Assignment_2, Div: CSAI-B, Roll No.: 37

Atharva Salitri

25-01-2025

#Exercise 1

```
s <- 2
p <- 2L

class_s <- class(s)
class_p <- class(p)
cat("Class of s:", class_s, "\n")
```

```
## Class of s: numeric
```

```
cat("Class of p:", class_p, "\n")
```

```
## Class of p: integer
```

```
q <- as.integer(s)


class_q <- class(q)
cat("Class of q:", class_q, "\n")
```

```
## Class of q: integer
```

```
b <- 4/3


as_integer_b <- as.integer(b)
cat("as.integer(b):", as_integer_b, "\n")
```

```
## as.integer(b): 1
```

```
# ii. class(b)
class_b <- class(b)
cat("class(b):", class_b, "\n")
```

```
## class(b): numeric
```

```
# iii. as.numeric(b)
as_numeric_b <- as.numeric(b)
cat("as.numeric(b):", as_numeric_b, "\n")
```

```
## as.numeric(b): 1.333333
```

```
# iv. Use is.integer and is.numeric for b
is_integer_b <- is.integer(b)
is_numeric_b <- is.numeric(b)
cat("is.integer(b):", is_integer_b, "\n")
```

```
## is.integer(b): FALSE
```

```
cat("is.numeric(b):", is_numeric_b, "\n")
```

```
## is.numeric(b): TRUE
```

```
# Step 3
x <- 1
y <- 2

z <- x > y

# Print z and its class
cat("Value of z:", z, "\n")
```

```
## Value of z: FALSE
```

```
cat("Class of z:", class(z), "\n")
```

```
## Class of z: logical
```

```
# Step 4
x <- "My SGPA "
y <- "for last semester is "
z <- 9.12
print(paste(x, y, z))
```

```
## [1] "My SGPA  for last semester is  9.12"
```

#Exercise 2

```
vec_seq <- seq(1, 37, by = 3)
cat("Vector using seq():", vec_seq, "\n")
```

```
## Vector using seq(): 1 4 7 10 13 16 19 22 25 28 31 34 37
```

```
vec_rep <- rep(5, times = 10)
cat("Vector using rep():", vec_rep, "\n")
```

```
## Vector using rep(): 5 5 5 5 5 5 5 5 5 5
```

```
x <- c(1, 5, 2)
y <- c(3, 7, 9)

# i. Augment x by adding y to the left
x <- c(y, x)
cat("Augmented x:", x, "\n")
```

```
## Augmented x: 3 7 9 1 5 2
```

```
# ii. Augment y by adding elements 4, 3, 2 at the end
y <- c(y, 4, 3, 2)
cat("Augmented y:", y, "\n")
```

```
## Augmented y: 3 7 9 4 3 2
```

```
# iii. Find maximum value of y and minimum value of x
max_y <- max(y)
min_x <- min(x)
cat("Maximum value of y:", max_y, "\n")
```

```
## Maximum value of y: 9
```

```
cat("Minimum value of x:", min_x, "\n")
```

```
## Minimum value of x: 1
```

```
x <- c(1, 5, 2, 3, 7, 6, 8)

y <- x^2
z <- 1/x
w <- log10(x)

cat("Vector y (x^2):", y, "\n")
```

```
## Vector y (x^2): 1 25 4 9 49 36 64
```

```
cat("Vector z (1/x):", z, "\n")
```

```
## Vector z (1/x): 1 0.2 0.5 0.3333333 0.1428571 0.1666667 0.125
```

```
cat("Vector w (log10(x)):", w, "\n")
```

```
## Vector w (log10(x)): 0 0.69897 0.30103 0.4771213 0.845098 0.7781513 0.90309
```

```
age <- c(22, 27, 31, 41, 30, 25, 19, 20, 23, 35)

# i. Access age of the fourth person
fourth_age <- age[4]
cat("Age of the fourth person:", fourth_age, "\n")
```

```
## Age of the fourth person: 41
```

```
# ii. Create a vector age30 with age of persons > 30
age30 <- age[age > 30]
cat("Ages greater than 30:", age30, "\n")
```

```
## Ages greater than 30: 31 41 35
```

```
# iii. Access age of the last 3 persons
last3_ages <- tail(age, 3)
cat("Ages of the last 3 persons:", last3_ages, "\n")
```

```
## Ages of the last 3 persons: 20 23 35
```

```
# iv. Find number of elements in the vector age
num_elements <- length(age)
cat("Number of elements in age:", num_elements, "\n")
```

```
## Number of elements in age: 10
```

```
# v. Access ages of persons except the 5th and 7th
ages_except <- age[-c(5, 7)]
cat("Ages except 5th and 7th persons:", ages_except, "\n")
```

```
## Ages except 5th and 7th persons: 22 27 31 41 25 20 23 35
```

```
# vi. Create a vector age2 with ages between 20 and 25
age2 <- age[age >= 20 & age <= 25]
cat("Ages between 20 and 25:", age2, "\n")
```

```
## Ages between 20 and 25: 22 25 20 23
```

