Golang Workshop Exercises

Baiju Muthukadan
Sr. Software Engineer, Red Hat

Exercises

Exercise 1

Write a program to print "Hello, World!" and save this in a file named *helloworld.go*. Compile the program and run it like this: *./helloworld*

Exercise 2

Write a program to print whether the number given as the command line argument is even or odd

Exercise 3

Write a program to print sum of all numbers below 50 completely divisible by 3 or 5 (i.e., remainder 0)

Exercise 4

Write a program to print area and perimeter of a given circle and rectangle. Use *struct* to represent circle and rectangle. Area and Perimeter can be defined as methods. Use *float64* as the type for radius, width & length. The type of shape and dimensions can be read from command line as given here:

```
$ ./shape circle 2
Area: 12.56
Perimeter: 12.56
$ ./shape rectangle 2 3
Area: 6
Perimeter: 10
```

Hint: Use *strconv.ParseFloat* function to covert *string* to *float64*

Exercise 5

Update the previous program (Exercise 4) to use interfaces.

Exercise 6

Rewrite the program in Exercise 3 using goroutine and channel. The task is to print sum of all numbers below 50 completely divisible by 3 or 5 (i.e., remainder 0)

Exercise 7

Write a program to download a list of web pages concurrently using Goroutines.
Hint: Use this tool for serving junk content for testing: https://github.com/baijum/lipsum

Answers

Answer 1

1. Content of *helloworld.go*:

```
package main

import "fmt"

func main() {
        fmt.Println("Hello, World!")
}
```

2. Build program:

```
$ go build helloworld.go
```

3. Run program and verify output like this:

```
$ ./helloworld
Hello, World!
```

Answer 2

1. Content of the file *evenodd.go*:

```
package main

import (
        "fmt"
        "os"
        "strconv"
)

func main() {
        i := os.Args[1]
        n, err := strconv.Atoi(i)
        if err != nil {
                fmt.Println("Not a number:", i)
                os.Exit(1)
        }

        if n%2 == 0 {
                fmt.Println("even number:", n)
        } else {
                fmt.Println("odd number:", n)
        }
}
```

2. Run program and verify output like this:

```
$ go run evenodd.go 2
even number: 2
$ go run evenodd.go 3
odd number: 3
```

Answer 3

1. Content of the file *sum.go*:

```
package main

import "fmt"

func main() {
        sum := 0
        for i := 1; i < 50; i++ {
                if i%3 == 0 {
                        sum = sum + i
                } else {
                        if i%5 == 0 {
                                sum = sum + i
                        }
                }
        }
        fmt.Println(sum)
}
```

2. Run program and verify output like this:

```
$ go run sum.go
543
```

Answer 4

1. Content of the file *shapes.go*:

```
package main

import (
        "fmt"
        "os"
        "strconv"
)

type Rectangle struct {
        length float64
        width  float64
}

type Circle struct {
        radius float64
}

func (r Rectangle) Area() float64 {
        return r.length * r.width
}

func (r Rectangle) Perimeter() float64 {
        return 2 * (r.length + r.width)
}
```

```
func (c Circle) Area() float64 {
        return 3.14 * c.radius * c.radius
}

// Circumference
func (c Circle) Perimeter() float64 {
        return 2 * 3.14 * c.radius
}

func main() {

        shape := os.Args[1]

        if shape == "circle" {

                r := os.Args[2]
                radius, _ := strconv.ParseFloat(r, 64)

                circle := Circle{radius}

                area := circle.Area()
                fmt.Println("Area:", area)

                perimeter := circle.Perimeter()
                fmt.Println("Perimeter:", perimeter)

        } else {
                w := os.Args[2]
                h := os.Args[3]

                width, _ := strconv.ParseFloat(w, 64)
                height, _ := strconv.ParseFloat(h, 64)

                rectangle := Rectangle{width, height}

                area := rectangle.Area()
                fmt.Println("Area:", area)

                perimeter := rectangle.Perimeter()
                fmt.Println("Perimeter:", perimeter)
        }
}
```

2. Run program and verify output like this:

```
$ go run shapes1.go circle 4
Area: 50.24
Perimeter: 25.12
$ go run shapes1.go rectangle 2 3
Area: 4
Perimeter: 8
```

Answer 5

1. Change the previous source file (*shapes.go*) to include the below code:

```
type Geometry interface {
        Area() float64
        Perimeter() float64
}

func Measure(g Geometry) {
        area := g.Area()
        fmt.Println("Area:", area)

        perimeter := g.Perimeter()
        fmt.Println("Perimeter:", perimeter)
}

func main() {

        shape := os.Args[1]

        if shape == "circle" {

                r := os.Args[2]
                radius, _ := strconv.ParseFloat(r, 64)

                circle := Circle{radius}
                Measure(circle)

        } else {
                w := os.Args[2]
                h := os.Args[3]

                width, _ := strconv.ParseFloat(w, 64)
                height, _ := strconv.ParseFloat(h, 64)

                rectangle := Rectangle{width, height}
                Measure(rectangle)
        }
}
```

Answer 6

1. Content of the file *newsum.go*:

```
package main

import "fmt"

func Sum(s chan int) {
        sum := 0
        for i := 1; i < 50; i++ {
                if i%3 == 0 {
                        sum = sum + i
                } else {
                        if i%5 == 0 {
                                sum = sum + i
                        }
                }
```

```
        }
        s <- sum
}

func main() {
        t := make(chan int)
        go Sum(t)
        fmt.Println("Sum:", <-t)
}
```

2. Run program and verify output like this:

```
$ go run sum.go
543
```

Answer 7

1. Content of the file *download.go*:

```
package main

import (
        "io/ioutil"
        "log"
        "net/http"
        "net/url"
        "sync"
)

func main() {
        urls := []string{
                "http://localhost:9999/1.txt",
                "http://localhost:9999/2.txt",
                "http://localhost:9999/3.txt",
                "http://localhost:9999/4.txt",
        }
        var wg sync.WaitGroup
        for _, u := range urls {
                wg.Add(1)
                go func(u string) {
                        defer wg.Done()
                        ul, err := url.Parse(u)
                        fn := ul.Path[1:len(ul.Path)]
                        res, err := http.Get(u)
                        if err != nil {
                                log.Println(err, u)
                        }
                        content, _ := ioutil.ReadAll(res.Body)
                        ioutil.WriteFile(fn, content, 0644)
                        res.Body.Close()
                }(u)
        }
        wg.Wait()
}
```