Напишите функцию `bridge`, которая вычитывает очередной канал из канала `ins` и перенаправляет данные из вычитанного канала в возвращаемый канал, пока вычитанный канал открыт и контекст не отменен. Функция продолжает работу, пока канал `ins` открыт и контекст не отменен.

```
package main

import (
 "context"
 "reflect"
)

func main() {
 genVals := func() <-chan <-chan interface{} {
  out := make(chan (<-chan interface{}))
  go func() {
   defer close(out)
   for i := 0; i < 3; i++ {
    stream := make(chan interface{}, 1)
    stream <- i
    close(stream)
    out <- stream
   }
  }()
  return out
 }

 var res []interface{}
 for v := range bridge(context.Background(), genVals()) {
  res = append(res, v)
 }

 if !reflect.DeepEqual(res, []interface{}{0, 1, 2}) {
  panic("wrong code")
 }
}

func bridge(ctx context.Context, ins <-chan <-chan interface{}) <-chan interface{} {
 // напишите ваш код здесь
}

func orDone(ctx context.Context, in <-chan interface{}) <-chan interface{} {
 out := make(chan interface{})
 go func() {
  defer close(out)
  for {
   select {
   case <-ctx.Done():
    return
   case v, ok := <-in:
    if !ok {
     return
    }
    select {
    case out <- v:
    case <-ctx.Done():
```

```
    }
   }
  }
 }()
 return out
}
```