Bios 301: Assignment 1

Danielle Nemetz

October 7, 2013

1 Working with data

In the `datasets` folder on the course GitHub repo, you will find a file called `cancer.csv`, which is a dataset in comma-separated values (csv) format. This is a large cancer incidence dataset that summarizes the incidence of different cancers for various subgroups. (18 points)

1. Load the data set into R and make it a data frame called cancer.df. (2 points)

```
setwd("/Users/Danie/Bios301/datasets")  #setting working directory
cancer.df <- read.csv("cancer.csv")  #reading .csv file
```

2. Determine the number of rows and columns in the data frame. (2)

```
nrow(cancer.df)  #No. rows

## [1] 42120
```

```
ncol(cancer.df)  #No. columns

## [1] 8
```

3. Extract the names of the columns in cancer.df. (2)

```
names(cancer.df)

## [1] "year"       "site"       "state"      "sex"        "race"
## [6] "mortality"  "incidence"  "population"
```

4. Report the value of the 3000th row in column 6. (2)

```
cancer.df[3000, 6]

## [1] 350.7
```

5. Report the contents of the 172nd row. (2)

```
cancer.df[172, ]

##     year                            site  state  sex  race mortality
## 172 1999 Brain and Other Nervous System nevada Male Black         0
##     incidence population
## 172         0      73172
```

6. Create a new column that is the incidence *rate* (per 100,000) for each row.(3)

```
cancer.df["incidence rate"] <- cancer.df$incidence/cancer.df$population * 1e+05
```

7. How many subgroups (rows) have a zero incidence rate? (2)

```
nrow(subset(cancer.df, cancer.df$"incidence rate" == "0"))

## [1] 23191
```

8. Find the subgroup with the highest incidence rate.(3)

```
which.max(cancer.df$"incidence rate")

## [1] 5797
```

2 Data types

1. Create the following vector: `x <- c("5","12","7")`. Which of the following commands will produce an error message? For each command, Either explain why they should be errors, or explain the non-erroneous result. (6 points)

```
max(x)
sort(x)
sum(x)
```

```
x <- c("5", "12", "7")
max(x)

## [1] "7"

sort(x)

## [1] "12" "5"  "7"

sum(x)

## Error:  invalid 'type' (character) of argument
```

The functions max and sort operate by looking at the numerical representations of the first character in the string. The character "5" is observed as 5 in decimal, "12" is observed as 1 in decimal and "7" is observed as 7 in decimal. Therefore, the max function returns 7 and the sort function returns the characters in ascending order by its numerical representations: 12, 5, 7. The last command results in an error because it only accepts numeric values.

2. For the next two commands, either explain their results, or why they should produce errors. (4 points)

```
y <- c("5",7,12)
y[2] + y[3]
```

```
y <- c("5", 7, 12)
y[2] + y[3]

## Error:  non-numeric argument to binary operator
```

The first command assigns a vector to y . The second command produces an error because a vector can only have one mode. Because the first input is a character, the following two inputs are converted to characters as well. This is confirmed by the class function. Similar to the sum function in the previous problem, the second command cannot be performed because two characters cannot be added by the binary operator.

```
class(y)

## [1] "character"
```

3. For the next two commands, either explain their results, or why they should produce errors. (4 points)

```
z <- data.frame(z1="5",z2=7,z3=12)
z[1,2] + z[1,3]
```

```
z <- data.frame(z1 = "5", z2 = 7, z3 = 12)
z[1, 2] + z[1, 3]

## [1] 19

class(z)

## [1] "data.frame"
```

The first command assigns a data frame containing a string and two numeric values to the variable z. The second command adds the first row, second value (7) to the first row, third value (12). Because both values are numeric (verified via class function), they can be added.

3 Data structures

Give R expressions that return the following matrices and vectors (*i.e.* do not construct them manually). (3 points each, 12 total)

1. $(1, 2, 3, 4, 5, 6, 7, 8, 7, 6, 5, 4, 3, 2, 1)$

```
c(1:8, 7:1)

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

2. $(1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5)$

```
c(rep(1, 1), rep(2, 2), rep(3, 3), rep(4, 4), rep(5, 5))

##  [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
```

3. $\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$

```
m1 <- matrix(0, 3, 3)
m1[lower.tri(m1)] <- 1
m1[upper.tri(m1)] <- 1
m1[2, 1] = 0
m1

##      [,1] [,2] [,3]
## [1,]    0    1    1
## [2,]    0    0    1
## [3,]    1    1    0
```

4. $\begin{pmatrix} 0 & 2 & 3 \\ 0 & 5 & 0 \\ 7 & 0 & 0 \end{pmatrix}$

```
m2 <- matrix(1:9, 3, 3, byrow = TRUE)
m2[1, 1] = 0
m2[2, 1] = 0
m2[2, 3] = 0
m2[3, 2] = 0
m2[3, 3] = 0
m2

##      [,1] [,2] [,3]
## [1,]    0    2    3
## [2,]    0    5    0
## [3,]    7    0    0
```

4 Basic programming

Let $h(x,n) = 1 + x + x^2 + \ldots + x^n = \sum_{i=0}^{n} x_i$. Write an R program to calculate $h(x,n)$ using a `for` loop. (6 points)

```
h = function(x, n) {
    S = 1
    for (i in 1:n) {
        S = x^i + S
    }
    return(S)
}
h(4, 5)  #Test of function with x=4, n=5

## [1] 1365
```

```r
1 + sum(4^c(1:5))  #Actual

## [1] 1365
```