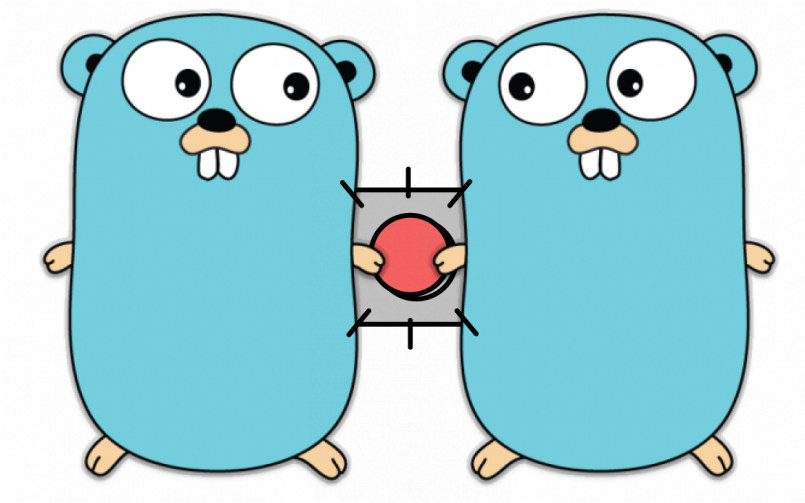
Exercise: Buzz Game

Let's implement a buzz game using the tools we have learnt so far!

In the code below, you will find two goroutines which represent our players and send messages over channels signaling `Buzz` to the main routine. Now the problem with the code below is that `channel1` blocks the code, which implies that we can't receive any message from `channel2` until we receive a message from `channel1`. Therefore, if player 2, i.e. the second goroutine *buzzes* before player 1, we'll never be able to know!



Your job is to rectify the game provided to you in the code snippet below and make it fair such that we know which player buzzed first.

Current Output:

```
Player 1 Buzzed
Player 2 Buzzed
```

Expected Output:

If Player 1 buzzes first,

```
Player 1 Buzzed
Player 2 Buzzed
```

If Player 2 buzzes first,

```
Player 2 Buzzed
Player 1 Buzzed
```

Show Hint

```go
package main

import (
  "fmt"
  "time"
  "math/rand"
)

func main() {
  channel1 := make(chan string)
  channel2 := make(chan string)

  go func() {
    rand.Seed(time.Now().UnixNano())
    time.Sleep(time.Duration(rand.Intn(500)+500) * time.Millisecond)
    channel1 <- "Player 1 Buzzed"
  }()

  go func() {
     rand.Seed(time.Now().UnixNano())
     time.Sleep(time.Duration(rand.Intn(500)+500) * time.Millisecond)
     channel2 <- "Player 2 Buzzed"
  }()

  fmt.Println(<-channel1)
  fmt.Println(<-channel2)

}
```

Buzz Game

Hope you were able to figure out the correct implementation! Check it out in the next lesson.