Question	Answer 1	Answer 2	Answer 3	Answer 4	Notes
When was Go 1.0 released?	November 2009	November 2009	March 2012	June 2012	Go 1.0 was released March 28, 2012
How many GOOS-GOARCH combinations does Go 1.13 support?	40	42	44	46	`go tool dist list`

What does the following program print?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`  for i := 0; i < 4; i++ {`
`  go func() {`
`  fmt.Printf("%d, ", i)`
`  }()`
`  }`
`  select {}`
`}`
	1, 2, 3, 4,	4, 4, 4, 4,			

What does the following program print?
 	
`package main`
 	
`import (`
`"fmt"`
`)`
 	
`func main() {`
`var x int`
`for i := 0; i < 10; i++ {`
`go func() {`
`for i := 0; i < 1_000_000; i++ {`
`x++`
`}`
`}()`
`}`
`fmt.Println(x)`
`}`
	1000000	0	Impossible to know		The correct answer in the quiz at the meetup was 0 but it should probably be "Impossible to know". There is no guarantee when go routines start and we cannot safely assume they don't start.

What does the following program print?
 	
`package main`
 	
`import (`
`"fmt"`
`"time"`
`)`
 	
`func main() {`
`var x int`
`for i := 0; i < 10; i++ {`
`go func() {`
`for i := 0; i < 1_000_000; i++ {`
`x++`
`}`
`}()`
`}`
`time.Sleep(time.Second)`
`fmt.Println(x)`
`}`
	1000000	0	Impossible to know		

What does this program print?
 	
`package main`
 	
`import "log"`
 	
`func main() {`
`  err := makeASandwich()`
`  if err != nil {`
`  log.Fatal("sad")`
`  }`
`  log.Println("happy")`
`}`
 	
`func makeASandwich() error {`
`  var err *IngredientMissingError`
`  return err`
`}`
 	
`type IngredientMissingError struct {`
`  Ingredient string`
`}`
 	
`func (e IngredientMissingError) Error() string {`
`  return "missing ingredient: " + e.Ingredient`
`}`
	happy	sad			

Can global variables be declared using `:=` ?	Yes	No			
Does Go support method overloading	Yes	No			
Is the following code allowed: `var a, b, c = 1, 2, "3"`	Yes	No			

What does this program print?
 	
`package main`
 	
`import (`
`  "encoding/json"`
`  "fmt"`
`)`
 	
`type X struct {`
`  json.Marshaler`
`  Y string `json:"y"``
`}`
 	
`func main() {`
`  x := X{Y: "Z"}`
`  b, _ := json.Marshal(x)`
`  fmt.Println(string(b))`
`}`
	{"y":"Z"}	{"Y":"Z"}	Does not compile	Panics	

What does this program print?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`fmt.Println(x())`
`}`
 	
`func x() (v int) {`
`v = 1`
 	
`defer func() {`
`v = 2`
`}()`
 	
`defer func() {`
`v = 3`
`}()`
 	
`return 4`
`}`
	1	2	3	4	

What does the following line print on play.golang.org `fmt.Println(time.Now().Format("2006-01-02"))`	2001-01-02	2019-11-28	2009-11-10	1970-01-01	The time in the playground is fixed to the date when Go was first publically announced

What does this program print?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`x := map[string]int{`
`"a": 1,`
`"c": 3,`
`"b": 2,`
`}`
`for k, v := range x {`
`fmt.Println(k, v)`
`}`
`}`
	a 1
	c 3
	b 2
		a 1
		b 2
		c 3
			Not deterministic		Maps are iterated in a nondeterministic order

What does this program print?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`x := map[string]int{`
`"a": 1,`
`"c": 3,`
`"b": 2,`
`}`
`fmt.Println(x)`
`}`
	map[a:1 b:2 c:3]	map[a:1 c:3 b:2]	Not deterministic		Since Go 1.12, when printing a map the keys are sorted (order is described in docs)

What does this program print?
 	
`package main`
 	
`import (`
`"fmt"`
`"math"`
`)`
 	
`func main() {`
`fmt.Println(math.NaN() == math.NaN())`
`fmt.Println(math.Inf(1) == math.Inf(1))`
`}`
	true
	true
		false
		true
			true
			false
				false
				false
					IEEE 754 says that only NaNs satisfy f != f

How long will this sleep for? `time.Sleep(1_000_000)`	1 second	1 millisecond	1 microsecond	1 nanosecond	Duration represents time as nanoseconds. 1M nanoseconds = 1 millisecond

What does this program print?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`a := []int{1, 2, 3, 4, 5}`
`b := a[2:3:4]`
`b[0] = 7`
`fmt.Println(a)`
`}`
	1 2 3 4 5	1 2 3 7 4 5	1 2 3 7 4	syntax error	In the quiz there was a typo on in the correct answer. The actually correct answer is 1 2 7 4 5. Sorry!

Does this compile?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`foo := nil`
`fmt.Println(foo)`
`}`
	Yes	No			./main.go:6:6: use of untyped nil

Does this compile?
 	
`package main`
 	
`import "fmt"`
 	
`func main() {`
`nil := "foo"`
`fmt.Println(nil)`
`}`
	Yes	No			You're allowed to shadow nil. Don't do this though :)

Which of the following are new features landing in Go 1.14?	testing: Streaming output from `t.`	ioutil: Allow controlloing where r	sync: Significant performance ir	testing: Add `t.Cleanup()` to perform cleanup after test has finished	