

SISTEME DISTRIBUITE

LABORAOTR 2 – Canale pentru conectări concurente

Săptămâna 18.10.2021 20.10.2021

Dr. Marius Iulian Mihailescu

marius-iulian.mihailescu@g.unibuc.ro

1. Să se implementeze o aplicație care să transfere cu succes un mesaj (ex.: salut) de la o rutină *go* (*goroutine*) la alta prin intermediul unui canal de comunicare.
2. Să se creeze o aplicație care primește două mesaje pe același canal de comunicare.
3. Creați o aplicație care să notifice o altă rutină *go* că procesarea unei funcții a fost efectuată cu succes.
4. Implementați o aplicație care să folosească un canal pentru primirea datelor și altul pentru trimiterea datelor.
5. Implementați o aplicație care să folosească instrucțiunea *select* cu scopul de a combina două rutine *go* (*goroutines*) pentru trimiterea de două sau mai multe mesaje.
6. Să se implementeze o aplicație în care presupunem că executăm un apel extern care returnează rezultatul său pe canalul 1 după 2 secunde.

Rezolvările se află pe paginile următoare.

Problema 1

```
package main

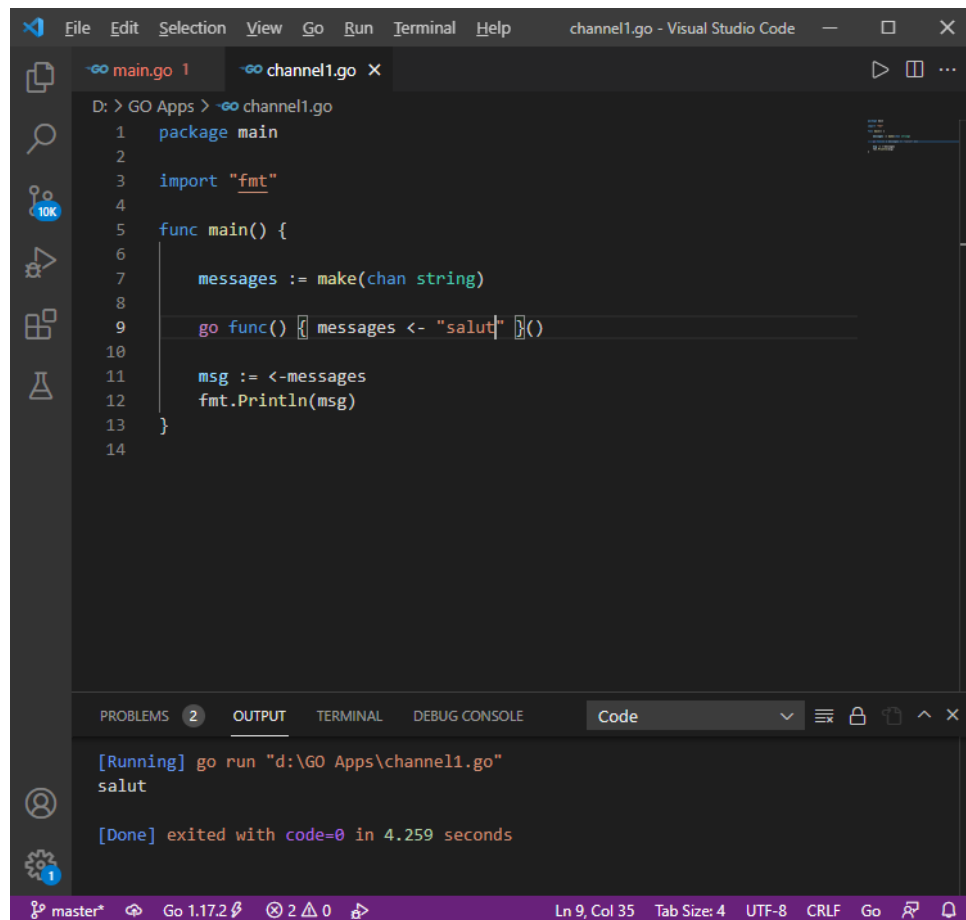
import "fmt"

func main() {

    messages := make(chan string)

    go func() { messages <- "salut" }()

    msg := <-messages
    fmt.Println(msg)
}
```



