S3

20173204 곽명빈

2020-05-27

Data

Expected

```
play()
## B 7 0
## $0
```

Observed

```
load("./Programs.RData")
play
```

```
## function() {
##   symbols <- get_symbols()
##   print(symbols)
##   score(symbols)
## }
```

```
play()
```

```
## [1] "DD" "B"  "0"
```

```
## [1] 0
```

```
one_play <- play()
```

```
## [1] "0" "0" "0"
```

```
one_play
```

```
## [1] 0
```

S3 System

```
num <- 1000000000
print(num)
```

```
## [1] 1e+09
```

```
class(num) <- c("POSIXct", "POSIXt")
print(num)
```

```
## [1] "2001-09-09 10:46:40 KST"
```

Attributes

```
load("./Environments.RData")
attributes(DECK)
```

```
## $names
## [1] "face"  "suit"  "value"
## 
## $row.names
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
## [51] 51 52
## 
## $class
## [1] "data.frame"
```

```
row.names(DECK)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
## [16] "16" "17" "18" "19" "20" "21" "22" "23" "24" "25" "26" "27" "28" "29" "30"
## [31] "31" "32" "33" "34" "35" "36" "37" "38" "39" "40" "41" "42" "43" "44" "45"
## [46] "46" "47" "48" "49" "50" "51" "52"
```

```
row.names(DECK) <- 101:152
levels(DECK) <- c("level 1", "level 2", "level 3")
attributes(DECK)
```

```
## $names
## [1] "face"  "suit"  "value"
## 
## $row.names
##  [1] 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
## [20] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138
## [39] 139 140 141 142 143 144 145 146 147 148 149 150 151 152
## 
## $class
## [1] "data.frame"
## 
## $levels
## [1] "level 1" "level 2" "level 3"
```

```
attributes(one_play)
```

```
## NULL
```

```
#> attr(x, which) <- value
attr(one_play, "symbols") <- c("B", "0", "B")
attributes(one_play)
```

```
## $symbols
## [1] "B" "0" "B"
```

```
one_play
```

```
## [1] 0
## attr(,"symbols")
## [1] "B" "0" "B"
```

```
one_play + 1
```

```
## [1] 1
## attr(,"symbols")
## [1] "B" "0" "B"
```

```
play <- function() {
  symbols <- get_symbols()
  prize <- score(symbols)
  attr(prize, "symbols") <- symbols
  prize
}
play()
```

```
## [1] 5
## attr(,"symbols")
## [1] "B"  "B"  "BB"
```

```
play <- function() {
  symbols <- get_symbols()
  structure(score(symbols), symbols = symbols)
}
three_play <- play()
three_play
```

```
## [1] 0
## attr(,"symbols")
## [1] "BBB" "0"   "0"
```

```
attr(three_play, "symbols")
```

```
## [1] "BBB" "0"   "0"
```

```
slot_display <- function(prize) {
  # extract symbols
  symbols <- attr(prize, "symbols")
  # collapse symbols into single string
  symbols <- paste(symbols, collapse = " ")
  # combine symbol with prize as a regular expression
  # ₩n is regular expression for new line (i.e. return or enter)
  string <- paste(symbols, prize, sep = "₩n$")
  # diplay regular expression in console without quotes
  cat(string)
}
one_play
```

```
## [1] 0
## attr(,"symbols")
## [1] "B" "0" "B"
```

```
attr(one_play, "symbols")
```

```
## [1] "B" "0" "B"
```

```
symbols <- attr(one_play, "symbols")
symbols
```

```
## [1] "B" "0" "B"
```

```
paste(symbols, collapse = " ")
```

```
## [1] "B 0 B"
```

```
symbols <- paste(symbols, collapse = " ")
symbols
```

```
## [1] "B 0 B"
```

```
paste(symbols, one_play, sep = "₩n$")
```

```
## [1] "B 0 B₩n$0"
```

```
string <- paste(symbols, one_play, sep = "₩n$")
string
```

```
## [1] "B 0 B₩n$0"
```

```
cat(string)
```

```
## B 0 B
## $0
```

```
slot_display(one_play)
```

```
## B 0 B
## $0
```

```
slot_display(three_play)
```

```
## BBB 0 0
## $0
```

```
slot_display(play())
```

```
## BBB 0 0
## $0
```

```
replicate(20, slot_display(play()))
```

```
## BB B 0
## $00 BBB 0
## $00 0 B
## $00 B B
## $00 0 0
## $0BB 0 B
## $00 BB 0
## $00 B 0
## $0BB 0 0
## $0BBB 0 0
## $0DD 0 BBB
## $0BB BBB 0
## $0BBB 0 BBB
## $0B BBB 0
## $00 0 0
## $00 0 0
## $0B B 0
## $0B 0 0
## $0DD 0 0
## $00 0 0
## $0
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## NULL
##
## [[18]]
## NULL
##
## [[19]]
## NULL
##
```

```
## [[20]]
## NULL
```

```
##### cat()의 도움말을 살펴보고 대책을 강구하자.
slot_display <- function(prize) {
  symbols <- attr(prize, "symbols")
  symbols <- paste(symbols, collapse = " ")
  string <- paste0(symbols, "₩n$", prize, "₩n")
  cat(string)
}
paste("$", one_play, sep = "")
```

```
## [1] "$0"
```

```
slot_display(play())
```

```
## 0 0 BBB
## $0
```

```
replicate(20, slot_display(play()))
```

```
## BB BBB 0
## $0
## B B 7
## $0
## B 0 0
## $0
## 0 B 0
## $0
## 0 B B
## $0
## 0 0 0
## $0
## 0 B 0
## $0
## BB 0 BBB
## $0
## B 0 0
## $0
## BBB 0 B
## $0
## 0 0 0
## $0
## 0 0 B
## $0
## 0 0 0
## $0
## BB DD B
## $0
## 0 0 0
## $0
## 0 B 0
## $0
## BBB 0 0
## $0
## 0 7 BB
## $0
## BBB BB 0
## $0
## B B BBB
## $5
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## NULL
##
## [[18]]
## NULL
##
## [[19]]
## NULL
##
```

```
## [[20]]
## NULL
```

