SVEUČILIŠTE U ZAGREBU

Fakultet elektrotehnike i računarstva

Programski jezik GO

3. Domaća zadaća

ak. god. 2020/2021

UPUTE:

1. Zadaci se predaju na https://fergo.dev/
2. Prvi put se registrirajte na https://fergo.dev/signup (pazite na email i JMBAG)
 - ubuduće se u sustav prijavljujete s emailom i lozinkom koju ste napisali u ovom koraku
 - sustav nije povezan s ferweb / AAIEdu i sl. sustavima, nemojte koristiti istu lozinku!
 - ako zaboravite lozinku, javite se nekom od nastavnika.
3. Odaberite 2. zadaću i zadatak koji želite predati
 Tab „Zadatak“ sadrži tekst zadatka i kratke upute, u tab „Rješenje“ pišete svoje rješenje.
 (Demo zadatak služi za isprobavanje funkcionalnosti sustava i ne boduje se.)
4. Copy/paste ili natipkajte rješenje zadatka u formular
5. Kliknuti na "Predaja zadatka"
6. Ponoviti korake 3-5 za ostale zadatke

NAPOMENE

- predaje se **cijeli program** (package, imports, funkcija main i funkcija koju trebate dopisati)
- isti zadatak se može predati i **više puta**, pamti se samo **zadnja predaja**
- imena funkcija **MORAJU BITI ISTA** kao u zadatku

1. ZADATAK: Definiran je sljedeći paket (**main.go**):

```go
package someRandomPackage

import (
    "errors"
)

type ShopItem struct {
    Name     string
    Price    int
    Type     string
    Callory  int64
    Quantity int
}

var (
    errToExpensive = errors.New("Too expensive item")
    errWrongItem   = errors.New("Item not allowed")
    errToCaloric   = errors.New("to much calories")
)

const (
    ToExpensiveLimit = 20
    AllowedType      = "Prehrana"
)

func TotalCost(items []ShopItem) (int, error) {
    total := 0
    for _, v := range items {
        if v.Price*v.Quantity > 20 {
            return 0, errToExpensive
        }
        if v.Type != AllowedType {
            return 0, errWrongItem
        }
        if v.Callory > 300 {
            return 0, errToCaloric
        }
        total += v.Price * v.Quantity
    }
    return total, nil
}
```

Potrebno je napisati testove (main_test.go) koji pokrivaju sve moguće varijacije koje se mogu dogoditi kao rezultat izvršavanja

NAPOMENA: U ovom slučaju ne predaje se program nego test datoteka! test (ili testovi) trebaju biti napisani u istoj datoteci (samo jedna se predaje)

2. ZADATAK: Potrebno je nadopuniti zadani program, tj. napisati funkcije countTriangles i countPolygons.

Rješenje predajete u sljedećem formatu:

```
package main

import "fmt"

func main() {
    shapes := []GeometricShape{
        &Dot{},
        &Henagon{},
        &Digon{},
        &RightTriangle{},
        &RightTriangle{},
        &EquilateralTriangle{},
    }
    fmt.Println(countTriangles(shapes)) // 3
    fmt.Println(countPolygons(shapes))  // 5
}

func countTriangles(shapes []GeometricShape) int {
    // tijelo funkcije
}

func countPolygons(shapes []GeometricShape) int {
    // tijelo funkcije
}
```

NAPOMENE:
Predaje se cijeli program, tj. cijeli sadržaj gornjeg okvira (definicije paketa, importi, funkcija main, funkcija countTriangles, countPolygons)

3. ZADATAK: Potrebno je nadopuniti zadani program, tj. dopuniti **homework.go** sa svim nedostajućim funkcijama

Rješenje predajete u sljedećem formatu:

```go
package main

import "fergo.dev/H9/shapes"

type Octagon struct{}

// dopisati funkcije za Octagon da bi se program mogao pokrenuti

func countSides(shapes []shapes.GeometricShape) int {
  // tijelo funkcije
}
```

NAPOMENA: Predaje se cijela datoteka **homework.go**

PRIMJER:
navedeni main iz zadatka treba ispisati **20**.

4. ZADATAK: Potrebno je nadopuniti zadani program, tj. napisati funkciju fields koja za dobivenu strukturu vraća samo exportana polja.

Rješenje predajete u sljedećem formatu:

```
package main

import (
    "fmt"
    "reflect"
    "unicode"
)

type Point struct {
    X       int
    Y       int
    private string
}

func main() {
    fmt.Println(fields(Point{})) // [X Y]
}

func fields(t interface{}) []string {
    // tijelo funkcije
}
```

NAPOMENA: Predaje se cijeli program, tj. cijeli sadržaj gornjeg okvira (definicije paketa, importi, funkcija main, funkcija fields).

PRIMJER:
za Point iz zadatka programa funkcija fields vraća [X Y].