Exercise 3

```
ls <- list(
  Rollno = 1:4,
  FirstName = c("Ravi", "Om", "Ajay", "Shiv"),
  LastName = c("Dev", "Gandhi", "Pande", "Rao"),
  Subject = c("AE", "DS", "ML", "OS"),
  Marks = c(35, 40, 38, 2),
  Result = c("P", "P", "P", "F")
)
```

```
print(ls)
```

```
## $Rollno
## [1] 1 2 3 4
##
## $FirstName
## [1] "Ravi" "Om"   "Ajay" "Shiv"
##
## $LastName
## [1] "Dev"    "Gandhi" "Pande"  "Rao"
##
## $Subject
## [1] "AE" "DS" "ML" "OS"
##
## $Marks
## [1] 35 40 38  2
##
## $Result
## [1] "P" "P" "P" "F"
```

```
print(ls$Rollno)
```

```
## [1] 1 2 3 4
```

```
print(ls$FirstName)
```

```
## [1] "Ravi" "Om"   "Ajay" "Shiv"
```

```
print(ls$LastName)
```

```
## [1] "Dev"    "Gandhi" "Pande"  "Rao"
```

```
print(ls$Subject)
```

```
## [1] "AE" "DS" "ML" "OS"
```

```
print(ls$Marks)
```

```
## [1] 35 40 38  2
```

```
print(ls$Result)
```

```
## [1] "P" "P" "P" "F"
```

```
sapply(ls, class)
```

```
##      Rollno   FirstName    LastName     Subject       Marks      Result
##   "integer" "character" "character" "character"   "numeric" "character"
```

```
# a) Print(ls[[2]][1]) - First name of the first student
print(ls[[2]][1])
```

```
## [1] "Ravi"
```

```
# b) print(ls[[4]][4]) - Subject of the fourth student
print(ls[[4]][4])
```

```
## [1] "OS"
```

```
# c) print(ls[5]) - List containing Marks
print(ls[5])
```

```
## $Marks
## [1] 35 40 38  2
```

```
ls$Marks[3] <- 45
print(ls$Marks)
```

```
## [1] 35 40 45  2
```

```
ls$Subject[4] <- "OE"
print(ls$Subject)
```

```
## [1] "AE" "DS" "ML" "OE"
```

```
ls$NativePlace <- c("Pune", "Nagpur", "Mumbai", "Nashik")
print(ls$NativePlace)
```

```
## [1] "Pune"   "Nagpur" "Mumbai" "Nashik"
```

```
ls$Rollno <- c(ls$Rollno, 5)
ls$FirstName <- c(ls$FirstName, "Julie")
ls$LastName <- c(ls$LastName, "Gommes")
ls$Subject <- c(ls$Subject, "DS")
ls$Marks <- c(ls$Marks, 30)
ls$Result <- c(ls$Result, "P")
ls$NativePlace <- c(ls$NativePlace, "Hyderabad")

print(ls)
```

```
## $Rollno
## [1] 1 2 3 4 5
##
## $FirstName
## [1] "Ravi"  "Om"    "Ajay"  "Shiv"  "Julie"
##
## $LastName
## [1] "Dev"    "Gandhi" "Pande"  "Rao"    "Gommes"
##
## $Subject
## [1] "AE" "DS" "ML" "OE" "DS"
##
## $Marks
## [1] 35 40 45  2 30
##
## $Result
## [1] "P" "P" "P" "F" "P"
##
## $NativePlace
## [1] "Pune"      "Nagpur"    "Mumbai"    "Nashik"    "Hyderabad"
```

