var postfix [100]string
var now int = 0

func stringToNum(str string) (int, float64) {
    if strings.Contains(str, ".") == false {
        i, _ := strconv.ParseInt(str, 10, 64)
        return int(i), fmaxn
    } else {
        i, _ := strconv.ParseFloat(str, 10)
        return imaxn, i
    }
}

func Property(a string) int {
    switch a {
    case "/":
        return 3
    case "ln", "cos", "sin", "g", "p":
        return 2
    case "m", "d":
        return 1
    case "+", "-":
        return 0
    }
    return -1
}

func isOption(a string) bool {
    if a == "sin" || a == "cos" || a == "+" || a == "-
" || a == "m" || a == "d" || a == "p" || a == "g" || a == "/" || a == "
ln" {
        return true
    }
    return false
}

func infixToPostfix(reply string) {
    //用正则表达式匹配数字和符号
    //m 表示✖，d 表示÷，g 表示开根号，p 表示平方
    var stack [100]string
```

```
    count := -1
    r := regexp.MustCompile(`\d+\.?\d*|sin|cos|\+|\-|m|d|p|g|\/|ln`)
    str := r.FindAllString(reply, -1)
    for i := 0; i < len(str); i++ {
        if isOption(str[i]) {
            if count == -1 {
                count++
                stack[count] = str[i]
            } else {
                for {
                    if count == -1 {
                        count++
                        stack[count] = str[i]
                        break
                    }
                    if Property(str[i]) > Property(stack[count]) {
                        count++
                        stack[count] = str[i]
                        break
                    } else {
                        postfix[now] = stack[count]
                        now++
                        count--
                    }
                }
            }
        } else {
            postfix[now] = str[i]
            now++
        }
    }
    for i := count; i >= 0; i-- {
        postfix[now] = stack[i]
        now++
    }
}

func calculate() string {
    var stack [100]string
    count := 0
    flag := 1
    for i := 0; i < now; i++ {
        num11, num12 := stringToNum(stack[count])
```

```
        if postfix[i] == "+" || postfix[i] == "-
" || postfix[i] == "m" || postfix[i] == "d" || postfix[i] == "/" {
            count--
        }
        num21, num22 := stringToNum(stack[count])
        if postfix[i] == "+" {
            if num11 == imaxn {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num12+num22, 'g'
, 6, 64)
                } else {
                    stack[count] = strconv.FormatFloat(num12+float64(nu
m21), 'g', 6, 64)
                }
            } else {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22+float64(nu
m11), 'g', 6, 64)
                } else {
                    stack[count] = strconv.Itoa(num11 + num21)

                }
            }
        } else if postfix[i] == "-" {
            if num11 == imaxn {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22-
num12, 'g', 6, 64)
                } else {
                    stack[count] = strconv.FormatFloat(float64(num21)-
num12, 'g', 6, 64)
                }
            } else {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22-
float64(num11), 'g', 6, 64)
                } else {
                    stack[count] = strconv.Itoa(num21 - num11)
                }
            }
        } else if postfix[i] == "m" {
            if num11 == imaxn {
                if num21 == imaxn {
```

```
                    stack[count] = strconv.FormatFloat(num12*num22, 'g'
, 6, 64)
                } else {
                    stack[count] = strconv.FormatFloat(num12*float64(nu
m21), 'g', 6, 64)
                }
            } else {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22*float64(nu
m11), 'g', 6, 64)
                } else {
                    stack[count] = strconv.Itoa(num11 * num21)
                }
            }
        } else if postfix[i] == "d" || postfix[i] == "/" {
            if num11 == imaxn {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22/num12, 'g'
, 6, 64)
                } else {
                    stack[count] = strconv.FormatFloat(float64(num21)/n
um12, 'g', 6, 64)
                }
            } else {
                if num21 == imaxn {
                    stack[count] = strconv.FormatFloat(num22/float64(nu
m11), 'g', 6, 64)
                } else {
                    stack[count] = strconv.FormatFloat(float64(num21)/f
loat64(num11), 'g', 6, 64)
                }
            }
        } else if postfix[i] == "sin" {
            if num11 == imaxn {
                stack[count] = strconv.FormatFloat(math.Sin(num12), 'g'
, 6, 64)
            } else {
                stack[count] = strconv.FormatFloat(math.Sin(float64(num
11)), 'g', 6, 64)
            }
        } else if postfix[i] == "cos" {
            if num11 == imaxn {
                stack[count] = strconv.FormatFloat(math.Cos(num12), 'g'
, 6, 64)
```

```
            } else {
                stack[count] = strconv.FormatFloat(math.Cos(float64(num
11)), 'g', 6, 64)
            }
        } else if postfix[i] == "g" {
            if num11 == imaxn {
                stack[count] = strconv.FormatFloat(math.Sqrt(num12), 'g
', 6, 64)
            } else {
                stack[count] = strconv.FormatFloat(math.Sqrt(float64(nu
m11)), 'g', 6, 64)
            }
        } else if postfix[i] == "p" {
            if num11 == imaxn {
                stack[count] = strconv.FormatFloat(num12*num12, 'g', 6,
 64)
            } else {
                stack[count] = strconv.Itoa((num11) * num11)
            }
        } else if postfix[i] == "ln" {
            if num11 == imaxn {
                stack[count] = strconv.FormatFloat(math.Log(num12), 'g'
, 6, 64)
            } else {
                stack[count] = strconv.FormatFloat(math.Log(float64(num
11)), 'g', 6, 64)
            }
        } else {
            if flag == 1 {
                flag = 0
            } else {
                count++
            }
            stack[count] = postfix[i]
        }
    }
    return stack[0]
}

func Echo(ws *websocket.Conn) {
    var err error
    for {
        now = 0
        var reply string
```

```
        if err = websocket.Message.Receive(ws, &reply); err != nil {
            fmt.Println("接收失败: ", err)
            break
        }
        fmt.Println("接收到来自前端的信息: " + reply)
        infixToPostfix(reply)
        //处理
        fmt.Printf("转为后缀表达式: ")
        for i := 0; i < now; i++ {
            fmt.Printf("%s ", postfix[i])
        }
        fmt.Println()
        msg := calculate()
        fmt.Println("发送信息到前端: " + msg)
        if err = websocket.Message.Send(ws, msg); err != nil {
            fmt.Println("发送失败: ", err)
            break
        }
    }
}

func main() {
    fmt.Println("后台服务开启...")
    http.Handle("/websocket", websocket.Handler(Echo))
    if err := http.ListenAndServe(":1234", nil); err != nil {
        log.Fatal("ListenAndServe:", err)
    }
}
```