QUIZ

Complete the following code to create a `switch` statement that uses a scoped short declaration for the variable `pizzaTopping`.

```
switch pizzaTopping := "pepperoni" ; pizzaTopping {
  case "cheese":
  fmt.Println("One pizza with extra cheese coming up!")
  case "pepperoni":
  fmt.Println("One pepperoni pizza in the oven!")
  default :
  fmt.Println("I don't have that topping, get something else.")
}
```

👏 You got it!

What is the purpose of a seed value in Go?

Go uses seeds as references for where to store encrypted files.

A seed value is the start of many binary trees.

A seed value provides a starting point for Go to generate random numbers.

👏 Correct, by default Go uses a default seed value of `1` which can lead to predictable numbers being generated. We can generate random numbers by using different seed values.

A seed value limits the length of strings to maximize storage space in a Go program.

What is the purpose of using conditionals?

Conditionals allow different blocks of code to be run depending on the values passed to it.

Correct! Using conditionals, we can account for different situations by running different blocks of code.

We use conditionals when we want to use a different language in a Go program.

Conditionals are used to run every single line of code regardless of any errors.

Conditionals automatically tell the Go compiler to stop running so we can debug our code.

Given the following code snippet, what is printed to the terminal?

```
walksLikeDuck := true
talksLikeDuck := false

if walksLikeDuck && talksLikeDuck {
  fmt.Println("It's a duck!")
} else if walksLikeDuck || talksLikeDuck {
  fmt.Println("It may or may not be a duck.")
} else {
  fmt.Println("It's not a duck!")
}
```

`It's not a duck!`

`It may or may not be a duck.`

Yes, our `if` condition evaluates to `false` but our `else if` conditional (`walksLikeDuck || talksLikeDuck`) evaluates to `true` since only one of the conditions need to be true. Therefore, `It may or may no...

Show more

Complete the code snippet below to have a working conditional.

```
if randNum := rand.Intn(10) ; randNum < 7 {
  fmt.Println("Less than 7.")
} else if randNum > 5 {
  fmt.Println("Less than 5.")
} else {
  fmt.Println("Greater than 7.")
}
```

👏 You got it!

Given the following code snippet, what will print to the terminal?

```
weekend := true
dayOfWeek := 7

if !weekend {
  switch dayOfWeek {
  case 1:
    fmt.Println("It's Monday already?")
  case 2:
    fmt.Println("Tuesday... still better than Monday.")
  case 3:
    fmt.Println("Wednesday, it's hump day!")
  case 4:
    fmt.Println("Thursday, almost done.")
  case 5:
    fmt.Println("Friday! TGIF!")
  default:
    fmt.Println("What day of the week is it?")
  }
} else {
  fmt.Println("Take the day off!")
}
```

```
What day of the week is it?
```

```
Thursday, almost done.
```

```
Take the day off!
```

👏 Correct, since `!weekend` is `false`, the code from the `else` block runs.

Given the following code snippet, what is the value of `mixedOperators`?

```
var mixedOperators bool

mixedOperators = true || false
mixedOperators = mixedOperators && true
mixedOperators = !mixedOperators
```

`true`

`false`

Correct! The first `true || false` evaluates to `true`. Then `true && true` is `true`. Then `!true` is `false`.

Given the following code snippet, where would an `else if` statement go?

```
if favColor == "red" {
  fmt.Println("Excellent choice.")
} else {
  fmt.Println("You do you.")
}
```

The `else if` statement would go after the `if` block and before the `else` statement.

Right, that's the correct syntax!

Given the following code snippet, what will print to the terminal?

```
gradeAvg := 89

if gradeAvg >= 90 {
  fmt.Println("You've earned an A!")
} else if gradeAvg >= 80 {
  fmt.Println("You've earned a B.")
} else if gradeAvg >= 70 {
  fmt.Println("You've earned a C?")
} else if gradeAvg >= 60 {
  fmt.Println("You've earned a D...")
} else {
  fmt.Println("An F???")
}
```

`You've earned an A!`

`An F???`

`You've earned a C?`

`You've earned a B.`

👏 `gradeAvg` has a value of `89` and thus is `true` for the first `else if` condition. Therefore, its code runs and `You've earned a B.` is printed.

When is an `else` statement used?

`else` statements check a provided condition and runs code if the condition is true.

Including an `else` statement provides a default block of code to run when none of the previous conditions are true.

👏 That's right! The code inside an `else` runs by default when none of the previous conditions are true.

An `else` statement skips over any code that might be problematic.

`else` statements are used as a substitute for `if` statements when a developer deems it appropriate.