Generic Functions

```
print(pi)
```

```
## [1] 3.141593
```

```
pi
```

```
## [1] 3.141593
```

```
print(head(DECK))
```

```
##      face   suit value
## 101  king spades    13
## 102 queen spades    12
## 103  jack spades    11
## 104   ten spades    10
## 105  nine spades     9
## 106 eight spades     8
```

```
head(DECK)
```

```
##      face   suit value
## 101  king spades    13
## 102 queen spades    12
## 103  jack spades    11
## 104   ten spades    10
## 105  nine spades     9
## 106 eight spades     8
```

```
print(play())
```

```
## [1] 0
## attr(,"symbols")
## [1] "DD" "0"  "B"
```

```
play()
```

```
## [1] 0
## attr(,"symbols")
## [1] "DD" "0"  "7"
```

```
num <- 1000000000
class(num)
```

```
## [1] "numeric"
```

```
print(num)
```

```
## [1] 1e+09
```

```
class(num) <- c("POSIXct", "POSIXt")
print(num)
```

```
## [1] "2001-09-09 10:46:40 KST"
```

```
num
```

```
## [1] "2001-09-09 10:46:40 KST"
```

```
print(unclass(num))
```

```
## [1] 1e+09
```

Methods

```
print.factor
```

```
## function (x, quote = FALSE, max.levels = NULL, width = getOption("width"),
##     ...)
## {
##     ord <- is.ordered(x)
##     if (length(x) == 0L)
##         cat(if (ord)
##             "ordered"
##         else "factor", "(0)₩n", sep = "")
##     else {
##         xx <- character(length(x))
##         xx[] <- as.character(x)
##         keepAttrs <- setdiff(names(attributes(x)), c("levels",
##             "class"))
##         attributes(xx)[keepAttrs] <- attributes(x)[keepAttrs]
##         print(xx, quote = quote, ...)
##     }
##     maxl <- if (is.null(max.levels))
##         TRUE
##     else max.levels
##     if (maxl) {
##         n <- length(lev <- encodeString(levels(x), quote = ifelse(quote,
##             "₩"", "")))
##         colsep <- if (ord)
##             " < "
##         else " "
##         T0 <- "Levels: "
##         if (is.logical(maxl))
##             maxl <- {
##                 width <- width - (nchar(T0, "w") + 3L + 1L +
##                   3L)
##                 lenl <- cumsum(nchar(lev, "w") + nchar(colsep,
##                   "w"))
##                 if (n <= 1L || lenl[n] <= width)
##                   n
##                 else max(1L, which.max(lenl > width) - 1L)
##             }
##         drop <- n > maxl
##         cat(if (drop)
##             paste(format(n), ""), T0, paste(if (drop)
##             c(lev[1L:max(1, maxl - 1)], "...", if (maxl > 1) lev[n])
##         else lev, collapse = colsep), "₩n", sep = "")
##     }
##     if (!isTRUE(val <- .valid.factor(x)))
##         warning(val)
##     invisible(x)
## }
## <bytecode: 0x0000000013a3aa30>
## <environment: namespace:base>
```

