```
package main

import(
  "fmt"
  "time"
)

func print(x *int){
  fmt.Printf("Value of x : %d",*x)
}
func increment(x *int){
  fmt.Printf("Incrementing : ");
  *x += 1
}

func main(){
  var x int = 0;
  go operate(&x)
  go print(&x)

  time.Sleep(time.Second)
    fmt.Println("DONE")
}
```

If these two functions are excuted concurrently, the race condition would occur because these two functions are access the same resource and the output depends on non-deterministic ordering.

Race condition is a condition when two or more thread can access the same resource at the same time. The output of the program in race condition became non-deterministic. The behavior of the same program could output two difference things.