PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ FACULTAD DE CIENCIAS E INGENIERÍA

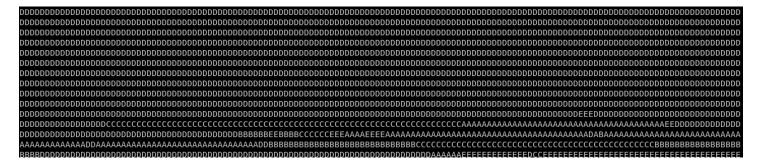
INF239 SISTEMAS OPERATIVOS

Semestre 2023-1 Laboratorio 2

1) (12 puntos – nombre del programa: *Preg1.go*) Se tiene el siguiente programa en Go

```
package main
    import (
"fmt"
 5
             "os"
             "sync"
 8
 9
    var wg sync.WaitGroup
11
    func admin() {
12
            // Aqui va el código del administrador
13
14
15
    func worker1() {
16
             for {
                     fmt.Printf("A")
17
18
19
             wg.Done()
2θ }
21
    func worker2() {
22
23
             for {
24
                     fmt.Printf("B")
25
26
             wg.Done()
27
    }
28
    func worker3() {
29
30
             for {
31
                      fmt.Printf("C")
32
33
             wg.Done()
34
    }
35
36
    func worker4() {
37
38
                     fmt.Printf("D")
39
4θ
             wg.Done()
41
   }
42
43
    func worker5() {
44
             for {
45
                     fmt.Printf("E")
46
47
             wg.Done()
48 }
49
    func main() {
5θ
            for i := 1; i < len(os.Args); i++ {
    cadena = os.Args[i]</pre>
51
52
53
                     switch cadena {
54
55
                     case "A":
56
                              fmt.Println(cadena)
57
                     case "B":
58
                               fmt.Println(cadena)
59
                     case "C"
60
                               fmt.Println(cadena)
61
                     case "D":
62
                               fmt.Println(cadena)
63
                     case "E":
                              fmt.Println(cadena)
64
65
66
             wg.Add(5)
67
68
             go admin()
69
             go worker1()
70
71
             go worker2()
             go worker3()
             go worker4()
72
73
             go worker5()
74
             wg.Wait()
75
```

Que al ejecutarse, proporciona la siguiente salida:



Se le solicita sincronizar las *gorutinas* haciendo uso de canales, para que dada una secuencia ingresada por línea de comandos, se imprima de forma infinita la secuencia. Por ejemplo:

\$./Preg1 A B C D E

Se debe obtener la siguiente salida

Pero si se escribe otra secuencia, como:

\$./Preq1 A A B C D E E

Se debe obtener la siguiente salida



La solución que debe plantear es la siguiente:

- 1) La gorutina principal debe leer de la línea de comandos los valores y guardarlos en un buffered channel.
- 2) La gorutina admin debe leer del buffered channel y sincronizar el respectivo iésimo worker

3) (8 puntos – nombre del programa: *Preg2.go*) Se tiene el siguiente programa en Go:

```
package main
    import "fmt"
5
6
    func main() {
        ch := make(chan int)
7
        done := make(chan bool)
        go producer(ch, done)
        go consumer(ch)
10
        <-done
11
12
13
    func producer(ch chan int, done chan bool) {
        for index := 0; index < 10; index++ {</pre>
14
             fmt.Printf("PRODUCTOR: envia %v\n", index)
15
16
             ch <- index
17
18
        close(ch)
19
        done <- true
20
21
22
   func consumer(ch chan int) {
        for val := range ch {
24
             fmt.Printf("CONSUMIDOR: lee %v\n", val)
```

Que implementa el problema del productor y consumidor usando canales y cuya salida es:

```
PRODUCTOR: envia 0
PRODUCTOR: envia 1
CONSUMIDOR: lee 0
CONSUMIDOR: lee 1
PRODUCTOR: envia 2
PRODUCTOR: envia 3
CONSUMIDOR: lee 2
CONSUMIDOR: lee 3
PRODUCTOR: envia 4
PRODUCTOR: envia 4
PRODUCTOR: envia 5
CONSUMIDOR: lee 6
CONSUMIDOR: lee 5
PRODUCTOR: envia 6
PRODUCTOR: envia 6
PRODUCTOR: envia 7
CONSUMIDOR: lee 6
CONSUMIDOR: lee 6
CONSUMIDOR: lee 7
PRODUCTOR: envia 8
PRODUCTOR: envia 9
CONSUMIDOR: lee 8
CONSUMIDOR: lee 8
```

- a) (3 puntos) En el mismo archivo fuente, indique la secuencia de ejecución de las líneas para obtener la salida mostrada.
- b) (5 puntos) Modifique el programa para que cumpla las siguientes condiciones:
 - 1) 3 productores
 - 2) 2 consumidores
 - 3) Emplee buffered channel con capacidad para 100 elementos.

Porf. Alejandro T. Bello Ruiz