Go lang

Prattya Datta

Task organization

Tasks

Read project descriptio-n	X							
Git Repo	X							
Create files	X							
Install Go	X							
Theoretical part								X
Http status code		X						

Return html			X					
Return xml				X				
Return Json				X				
Return Image					X			
Return plain text						X		
Documentation							X	X

Knowledge learned

Variables

```
var integer1 int = 15

var integer2 int8 = -25

var integer3 int32

var float1 float32 = 63.2

var string1 string = "Hello World!"

var boolean1 bool

var boolean2 bool = true
```

Pointers

```go
package main

import "fmt"

func main() {
    i, j := 42, 2701

    p := &i         // point to i
    fmt.Println(*p) // read i through the pointer
    *p = 21         // set i through the pointer
    fmt.Println(i)  // see the new value of i

    p = &j         // point to j
    *p = *p / 37   // divide j through the pointer
    fmt.Println(j) // see the new value of j
}
```

Functions

```go
package main

import "fmt"

func add(x int, y int) int {
    return x + y
}

func main() {
    fmt.Println(add(42, 13))
}
```

Conditionals

```go
package main

import "fmt"

func main() {

    if 7%2 == 0 {
        fmt.Println("7 is even")
    } else {
        fmt.Println("7 is odd")
    }

    if 8%4 == 0 {
        fmt.Println("8 is divisible by 4")
    }

    if num := 9; num < 0 {
        fmt.Println(num, "is negative")
    } else if num < 10 {
        fmt.Println(num, "has 1 digit")
    } else {
        fmt.Println(num, "has multiple digits")
    }
}
```

Loops

```go
package main

import "fmt"

func main() {
    sum := 0
    for i := 0; i < 10; i++ {
        sum += i
    }
    fmt.Println(sum)
}

```

Arrays and slices

```go
ackage main

import "fmt"

func main() {
    var a [2]string
    a[0] = "Hello"
    a[1] = "World"
    fmt.Println(a[0], a[1])
    fmt.Println(a)

    primes := [6]int{2, 3, 5, 7, 11, 13}
    fmt.Println(primes)
}

```

```go
package main

import "fmt"

func main() {
    primes := [6]int{2, 3, 5, 7, 11, 13}

    var s []int = primes[1:4]
    fmt.Println(s)
}

```

Maps

```go
package main

import "fmt"

func main() {

    m := make(map[string]int)


    m["k1"] = 7
    m["k2"] = 13

    fmt.Println("map:", m)


    v1 := m["k1"]
    fmt.Println("v1: ", v1)

    fmt.Println("len:", len(m))


    delete(m, "k2")
    fmt.Println("map:", m)

    _, prs := m["k2"]
    fmt.Println("prs:", prs)




    n := map[string]int{"foo": 1, "bar": 2}
    fmt.Println("map:", n)
}
```

Structs

```go
package main

import "fmt"

type Vertex struct {
    X int
    Y int
}

func main() {
    fmt.Println(Vertex{1, 2})
}
```

Interfaces

EXPLORER

OPEN EDITORS

interfaces.go src 4

MAIN

bin

pkg

src

interfaces.go 4

interfaces.go

```go
package main

import "fmt"

type Shape interface {
    Area() float64
    Perimeter() float64
}

type Rect struct {
    width  float64
    height float64
}

func (r Rect) Area() float64 {
    return r.width * r.height
}

func (r Rect) Perimeter() float64 {
    return 2 * (r.width + r.height)
}

func main() {
    var s Shape
    s = Rect{5.0, 4.0}
    r := Rect{5.0, 4.0}
    fmt.Printf("type of s is %T\n", s)
    fmt.Printf("value of s is %v\n", s)
    fmt.Println("area of rectange s", s.Area())
    fmt.Println("s == r is", s == r)
}
```

TERMINAL 1: bash

```
main ➜  go run src/interfaces.go
type of s is main.Rect
value of s is {5 4}
area of rectange s 20
s == r is true
main ➜
```

Mistakes

errors.go

```go
package main

import (
    "fmt"
    "time"
)

type MyError struct {
    When time.Time
    What string
}

func (e *MyError) Error() string {
    return fmt.Sprintf("at %v, %s",
        e.When, e.What)
}

func run() error {
    return &MyError{
        time.Now(),
        "it didn't work",
    }
}

func main() {
    if err := run(); err != nil {
        fmt.Println(err)
    }
}

```

Conclusions

Conclusions and recommendations

- GO-lang is very powerful and has ability to optimize codes greatly with its easy use of adding numerous developer functionalities, not possible in may other web development tools like pointers
- However, go still needs lot of development to counter the popularity of javascript