Необходимо написать worker pool: нужно выполнить параллельно `numJobs`заданий, используя `numWorkers` горутин, которые запущены единожды за время выполнения програмы.

Для этого напишите функции `worker` и `main`.

Функция `worker`:

- на вход получает функцию для выполнения `f`, канал для получения аргументов `jobs` и канал для записи результатов `results`
- читает из `jobs` и записывает результат выполнения `f(job)` в `results`.

Функция `main`:

- запускает функцию `worker` в `numWorkers` горутинах;
- в качестве первого аргумента `worker` использует функцию `multiplier`;
- пишет числа от 1 до `numJobs` в канал `jobs`;
- читает и выводит полученные значения из канала `results`, *паралелльно работе воркеров*

```
package main

import (
 "fmt"
 "sync"
)

func worker(f func(int) int, jobs <-chan int, results chan<- int) {
 // напишите ваш код здесь
}

const numJobs = 5
const numWorkers = 3

func main() {
 jobs := make(chan int, numJobs)
 results := make(chan int, numJobs)
 wg := sync.WaitGroup{}

 multiplier := func(x int) int {
  return x * 10
 }

 // напишите ваш код здесь
}
```