The screenshot shows the Visual Studio Code interface with a Go file named `channel1.go` open. The code in the editor is identical to the one in the previous block. Below the editor, the `OUTPUT` panel is active, showing the command `go run "d:\GO Apps\channel1.go"` and its output `salut`. Below the output, it says `[Done] exited with code=0 in 4.259 seconds`. The status bar at the bottom indicates the file is on line 9, column 35, and the Go version is 1.17.2.

```
File Edit Selection View Go Run Terminal Help channel1.go - Visual Studio Code
main.go 1 channel1.go x
D: > GO Apps > channel1.go
1 package main
2
3 import "fmt"
4
5 func main() {
6
7     messages := make(chan string)
8
9     go func() { messages <- "salut" }()
10
11     msg := <-messages
12     fmt.Println(msg)
13 }
14

PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE Code
[Running] go run "d:\GO Apps\channel1.go"
salut

[Done] exited with code=0 in 4.259 seconds
master* Go 1.17.2 2 0 Ln 9, Col 35 Tab Size: 4 UTF-8 CRLF Go
```

Problema 2

```
package main

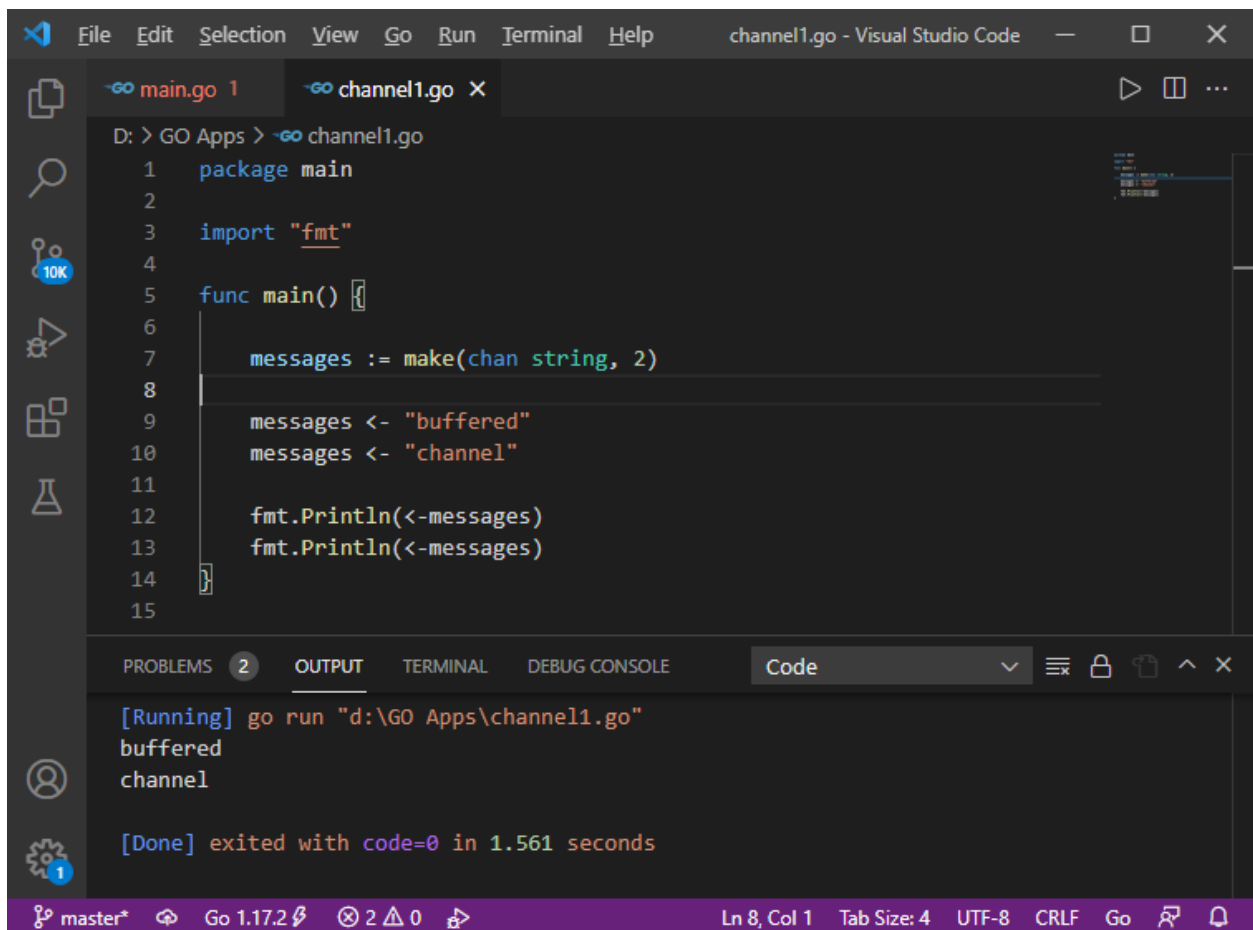
import "fmt"

func main() {

    messages := make(chan string, 2)

    messages <- "buffered"
    messages <- "channel"

    fmt.Println(<-messages)
    fmt.Println(<-messages)
}
```



