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 sa transfere cu success 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 success.

4. Implementați o aplicație care să folosească un canal pentru primirea datelor și altul pentru trimiterea datelor.

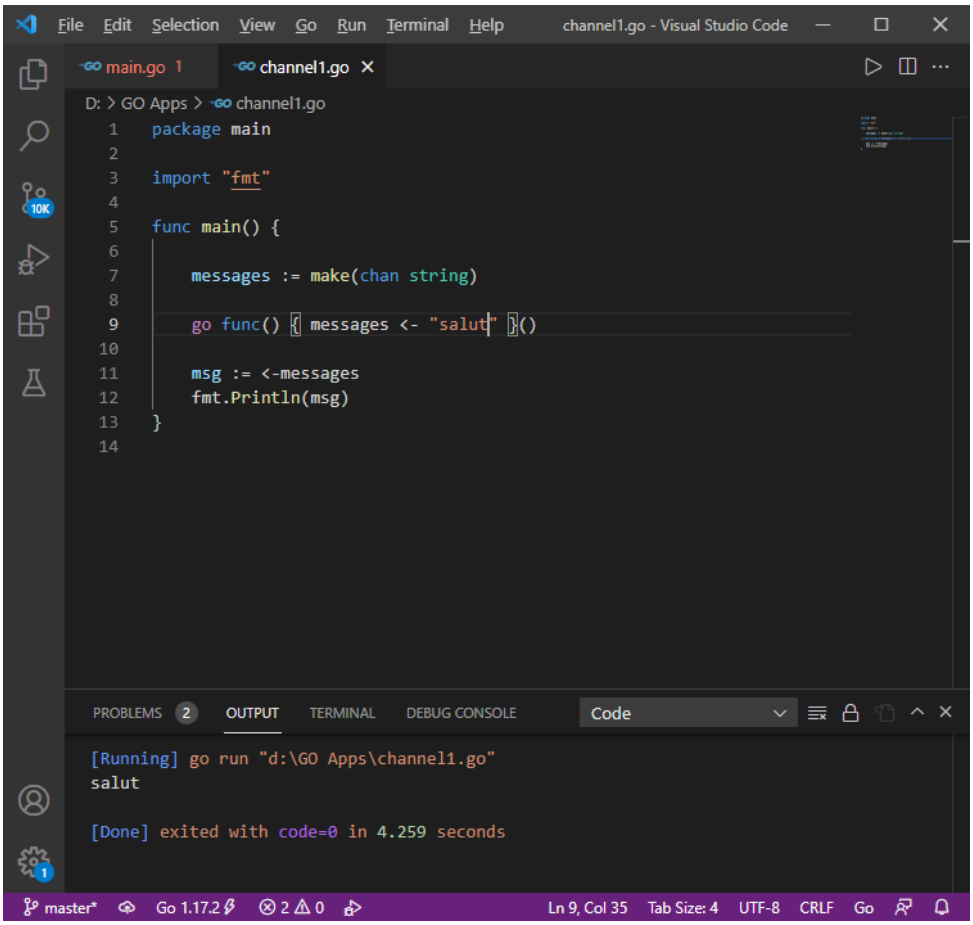
5. Implementați o aplicație care să folosească instrucțiunea *select* cu scopul de a combina două rutine *go* (*goroutines*) pentru trimiterea de două sau mai multe mesaje.

6. Să se implementeze o aplicație în care presupunem că executăm un apel extern care turnează rezultatul său pe canalul 1după 2 secunde.

Rezolvările se află pe paginile următoare.

