```go
// Copyright 2009 The Go Authors. All rights reserved.
// Use of this source code is governed by a BSD-style
// license that can be found in the LICENSE file.

package main

import "fmt"

// Send the sequence 2, 3, 4, ... to channel 'ch'.
func generate(ch chan int) {
    for i := 2; ; i++ {
        ch <- i  // Send 'i' to channel 'ch'.
    }
}

// Copy the values from channel 'in' to channel 'out',
// removing those divisible by 'prime'.
func filter(in, out chan int, prime int) {
    for {
        i := <-in  // Receive value of new variable 'i' from 'in'.
        if i % prime != 0 {
            out <- i  // Send 'i' to channel 'out'.
        }
    }
}

// The prime sieve: Daisy-chain filter processes together.
func main() {
    ch := make(chan int)  // Create a new channel.
    go generate(ch)  // Start generate() as a goroutine.
    for {
        prime := <-ch
        fmt.Println(prime)
        ch1 := make(chan int)
        go filter(ch, ch1, prime)
        ch = ch1
    }
}
```