UFCG UASC/CEEI

Disciplina: Programação Concorrente Professor: Thiago Emmanuel Pereira Prova 3

Q1 - Explique e justifique a saída esperada do programa abaixo.

```
package main
import "fmt"

func xpto(c chan int, int value) {
        c <- value
}

func main() {
    ch1 := make(chan int)
    ch2 := make(chan int)

    go xpto(ch1, 42)
    go xpto(ch1, 43)

    select {
    case v1 := <-ch1:
        fmt.Println("valor recebido de ch1:", v1)
    case v2 := <-ch2:
        fmt.Println("valor recebido de ch2:", v2)
    }
}
```

Q2 - Considere a API abaixo como uma função que retorna um canal no qual um número indeterminado de strings serão enviadas.

```
func request_stream() chan string
```

considere que em um programa, dois canais destes estão sendo manipulados da seguinte forma:

```
func main() {
       ch1 := request_stream()
       ch2 := request_stream()

}
```

considere que você deve incluir na sua função main uma chamada para a função

```
func ingest(in chan string)
```

que deve ser chamada como uma nova goroutine, ou seja:

```
go ingest
```

agora, o canal recebido pela função **ingest** deve conter itens disponibilizados pelos canais ch1 e ch2 na medida em que estiverem disponíveis. Ou seja, tão logo itens de ch1 e ch2 estejam disponíveis, estes podem ser enviados para o canal a ser passado para a função ingest.

Q3 - Considere um sistema que cria requisições a serem enviadas para componentes de processamento (workers). Esse sistema é crítico e não deve **criar** mais do que **maxCapacity** requisições. Considere que uma requisição é criada com a função abaixo:

func create_req() Request

Considere que uma requisição é processada através da execução da função abaixo. Esse processamento demora muito, então deve ser intermediado por goroutines.

func exec_req(req Request)

É importante que o sistema trabalhe na maior capacidade possível. Ou seja, se for possível criar mais requisições e mandá-las para processamento, isso deve ser feito. Ainda, não deve haver limitação do ponto de vista dos workers: caso uma requisição tenha sido criada é melhor que um worker a processe tão logo possa.

Altere o código abaixo, criando goroutines, canais, estrutura de dados e funções auxiliares para resolver o problema. Assuma que o problema irá funcionar indefinidamente.

```
package main

const maxCapacity = 10 // Maximum capacity of the system

func main() {

	//loop de criação de requisições. altere para realizar o controle de criação
	for {

		var req = create_req()
	}
}
```