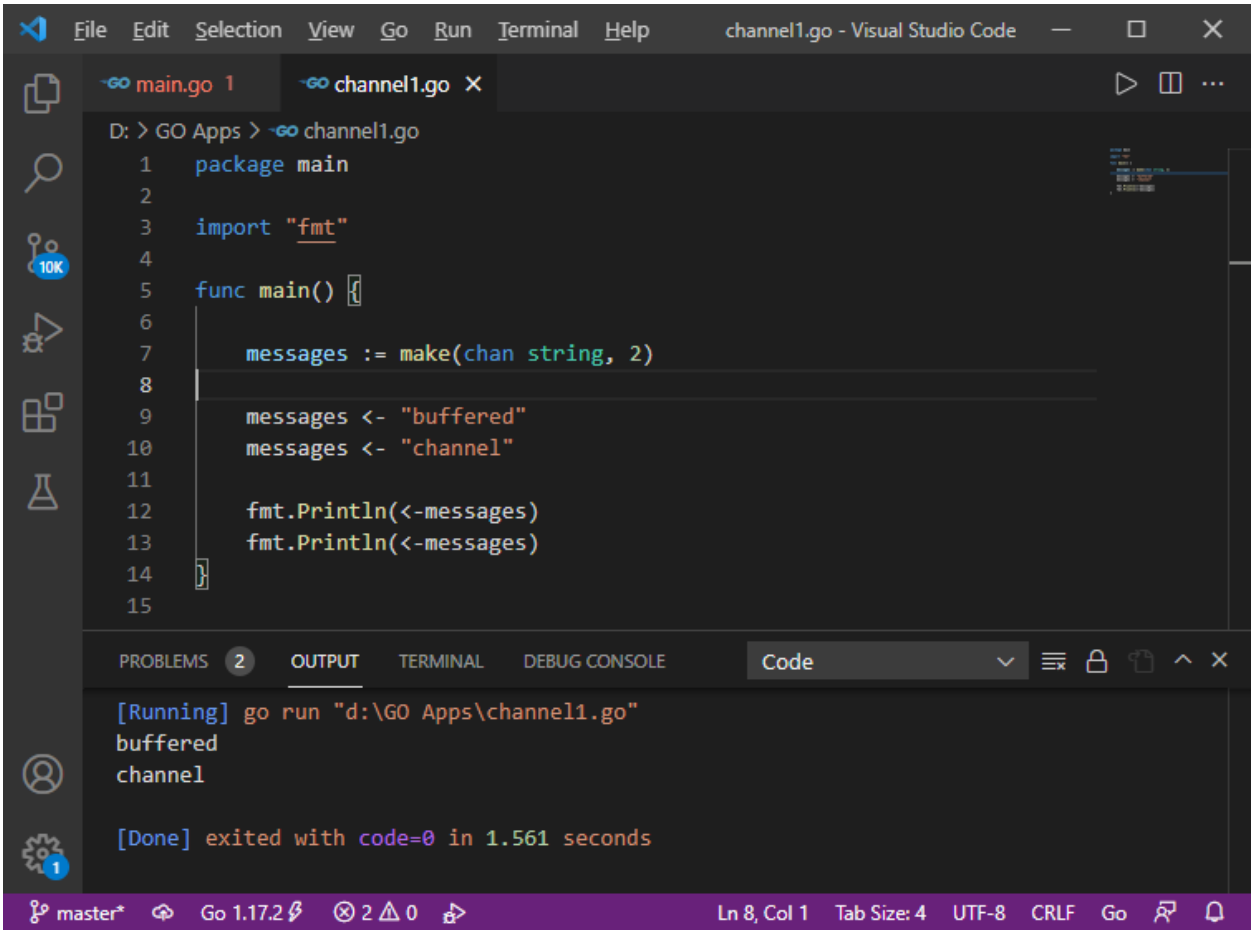
Problema 1

```go
package main

import "fmt"

func main() {

    messages := make(chan string)

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

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



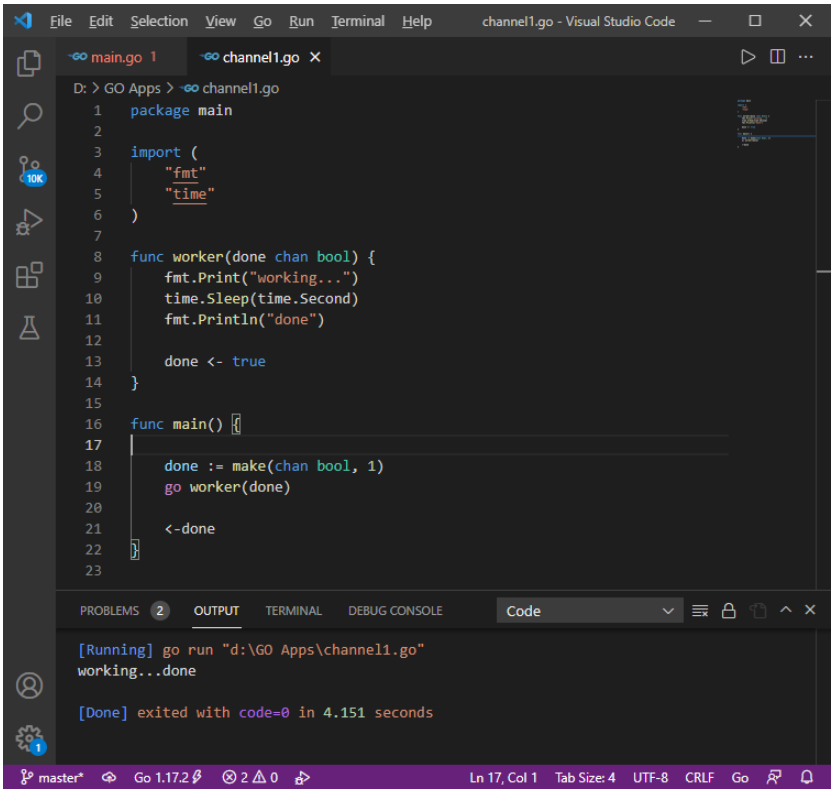
Problema 2