```
num.f <- factor(c(1:3, 3:1), levels = 1:3, labels = c("A", "B", "C"))
num.f
```

```
## [1] A B C C B A
## Levels: A B C
```

```
class(num.f)
```

```
## [1] "factor"
```

```
str(num.f)
```

```
##  Factor w/ 3 levels "A","B","C": 1 2 3 3 2 1
```

```
# methods(print)
```

Method Dispatch

```
print.function
```

```
## function (x, useSource = TRUE, ...)
## print.default(x, useSource = useSource, ...)
## <bytecode: 0x000000001510a5f8>
## <environment: namespace:base>
```

```
summary.matrix
```

```
## function (object, ...)
## {
##     summary.data.frame(as.data.frame.matrix(object), ...)
## }
## <bytecode: 0x00000000125c0b38>
## <environment: namespace:base>
```

```
class(one_play) <- "slots"
one_play
```

```
## [1] 0
## attr(,"symbols")
## [1] "B" "0" "B"
## attr(,"class")
## [1] "slots"
```

```
args(print)
```

```
## function (x, ...)
## NULL
```

```
#> Simple example
print.slots <- function(x, ...) {
  cat("I'm using the print.slots method")
}
print(one_play)
```

```
## I'm using the print.slots method
```

```
one_play
```

```
## I'm using the print.slots method
```

```
rm("print.slots")
#> Scoping Rule
now <- Sys.time()
attributes(now)
```

```
## $class
## [1] "POSIXct" "POSIXt"
```

```
#> UseMethod
print.slots <- function(x, ...) {
  slot_display(x)
}
one_play
```

```
## B 0 B
## $0
```

```
play <- function() {
  symbols <- get_symbols()
  structure(score(symbols), symbols = symbols, class = "slots")
}
class(play())
```

```
## [1] "slots"
```

```
play()
```

```
## B DD 0
## $0
```

```
play()
```

```
## DD B 0
## $0
```

Classes

```
methods(class = "factor")
```

```
##  [1] [             [[            [[<-          [<-           all.equal
##  [6] as.character  as.data.frame as.Date       as.list       as.logical
## [11] as.POSIXlt    as.vector     coerce        droplevels    format
## [16] initialize    is.na<-       length<-      levels<-      Math
## [21] Ops           plot          print         relevel       relist
## [26] rep           show          slotsFromS3   summary       Summary
## [31] xtfrm
## see '?methods' for accessing help and source code
```

```
play1 <- play()
play1
```

```
## 0 BB BB
## $0
```

```
play2 <- play()
play2
```

```
## C 7 BBB
## $2
```

```
c(play1, play2)
```

```
## [1] 0 2
```

```
play1[1]
```

```
## [1] 0
```

```
save.image("./S3.RData")
```

Comments

symbols <- paste(symbols, collapse = " “)를통해 하나의 벡터로 합친후에 paste(symbols, one_play, sep =”$")를 통해 결과값을 토출해낼수 있었습니다. 그다음 cat 함수를통해 우리가 원하는 모양을 만들수있었습니다. 또한 조별과제중 해결해야할 문제중 하나인 상금과 결과값을 도출해 내는것을 할수있게 되었습니다. 점점 게임의 완성도가 높아지는것에 따라 흥미가 더 생기는 수업이였습니다