The screenshot shows the Visual Studio Code interface with a Go file named `channel1.go` open. The code in the editor is identical to the one in the first block. The bottom panel shows the **OUTPUT** window with the following text:

```
[Running] go run "d:\GO Apps\channel1.go"
buffered
channel

[Done] exited with code=0 in 1.561 seconds
```

The status bar at the bottom indicates the file is on the `master` branch, the Go version is `1.17.2`, and the cursor is at `Ln 8, Col 1`.

Problema 3

```
package main

import (
    "fmt"
    "time"
)

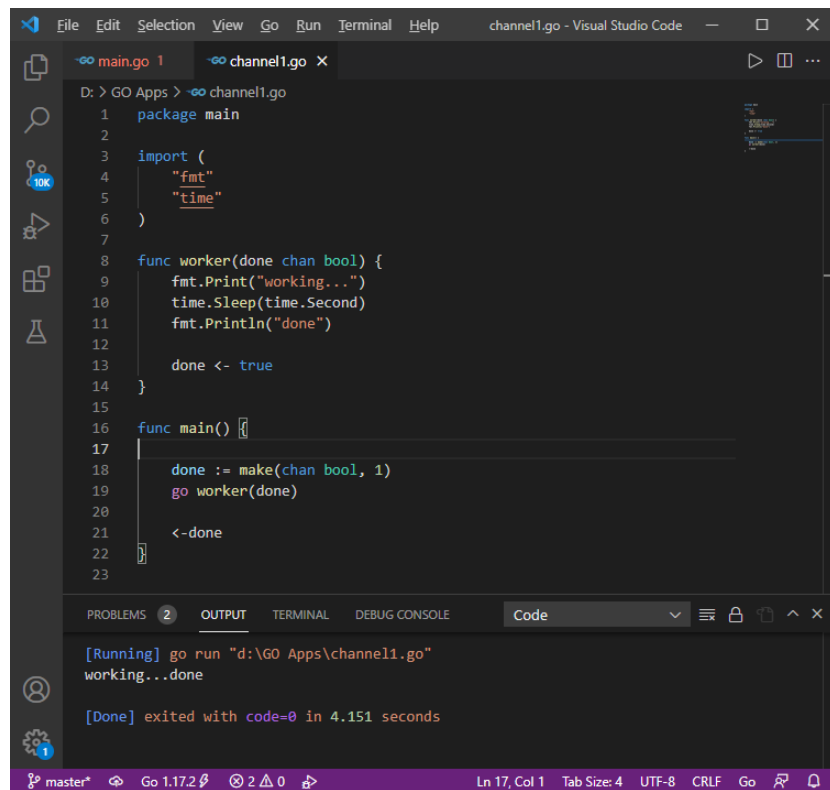
func worker(done chan bool) {
    fmt.Print("working...")
    time.Sleep(time.Second)
    fmt.Println("done")

    done <- true
}

func main() {

    done := make(chan bool, 1)
    go worker(done)

    <-done
}
```



The screenshot shows the Visual Studio Code editor with a Go file named `channel1.go`. The code is identical to the one in the first block. The interface includes a sidebar with icons for Explorer, Search, Run and Debug, Extensions, and Testing. The main editor area shows the code with line numbers from 1 to 23. The bottom panel is divided into 'PROBLEMS' (showing 2 items), 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The 'OUTPUT' tab is active, displaying the execution output: `[Running] go run "d:\GO Apps\channel1.go"`, `working...done`, and `[Done] exited with code=0 in 4.151 seconds`. The status bar at the bottom indicates the file is on line 17, column 1, with a tab size of 4, using UTF-8 encoding and CRLF line endings.

```
File Edit Selection View Go Run Terminal Help channel1.go - Visual Studio Code
main.go 1 channel1.go x
D:\> GO Apps > channel1.go
1 package main
2
3 import (
4     "fmt"
5     "time"
6 )
7
8 func worker(done chan bool) {
9     fmt.Print("working...")
10    time.Sleep(time.Second)
11    fmt.Println("done")
12
13    done <- true
14 }
15
16 func main() {
17
18     done := make(chan bool, 1)
19     go worker(done)
20
21     <-done
22 }
23

PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE Code
[Running] go run "d:\GO Apps\channel1.go"
working...done
[Done] exited with code=0 in 4.151 seconds
master Go 1.17.2 2 0 Ln 17, Col 1 Tab Size: 4 UTF-8 CRLF Go
```

