# DD1339 Introduktion till datalogi 2013/2014

# Uppgift nummer: 7

# Namn: Marcus Larsson

# Grupp nummer: 5

# Övningsledare: Marcus Dicander

# Betyg: ..... Datum: .............. Rättad av: .......................................

# Exercise Loops and Functions

package main

import (

"fmt"

"math"

)

// This method will approximate the square root of a given number.

func Sqrt(x float64) float64 {

z := float64(4)

diff := 1.0

for diff > 0.0001 {

a := z - (math.Pow(z, 2)-x)/(2\*z)

diff = math.Abs(z - a)

// fmt.Println(diff) //this line was to check how many iterations were made and what the diff was.

z = a

}

return z

}

// Prints approximation of sqrt of a number and then prints math.Sqrt result.

func main() {

fmt.Println("Newtons method")

fmt.Println("Sqrt(2): ", Sqrt(2))

fmt.Println("Sqrt(4): ", Sqrt(4))

fmt.Println("Sqrt(5): ", Sqrt(5))

fmt.Println("Sqrt(7): ", Sqrt(7))

fmt.Println("Sqrt(16): ", Sqrt(16))

fmt.Println()

fmt.Println("Go math.Sqrt")

fmt.Println("Sqrt(2): ", math.Sqrt(2))

fmt.Println("Sqrt(4): ", math.Sqrt(4))

fmt.Println("Sqrt(5): ", math.Sqrt(5))

fmt.Println("Sqrt(7): ", math.Sqrt(7))

fmt.Println("Sqrt(16): ", math.Sqrt(16))

}

# Exercise Slices

package main

import "code.google.com/p/go-tour/pic"

// This will generate a two-dimensional slice that indicates the bluescale

// of every pixel in a picture.

func Pic(dx, dy int) [][]uint8 {

a := make([][]uint8, dy, dy)

for i := 0; i < dy; i++ {

row := make([]uint8, dx, dx)

for j := 0; j < dx; j++ {

row[j] = uint8(i \* j)

}

a[i] = row

}

return a

}

func main() {

pic.Show(Pic)

}

# Exercise Maps

package main

import (

"code.google.com/p/go-tour/wc"

"strings"

)

// Counts how many times each word in a string occurs.

// Returns a map of all the unique words and how many times

// it occured in the string

func WordCount(s string) map[string]int {

m := make(map[string]int)

for \_, a := range strings.Fields(s) {

\_, ok := m[a]

if ok {

m[a]++

} else {

m[a] = 1

}

}

return m

}

func main() {

wc.Test(WordCount)

}

# Exercise Fibonacci

package main

import "fmt"

// fibonacci is a function that returns

// a function that returns an int.

func fibonacci() func() int {

curr := 0

prev := 0

return func() int {

if curr == 0 {

curr = 1

return 1

}

res := curr + prev

prev = curr

curr = res

return res

}

}

// print 10 first fibonacci numbers.

func main() {

f := fibonacci()

for i := 0; i < 10; i++ {

fmt.Println(f())

}

}

# Exercise Alert clock

package main

import (

"fmt"

"time"

)

// This method will print the given string with an

// interval of the given time.

func Remind(text string, paus time.Duration) {

for {

fmt.Println("Klockan är", time.Now().Format("15:04"), text)

time.Sleep(paus)

}

}

// Prints three different strings with 3 different intervals.

// Infinite loop, program will never stop.

func main() {

go Remind("Dags att äta", 3\*time.Hour)

go Remind("Dags att arbeta", 8\*time.Hour)

go Remind("Dags att sova", 24\*time.Hour)

select {}

}

# Exercise Sum

package main

import (

"fmt"

)

// Add adds the numbers in a and sends the result on res.

func Add(a []int, res chan<- int) {

sum := 0

for \_, t := range a {

sum += t

}

res <- sum

}

// Calculates the sum of all numbers in an array with parrallell programming.

func main() {

a := []int{1, 2, 3, 4, 5, 6, 7}

n := len(a)

ch := make(chan int)

go Add(a[:n/2], ch)

go Add(a[n/2:], ch)

x, y := <-ch, <-ch

fmt.Println("Sum is:", x+y)

}