Exercise 4

```
x <- list(
  n = c(2, 3, 5),
  s = c("aa", "bb", "cc", "dd"),
  b = c(TRUE, FALSE, TRUE, FALSE),
  value = 3
)
```

```
print(x[["s"]])
```

```
## [1] "aa" "bb" "cc" "dd"
```

```
print(x[c(2, 4)])
```

```
## $s
## [1] "aa" "bb" "cc" "dd"
##
## $value
## [1] 3
```

```
print(x$s)
```

```
## [1] "aa" "bb" "cc" "dd"
```

```
x$s[x$s == "aa"] <- "tt"

print(x$s)
```

```
## [1] "tt" "bb" "cc" "dd"
```

Exercise 5

```
A <- matrix(c(5, 0, 3, 1,
              2, 6, 8, 8,
              6, 2, 1, 5,
              1, 0, 4, 6),
            nrow = 4, byrow = TRUE)

B <- matrix(c(7, 1, 9, 5,
              5, 8, 4, 3,
              8, 2, 3, 7,
              0, 6, 8, 9),
            nrow = 4, byrow = TRUE)

cat("Matrix A:\n"); print(A)
```

```
## Matrix A:
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    5    0    3    1
## [2,]    2    6    8    8
## [3,]    6    2    1    5
## [4,]    1    0    4    6
```

```
cat("Matrix B:\n"); print(B)
```

```
## Matrix B:
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    7    1    9    5
## [2,]    5    8    4    3
## [3,]    8    2    3    7
## [4,]    0    6    8    9
```

```
largest_A <- max(A)
smallest_B <- min(B)
cat("Largest number in A:", largest_A, "\n")
```

```
## Largest number in A: 8
```

```
cat("Smallest number in B:", smallest_B, "\n")
```

```
## Smallest number in B: 0
```

```
c <- A[2, 3]
cat("2nd row, 3rd column element of A:", c, "\n")
```

```
## 2nd row, 3rd column element of A: 8
```

```
D <- B[4, ]
cat("Row 4 of B:", D, "\n")
```

```
## Row 4 of B: 0 6 8 9
```

```
largest_last_col_B <- max(B[, ncol(B)])
cat("Largest number in the last column of B:", largest_last_col_B, "\n")
```

```
## Largest number in the last column of B: 9
```

```
# Transpose of A
transpose_A <- t(A)
cat("Transpose of A:\n"); print(transpose_A)
```

```
## Transpose of A:
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    5    2    6    1
## [2,]    0    6    2    0
## [3,]    3    8    1    4
## [4,]    1    8    5    6
```

```
if (det(B) != 0) {
  inverse_B <- solve(B)
  cat("Inverse of B:\n"); print(inverse_B)
} else {
  cat("Matrix B is not invertible.\n")
}
```

```
## Inverse of B:
##             [,1]        [,2]        [,3]        [,4]
## [1,]  0.03488086  0.04347826  0.06730533 -0.08621960
## [2,] -0.05919921  0.13043478 -0.02972243  0.01252763
## [3,]  0.13559322  0.00000000 -0.11864407  0.01694915
## [4,] -0.08106116 -0.08695652  0.12527634  0.08769344
```