Problema 4

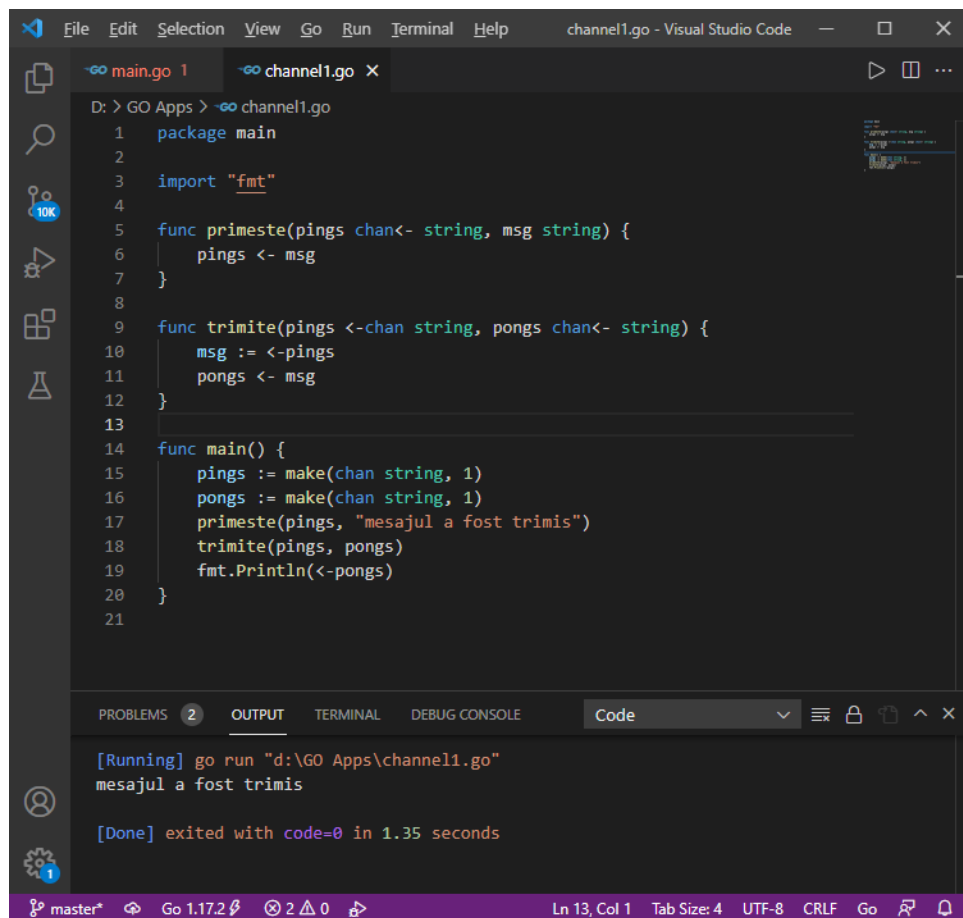
```
package main

import "fmt"

func primeste(pings chan<- string, msg string) {
    pings <- msg
}

func trimite(pings <-chan string, pongs chan<- string) {
    msg := <-pings
    pongs <- msg
}

func main() {
    pings := make(chan string, 1)
    pongs := make(chan string, 1)
    primeste(pings, "mesajul a fost trimis")
    trimite(pings, pongs)
    fmt.Println(<-pongs)
}
```



The screenshot shows the Visual Studio Code interface with two tabs open: `main.go` and `channel1.go`. The `main.go` tab is active, displaying the following Go code:

```
1 package main
2
3 import "fmt"
4
5 func primeste(pings chan<- string, msg string) {
6     pings <- msg
7 }
8
9 func trimite(pings <-chan string, pongs chan<- string) {
10     msg := <-pings
11     pongs <- msg
12 }
13
14 func main() {
15     pings := make(chan string, 1)
16     pongs := make(chan string, 1)
17     primeste(pings, "mesajul a fost trimis")
18     trimite(pings, pongs)
19     fmt.Println(<-pongs)
20 }
21
```

The bottom panel shows the **OUTPUT** window with the following text:

```
[Running] go run "d:\GO Apps\channel1.go"
mesajul a fost trimis

[Done] exited with code=0 in 1.35 seconds
```

The status bar at the bottom indicates the current file is `main.go`, the Go version is `1.17.2`, and the editor is at `Ln 13, Col 1`.

