Тест

1. Перечислите известные вам типы данных в GoLang.
2. Перечислите известные вам встроенные функции (predeclared functions) в Go.
3. Выберите код, который не вызывает ошибок компиляции:

|  |  |  |
| --- | --- | --- |
| В теле функции | Вне функции |  |
| + | + | var k int = 15 |
| - | - | var k int := 15 |
| - | - | k = 15 |
| + | - | k := 15 |
| + | - | var k int  k = 15 |
| - | - | var k int  k := 15 |
| + | + | var k, m = 15, 20 |
| + | - | var k, m int  k, m = 15, 20 |
| + | - | k, m := 15, 20 |
| - | - | const k := 15 |
| + | + | const k int = 15 |
| + | + | const k = 15 |
| + | + | const (  k = 15  m = 20  ) |
| + | + | const k, m = 15, 20 |
| - | - | const k = [2]int{1, 2} |
| + | + | type myType struct {}  var b = myType{} |
| - | - | type myType struct {}  const b = myType{} |
| - | - | type myInterface interface {}  const b = myInterface |
| + | - | type myInterface interface {}  var b myInterface  b = 5 |

1. If else: выберите код, который не вызывает ошибок компиляции:

|  |  |
| --- | --- |
| - | var k = 5  if k = 5 {  print(k)  } |
| + | k := 5  if k == 5 {  print(k)  } |
| - | var k int  if k = 5 {  print(k)  } |
| + | k := true  if k {  print(k)  } |
| + | k := 5  if k == 5 {  print(k)  } else if k == 4 {  print(k)  } else if k ==3 {  print(k)  } |
| + | if k := 5; k ==5 {  print(k)  } |
| + | if k,m := 5,4; k==5 {  print(k, m)  } |
| - | if k:=5; m:=4; k==5 {  print(k, m)  } |
| + | var k int  if k = 5; k==5 {  print(k)  } |

1. For: выберите код, который не вызывает ошибок компиляции:

|  |  |
| --- | --- |
| + | i := 0  for ;i <= 10; {  print(i)  i += 2  } |
| + | i := 0  for i <= 10 {  print(i)  i += 2  } |
| + | for i:=0; i <= 10; {  print(i)  i += 2  } |
| + | for i,j:=0,2; i <= 10; {  print(i)  i += 2  j++  } |
| - | for i:=0; j:=2; i <= 10; {  print(i)  i += 2  j++  } |
| + | for i,j := 1,10; i<=10 && j<=19; i,j = i+1,j+1 {  print(i, j)  } |

1. Что выведет следующий код:

|  |  |
| --- | --- |
| const (  a = iota + 1  \_  b  c  )  print(a, b, c) | 134 |
| for i:=0; i <= 10; i++{  if(i%2 == 0) {  print(i)  break  }  } | 0 |
| for i:=0; i <= 10; i++{  if(i%2 == 0) {  continue  }  print(i)  } | 13579 |
| func task1() {  a := 1  defer fmt.Println(a)  a = 3  } | 1 |
| func task2() {  a := 1  b := &a  defer fmt.Println(\*b)  c := 3  b = &c  } |  |
| func task3() {  for i := 0; i < 5; i++ {  defer fmt.Println(i)  }  println(0)  } | 043210 |
| func task4() {  a := 1  defer func(a int) {  fmt.Println(a)  }(a)  a = 2  } | 1 |
| func task5() {  a := 1  defer func() {  fmt.Println(a)  }()  a = 2  } | 2 |
| func task6() {  a := 1  f := func() {  a = 2  println(a)  }  a = 3  f()  println(a)  } | 2  2 |
| type myInterface interface{}  func task7() {  var k myInterface  print(k == nil)  } | true |
| type myInterface interface{}  type myString string  func task8 () {  var k myInterface  var m \*myString  k = m  print(k == nil)  } | false |

1. Напишите пример кода для бесконечного цикла
2. Что выведет следующий код:

|  |  |
| --- | --- |
| a:=[4]int{0,1,2,3}  s:=a[1:3]  fmt.Println(s) | [1 2] |
| a:=[4]int{0,1,2,3}  s:=a[1:3]  fmt.Println(len(s),cap(s)) | 2 3 |
| a:=[...]int{0,1,2,3,4,5,6,7,8}  s:=a[1:3]  fmt.Println(len(s),cap(s)) | 2 8 |
| a:=[4]int{0,1,2,3}  s:=a[2:]  s = append(s, 10)  fmt.Println(s, len(s), cap(s)) | [2 3 10] 3 4 |
| a:=[4]int{0,1,2,3}  s:=a[2:]  s = append(s, s...)  fmt.Println(s, len(s), cap(s)) | [2 3 2 3] 4 4 |
| a:=[4]int{0,1,2,3}  s:=a[2:3]  s = append(s, 10)  fmt.Println(a, s) | [0 1 2 10] [2 10] |
| a:=[4]int{0,1,2,3}  s:=a[2:3]  s = append(s, 10, 11)  fmt.Println(a, s) | [0 1 2 3] [2 10 11] |
| s := []int{}  fmt.Println(s, len(s), cap(s)) | [] 0 0 |
| s := make([]int, 5)  fmt.Println(s, len(s), cap(s)) | [0 0 0 0 0] 5 5 |
| s := make([]int, 0, 5)  fmt.Println(s, len(s), cap(s)) | [] 0 5 |
| a:=[...]int{0,1,2,3,4,5,6,7,8}  s:=a[1:]  t:=s[1:]  t[0] = 10  fmt.Println(a, s, t) | [0 1 10 4 5 6 7 8] [1 10 3 4 5 6 7 8] [10 3 4 5 6 7 8] |
| a:=[...]int{0,1,2,3,4,5,6,7,8}  s:=a[1:3]  t:=a[5:8]  n:=copy(s, t)  fmt.Println(n, s) | 2 [5, 6] |
| a:=[...]int{0,1,2,3,4,5,6,7,8}  s:=[]int{}  t:=a[5:8]  n:=copy(s, t)  fmt.Println(n, s) | 0 [] |
| func main() {  m:= [3]string{"one", "two", "three"}  changeValue(m)  fmt.Println(m)  }  func changeValue(m [3]string) {  m[2] = "zero"  } | [one two three] |
| func main() {  m:= []string{"one", "two", "three"}  changeValue(m)  fmt.Println(m)  }  func changeValue(m []string) {  m[2] = "zero"  } | [one two zero] |
| func main() {  m:= map[int]string{  1: "one",  2: "two",  3: "three",  }  changeValue(m)  fmt.Println(m)  }  func changeValue(m map[int]string) {  m[2] = "zero"  } | map[1:one 2:zero 3:three] |