```go
package main

import "fmt"

func main() {

    messages := make(chan string, 2)

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

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



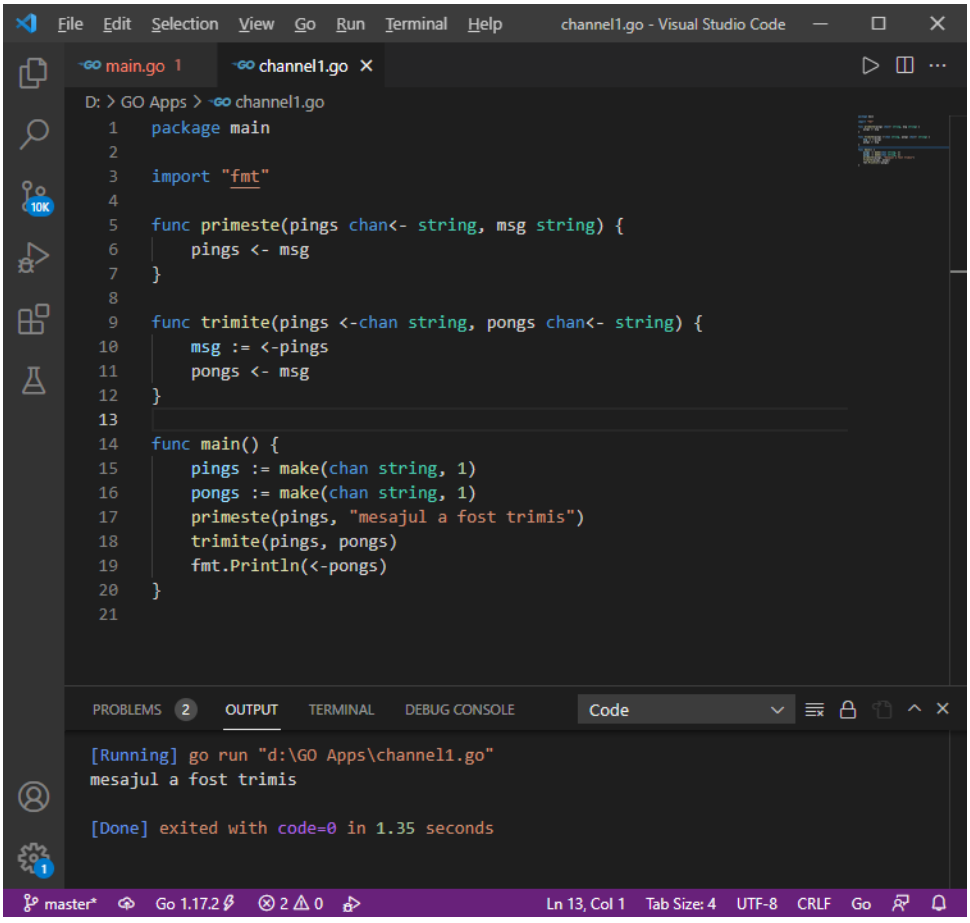
Problema 3

```go
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
}
```



