Applied Statistical Programming - Methods and Classes

Kimberly Acquilano, Zion Little, Jordan Newton, Alma Velazquez

2/9/2022

Write the R code to answer the following questions. Write the code, and then show what the computer returns when that code is run. Thoroughly comment your solutions.

You have until the beginning of class 2/14 at 10:00am to answer all of the questions below. You may use R, but not any online documentation. Submit the Rmarkdown and the knitted PDF to Canvas. Only one member of your group needs to submit the in class exercise, but everyone's names need to be included on the submitted document.

The Animal Kingdom

In this exercise, you will demonstrate knowledge of the S3 version of object-oriented programming using methods and classes. To do this, you will model creatures from the animal kingdom with different traits.

Consider five animals: a `cat`, a `dog`, a `cow`, a `cobra`, and an `iguana`. The first three of these are `mammals`, and the latter two of these are `reptiles`.

Every animal in the list can `eat` something (carnivor, herbivor, omnivore) and make `noise`. Each animal makes different noises.

1. With your group, decide how to define a class for `mammal` and `reptile` with common features for each. Create a constructor and a validator for each class

```r
new_mammal <- function(noise, eats, sleep) {
    mammal <- list(live_birth = TRUE, warm_blooded = TRUE, vertibra = TRUE, has_hair = TRUE,
        has_scales = FALSE, noise = noise, eats = eats, sleep = sleep)
    class(mammal) <- "mammal"
    return(mammal)
}

new_reptile <- function(noise, eats, sleep) {
    reptile <- list(live_birth = FALSE, warm_blooded = FALSE, vertibra = TRUE, has_hair = FALSE,
        has_scales = TRUE, noise = noise, eats = eats, sleep = sleep)
    class(reptile) <- "reptile"
    return(reptile)
}

validate_mammal <- function(x) {
    if (x$warm_blooded == FALSE) {
        stop("This animal is cold blooded!")
    }
    if (x$live_birth == FALSE) {
        stop("This animal lays eggs!")
    }
    if (x$has_hair == FALSE) {
        stop("This animal don't have hair!")
    }
    if (x$has_scales == TRUE) {
        stop("This animal has scales!")
    }
    if (x$vertibra == FALSE) {
        stop("This animal doesn't have a vertibra!")
    }
    return(x)
}

validate_reptile <- function(x) {
    if (x$warm_blooded == TRUE) {
        stop("This animal is warm blooded!")
    }
    if (x$live_birth == TRUE) {
        stop("This animal doesn't lay eggs!")
    }
    if (x$has_hair == TRUE) {
        stop("This animal has hair!")
    }
    if (x$has_scales == FALSE) {
        stop("This animal doesn't have scales!")
    }
    if (x$vertibra == FALSE) {
        stop("This animal doesn't have a vertibra!")
    }
    return(x)
}
```

2. Now create each of the five animals listed above appropriately differentiating each (e.g., they should all make different noises and be assigned to the correct class).

```r
# dog <- list(noise='woof', eats='omnivore', sleep='diurnal')
# class(dog)<-list('dog','mammal') cat <- list(noise='meow', eats='omnivore',
# sleep='nocturnal') class(cat)<-list('cat','mammal') cow <- list(noise='moo',
# eats='herbivore', sleep='diurnal') class(cow)<-list('cow','mammal') cobra <-
# list(noise='hiss', eats='carnivore', sleep='diurnal')
# class(cobra)<-list('cobra', 'reptile') iguana <- list(noise='...',
# eats='herbivore', sleep='diurnal') class(iguana)<-list('iguana', 'reptile')

##### ALTERNATIVELY, can define using our constructor:
dog <- new_mammal(noise = "woof", eats = "omnivore", sleep = "diurnal")

cat <- new_mammal(noise = "meow", eats = "omnivore", sleep = "nocturnal")

cow <- new_mammal(noise = "moo", eats = "herbivore", sleep = "diurnal")

cobra <- new_reptile(noise = "hiss", eats = "carnivore", sleep = "diurnal")

iguana <- new_reptile(noise = "...", eats = "herbivore", sleep = "diurnal")
```

```
dog <- validate_mammal(dog)
iguana <- validate_reptile(iguana)

# This should give an error message!  cat<-validate_reptile(cat)
```

3. Create a generic method called `replicate`. This function should work only to then call the appropriate method.

```
replicate <- function(animal) {
    UseMethod("replicate")
}
```

4. Create a `mammal` and `reptile` method for `replicate`. For mammals it should print out, "I have given live birth to offspring or am a monotreme!" For reptiles it should say, "I have laid some eggs or possibly am ovoviviparous or viviparous."

```
replicate.mammal <- function(animal) {
    print("I have given live birth to offspring or am a monotreme!")
}

replicate.reptile <- function(animal) {
    print("I have laid some eggs or possibly am ovoviviparous or viviparous.")
}

replicate(iguana)
```

```
## [1] "I have laid some eggs or possibly am ovoviviparous or viviparous."
```

```
replicate(cow)
```

```
## [1] "I have given live birth to offspring or am a monotreme!"
```

5. Create a method for `print` called `print.animal`. The output should include the noise. (Be creative.) Assign the class `animal` to your `cow` (it should then have two classes `mammal` and `animal`) and then call `print`. What does it do? Why?

```
print.animal <- function(animal) {
    print(animal$noise)
}

class(cow) <- list("cow", "mammal", "animal")

print(cow)  # This prints noise because we have assigned the method for animal to only print its noise
```

```
## [1] "moo"
```