UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
KATEDRA ZA PRIMENJENE RAČUNARSKE NAUKE

Paralelni i distribuirani algoritmi i strukture podataka

ms Nebojša Horvat

Zimski semestar 2019/2020.

Studijski program: Računarstvo i automatika

Modul: Računarstvo visokih performansi

Go (Golang)

Golang

- Go je programski jezik razvijen od strane Google-a
- 2009 godine, Robert Griesemer, Rob Pike, and Ken Thompson
- Otvorenog koda
- Veliki broj sličnosti sa jezikom C
- Odlične performanse
  - Zauzimanje promenljivih I nizova
  - Da bi se izbegao overhead pri pozivanju funkcija go radi Inlining
  - U c promenljiva ide na heap (malloc) ili stack (lokalna promenljiva) dok go proverava da li referenca promenljive izlazi iz okvira funkcije pa onda određuje gde je čuva

Konkurentan pristup – CSP (Communicating sequential processes)

Hello World from Go

- Deklaracija paketa
- Import naredbe

```
package main

import "fmt"

func main() {
    fmt.Println("Hello World from Go")
}
```

- Ulazna tačka u program – main funkcija.
- Standardni ulaz/izlaz - fmt paket

Uvodne napomene

- Ne postoji null vrednost, umesto toga je nil
- Promenljive koje su deklarisane se moraju koristiti
  - Rezultovaće greškom ako se ne koriste
- Uvučeni (import-ovani) paketi se moraju koristiti
  - Rezultovaće greškom ako se ne koriste
- Ne postoji ; na kraju iskaza
  - Za razliku od C, C++, C#, Java
- GOPATH
- Vežbanje online
  - https://tour.golang.org/

Standardni ulaz/izlaz

- Paket fmt
- Rad sa standardnim ulazom/izlazom sličan kao u jeziku C

```
fmt.Scanf("%d", &n)

fmt.Printf(“Number is %d \n ", n)

fmt.Println(“Some text”)
```

Prosti tipovi podataka

- bool
- string
- int int8 int16 int32 int64
- uint uint8 uint16 uint32 uint64 uintptr
- byte = uint8
- float32 float64
- complex64 complex128

Deklaracija promenljivih

- var a int
  - Neinicijalizovana int promenljiva ima vrednost 0
- var a int = 5
- var a, b int = 5, 6

- Deklaracija i inicijalizacija
  - s := “Hello World“
  - s := 5

Operatori

Aritmetički	Logički	Relacijski
+	&&	==
-	||	!=
*	!	<
/		<=
%		>
&		>=
|		
^		
>>		
<<		

Naredbe grananja

- if
- else if
- else
- Ne postoji ternari operator
  - var evenOrOdd = a % 2 == 0 ? "Even":"Odd";

Naredbe grananja

- Nisu potrebne zagrade u if izrazu
- } mora da bude u istom redu kao i else

```
if n % 2 == 0 {
        fmt.Printf(“%d is an even number”,n)
    } else {
    fmt.Printf(“%d is an odd number”,n)
 }
```

Naredbe grananja - Switch

```
fmt.Print("Go runs on ")
    //os:=runtime.GOOS; se izvrši neposredno pre switch dela
    switch os := runtime.GOOS; os {
    case "darwin":
        fmt.Println("OS X.")
    case "linux":
        fmt.Println("Linux.")
    default:
        fmt.Printf("%s.", os)
    }
```

- Nije potreban break unutar case-a
- Case izrazi ne moraju da budu konstante

Zadatak – Kalkulator

- Programu se prosleđuje operacija i 2 cela broja.
- Operacija može biti
  - “PLUS”,
  - “MINUS”,
  - “MULTIPLY”,
  - “DIVIDE”,
  - “REMAINDER”
- Na osnovu prosleđene operacije, izvšiti potrebnu aritmetičku operaciju i vratiti njen rezultat
  - Sprečiti potencijalne greške

Naredbe ciklusa – For

- Nema zagrada kod izraza unutar for klauzule
- for petlja

```
for i := 0; i < 10; i++ {
        sum += i
    }
```

- Beskonačna for petlja

```
for {
}
```

Naredbe ciklusa

- Ne postoji while petlja
  - Može se postići isti efekat for petljom:

```
for sum < 1000 {
    sum += sum
}
```

Zadatak

- Napisati program koji ispisuje prvih 25 prostih brojeva
  - Kada program bude logički dobro radio
    - Kako optimizovati algoritam?

Komentari

- // Komentarisanje jedne linije kao u C,C++,C#,Java

/* Komentarisanje više linija koda (multiline comment) je takođe standardno (C, C#, Java)
*/

Funkcije

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

Povratna vrednost

Argumenti funkcije

Funkcije

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

- Kog tipa su x i y ?

Funkcije

```
func add(x int, y int) int {
    return x + y
}
func main() {
    var result = add(42, 13)
    var resultPlus5 = result + 5
    fmt.Printf("Result is %d, %d", result,
resultPlus5)
}
```

Primer – faktorijel

fact(5) = 5 * 4 * 3 * 2 *1

- Faktorijel nad negativnim brojevima ?
- 0! = ?

```
func fact(n int) int {
    //} in the same line as else
    if n < 1 {
        return 1
    } else {
        return n * fact(n-1)
    }

}
```

Primer – prost broj pomoću funkcije

- Napisati funkciju koja proverava da li je broj prost
- Nije potrebno proveravati delioce do n
  - već do Sqrt(n)

```
func isPrime(n int) bool {
    //6k + 1 || 6k -1 except 2 and 3
    if n < 2 {
        return false
    }
    for i := 2; i < n; i++ {
        if n%i == 0 {
            return false
        }
    }
    return true
}
```

Zadatak

- Napisati funkciju koja izračunava n-ti član Fibonačijevog niza
  - Deklaracija funkcije je:
    - func fib (n int) int

Zadatak

- Uraditi prethodni zadatak korišćenjem rekurzije

Zadatak – Levi faktorijel

- Levi faktorijel (https://sh.wikipedia.org/wiki/Levi_faktorijel)

- !n = 0! + 1! +2! +...+ (n-1)!
- Napisati funkciju koja računa levi faktorijel od n pozivajući funkciju koja računa desni faktorijel

Zadatak – Levi faktorijel dokaz

- Za levi i desni faktorijel važi da je najveći zajednički delilac jednako 2
  - Napisati program koji će proveriti ovo za prvih 12 prirodnih brojeva