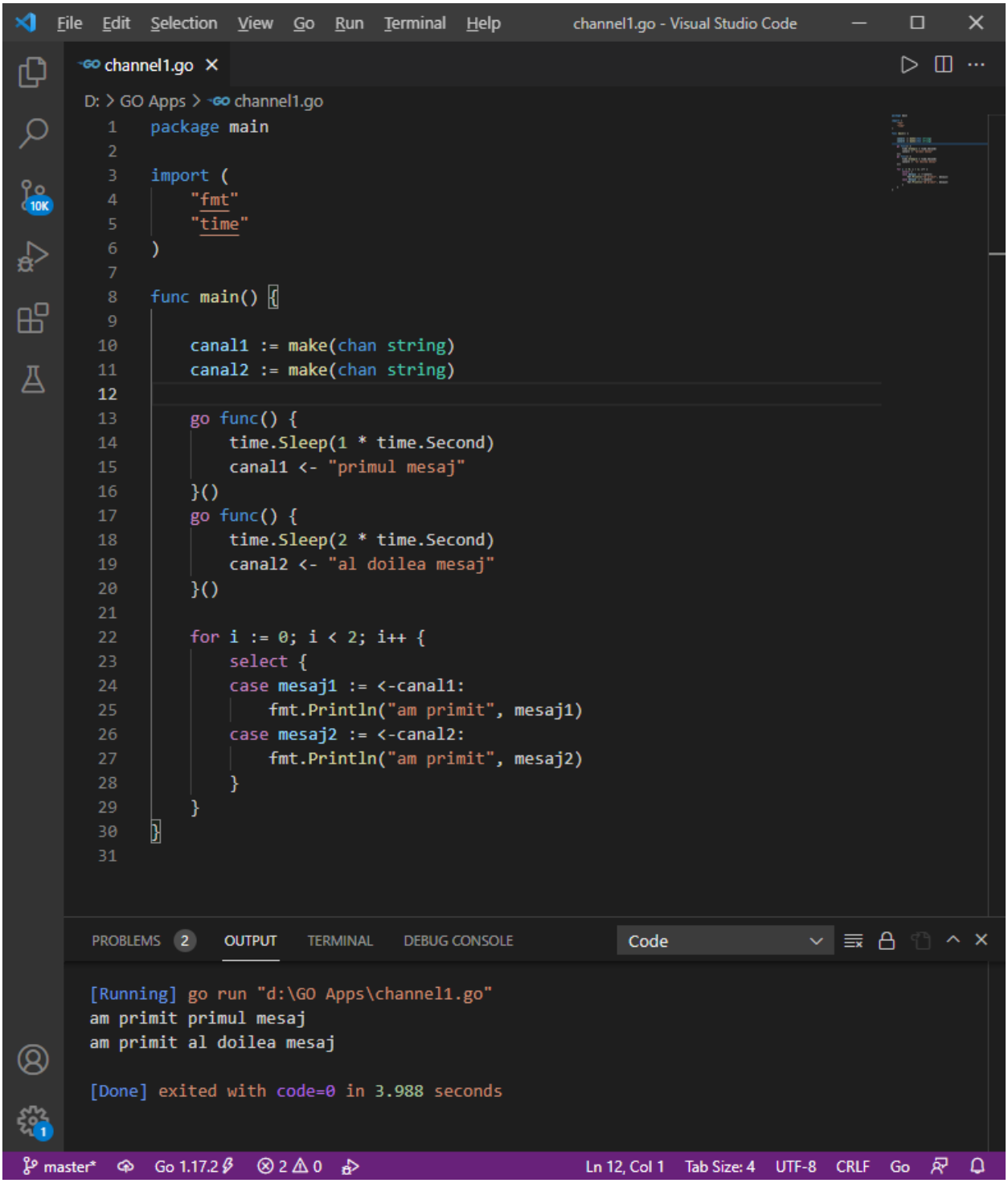
Problema 4

```go
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)
}
```