Problema 5

```
package main

import (
    "fmt"
    "time"
)

func main() {

    canal1 := make(chan string)
    canal2 := make(chan string)

    go func() {
        time.Sleep(1 * time.Second)
        canal1 <- "primul mesaj"
    }()
    go func() {
        time.Sleep(2 * time.Second)
        canal2 <- "al doilea mesaj"
    }()

    for i := 0; i < 2; i++ {
        select {
            case mesaj1 := <-canal1:
                fmt.Println("am primit", mesaj1)
            case mesaj2 := <-canal2:
                fmt.Println("am primit", mesaj2)
        }
    }
}
```

FileEditSelectionViewGoRunTerminalHelpchannel1.go - Visual Studio Code

channel1.go X

D: > GO Apps > -go channel1.go

```
1 package main
2
3 import (
4     "fmt"
5     "time"
6 )
7
8 func main() {
9
10     canal1 := make(chan string)
11     canal2 := make(chan string)
12
13     go func() {
14         time.Sleep(1 * time.Second)
15         canal1 <- "primul mesaj"
16     }()
17     go func() {
18         time.Sleep(2 * time.Second)
19         canal2 <- "al doilea mesaj"
20     }()
21
22     for i := 0; i < 2; i++ {
23         select {
24             case mesaj1 := <-canal1:
25                 fmt.Println("am primit", mesaj1)
26             case mesaj2 := <-canal2:
27                 fmt.Println("am primit", mesaj2)
28         }
29     }
30 }
31
```

PROBLEMS 2

OUTPUT

TERMINAL

DEBUG CONSOLE

Code

[Running] go run "d:\GO Apps\channel1.go"

am primit primul mesaj

am primit al doilea mesaj

[Done] exited with code=0 in 3.988 seconds

master* Go 1.17.2 2 0 Ln 12, Col 1 Tab Size: 4 UTF-8 CRLF Go

Problema 6

```
package main

import (
    "fmt"
    "time"
)

func main() {

    canal1 := make(chan string, 1)
    go func() {
        time.Sleep(2 * time.Second)
        canal1 <- "rezultatul 1"
    }()

    select {
    case result := <-canal1:
        fmt.Println(result)
    case <-time.After(1 * time.Second):
        fmt.Println("timeout 1")
    }

    canal2 := make(chan string, 1)
    go func() {
        time.Sleep(2 * time.Second)
        canal2 <- "rezultatul 2"
    }()
    select {
    case res := <-canal2:
        fmt.Println(res)
    case <-time.After(3 * time.Second):
        fmt.Println("timeout 2")
    }
}
```


FileEditSelectionViewGoRunTerminalHelpchannel1.go - Visual Studio Code

channel1.go X

D:\> GO Apps > -GO channel1.go

```
4      "fmt"
5      "time"
6  )
7
8  func main() {
9
10     canal1 := make(chan string, 1)
11     go func() {
12         time.Sleep(2 * time.Second)
13         canal1 <- "rezultatul 1"
14     }()
15
16     select {
17     case result := <-canal1:
18         fmt.Println(result)
19     case <-time.After(1 * time.Second):
20         fmt.Println("timeout 1")
21     }
22
23     canal2 := make(chan string, 1)
24     go func() {
25         time.Sleep(2 * time.Second)
26         canal2 <- "rezultatul 2"
27     }()
28     select {
29     case res := <-canal2:
30         fmt.Println(res)
31     case <-time.After(3 * time.Second):
32         fmt.Println("timeout 2")
33     }
34 }
35
```

PROBLEMS 2

OUTPUT

TERMINAL

DEBUG CONSOLE

Code

```
[Running] go run "d:\GO Apps\channel1.go"
timeout 1
rezultatul 2

[Done] exited with code=0 in 6.371 seconds
```

master*

Go 1.17.2

2

Ln 16, Col 13

Tab Size: 4

UTF-8

CRLF

Go