Unit 3—Lesson 2:
Type Casting and Inspection

Type inspection

```
func getClientPet() -> Animal {
  //returns the pet
}

let pet = getClientPet() //`pet` is of type `Animal`
```

Type inspection

```
if pet is Dog {
  print("The client's pet is a dog")
} else if pet is Cat {
  print("The client's pet is a cat")
} else if pet is Bird {
  print("The client's pet is a bird")
} else {
  print("The client has a very exotic pet")
}
```

Type inspection

```
let pets = allPets() //`pets` is of type `[Animal]`
var dogCount = 0, catCount = 0, birdCount = 0
for pet in pets {
  if pet is Dog {
    dogCount += 1
  } else if pet is Cat {
    catCount += 1
  } else if pet is Bird {
    birdCount += 1
  }
}
print("Brad looks after \(dogCount) dogs, \(catCount) cats, and \(birdCount) birds.")
```

Type casting

```
func walk(dog: Dog) {
  print("Walking \(dog.name)")
}

func cleanLitterBox(cat: Cat) {. . .}

func cleanCage(bird: Bird) {. . .}

for pet in pets {
  if pet is Dog {
    walk(dog: pet) // Compiler error. The compiler sees `pet` as an `Animal`, not a `Dog`.
  }
  . . .
}
```

Type casting

```
for pet in pets {
  if let dog = pet as? Dog {
    walk(dog: dog)
  } else if let cat = pet as? Cat {
    cleanLitterBox(cat: cat)
  } else if let bird = pet as? Bird {
    cleanCage(bird: bird)
  }
}
```

Any

```
var items: [Any] = [5, "Bill", 6.7, Dog()]
```

Any

```
var items: [Any] = [5, "Bill", 6.7, Dog()]
let firstItem = items[0]

if firstItem is Int {
  print("The first element is an integer")
} else if firstItem is String {
  print("The first element is a string")
} else {
  print("The first element is neither an integer nor a string")
}
```

Any

```
var items: [Any] = [5, "Bill", 6.7, Dog()]

if let firstItem = items[0] as? Int {
  print(firstItem + 4)
}
```

```
9
```

Unit 3—Lesson 2

Lab: Type Casting and Inspection

Open and complete the exercises in `Lab - Type Casting.playground`