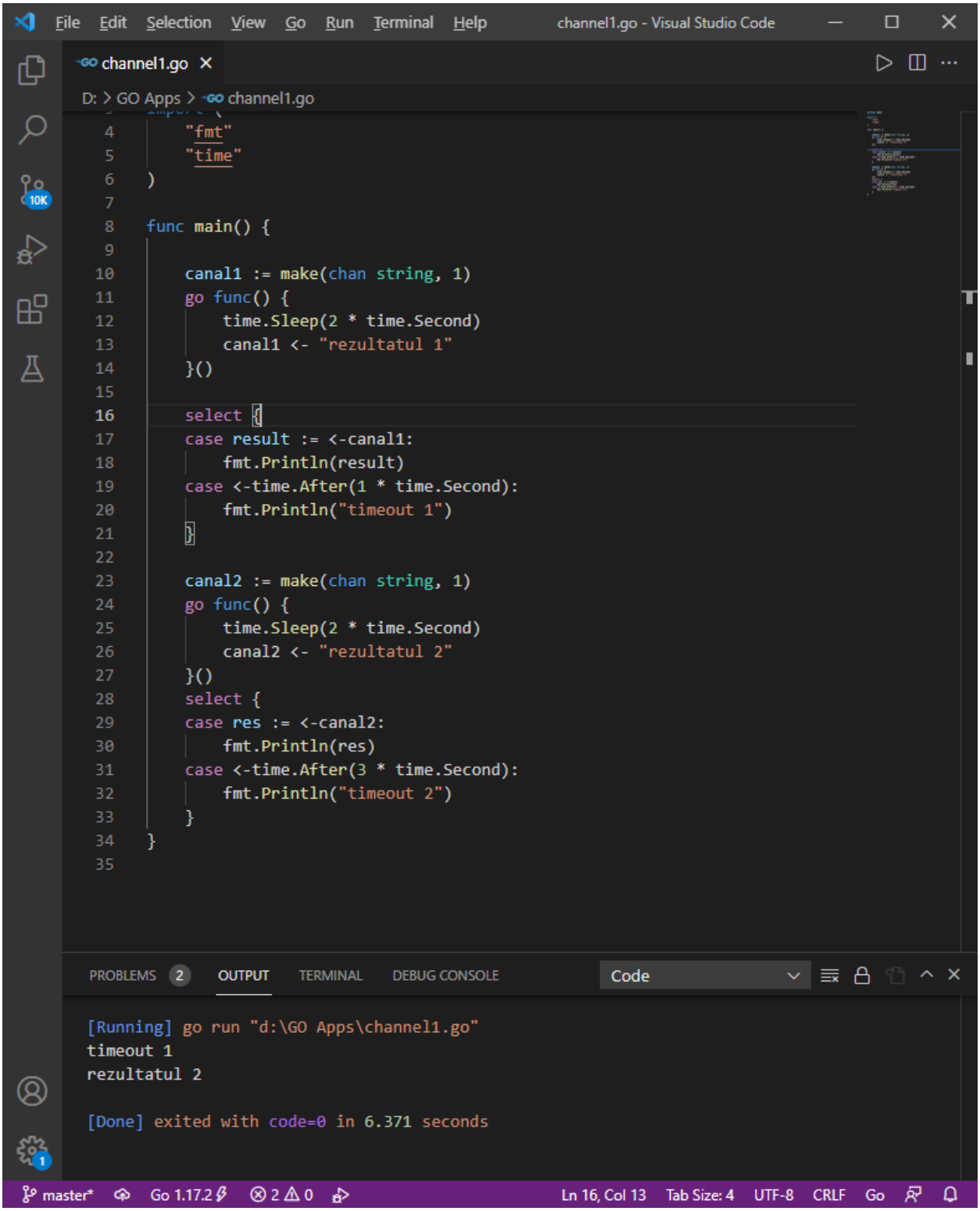
Problema 5

```go
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)
        }
    }
}
```


File Edit Selection View Go Run Terminal Help
channel1.go - Visual Studio Code
channel1.go
D: > GO Apps > 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

```go
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")
    }
}
```


File Edit Selection View Go Run Terminal Help
channel1.go - Visual Studio Code
channel1.go
D: > GO Apps > channel1.go
"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")
}
}
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 0
Ln 16, Col 13
Tab Size: 4
UTF-8
CRLF
Go