Please find answers to practical session

WS1 Practical session answers

1. The number of records in the data set.

   dim(gapminder)
2. The number of attributes for a record.
   dim(gapminder)

3. The column names.
   str(gapminder)
4. The data type of each attribute
   str(gapminder)

5. What are the values for the first 10 records.
   head(gapminder,10)
6. What are the values for the last record in the dataset.
7. tail(gapminder,1)

Correct these common subsetting errors.

1. Aim: Extract observations collected for the year 1957

gapminder[gapminder$year == 1957,]

2. Aim: Extract all columns except 1 through to 4

gapminder[,-c(1:4)]

3. Aim: Extract the rows where the life expectancy is longer the 80 years

gapminder[gapminder$lifeExp > 80,]

4. Aim: Extract the first row, and the fourth and fifth columns (continent and lifeExp).

gapminder[1, c(4, 5)]

5. Aim: Extract rows that contain information for the years 2002 and 2007 (advanced)

gapminder[gapminder$year == 2002 | gapminder$year == 2007,] or

gapminder[gapminder$year %in% c(2002, 2007),]

1. Why does gapminder[1:20]return an error?

Needs two indexs.

2. Create a new data.frame called gapminder_small that only contains rows 1 through 9 and 19 through 23. You can do this in one or two steps?

```
gapminder_small <- gapminder[c(1:9, 19:23),]
```

3. Selecting elements of a vector that match any of a list of components is a very common data analysis task. For example, the gapminder data set contains country and continent variables, but no information between these two scales. Suppose we want to pull out information from southeast Asia. How do we set up an operation to produce a logical vector that is TRUE for all of the countries in southeast Asia and FALSE otherwise? For this question we would consider Myanmar, Thailand, Cambodia, Vietnam, and Laos as SE Asia. (advanced)

Hint: ?unique() may be helpful here.

```
seAsia <- c("Myanmar","Thailand","Cambodia","Vietnam","Laos")

countries <- unique(as.character(gapminder$country))

logicalvec <- seAsia %in% countries
```

1. Use an **if()** statement to print a suitable message reporting whether there are any records from 2002 in the gapminder dataset.

```
if(nrow(gapminder[(gapminder$year == 2002),]) >= 1){

  print("Record(s) for the year 2002 found.")

}


if(any(gapminder$year == 2002)){

  print("Record(s) for the year 2002 found.")

}
```

2. Modify your code to remove hardcoded variables so that you can provide any year to be checked.

```
if(nrow(gapminder[(gapminder$year == year),]) >= 1){

  print(paste("Record(s) found for the year", year))

}


if(any(gapminder$year == year)){

  print(paste("Record(s) found for the year", year))

}
```

3. Now modify your code so you can check two years at the same time.

```
if(any(gapminder$year == year1)|| any(gapminder$year == year2){
  print(paste("Record(s) found”)}
```

Create a function

Define a function that calculates and returns the Gross Domestic Product of a nation (GDP * POP) from the data available in the dataset.

```
calcGDP <- function(dat) {
  gdp <- dat$pop * dat$gdpPercap
  return(gdp)
}
```

Now modify your function so that it returns the GDP as a new column on the data set.

```
calcGDP <- function(dat) {
  gdp <- dat$pop * dat$gdpPercap
 new <- cbind(dat, gdp=gdp)
return(new)


}
```

Now modify your function so that if given a country **or** a year it returns just the data for this selection (e.g., You could specify 'Australia' and get back all years for Australia, you could specify '2007' and get all countries in 2007, or you could specify 'Australia' in '2007' and just receive the data for Australia in 2007).

```
calcGDP <- function(dat, year=NULL, country=NULL) {
  if(!is.null(year)) {
    dat <- dat[dat$year %in% year, ]
  }
  if (!is.null(country)) {
    dat <- dat[dat$country %in% country,]
  }
  gdp <- dat$pop * dat$gdpPercap


  new <- cbind(dat, gdp=gdp)
  return(new)
}
```

Add defensive programming

What happens if we pass unexpected values into your function? (Try putting numeric values in as a country or logical value in as a year)

Add some additional checks into your function to make sure that the correct arguments are being passed in. Hint: you could check out ?is.numeric, ?is.character, and ?stopifnot().

```
calcGDP <- function(dat, year=NULL, country=NULL) {
  if(!is.null(year)) {
stopifnot(is.numeric(year))

    dat <- dat[dat$year %in% year, ]
  }
  if (!is.null(country)) {
stopifnot(is.character(country))

    dat <- dat[dat$country %in% country,]
  }
  gdp <- dat$pop * dat$gdpPercap

  new <- cbind(dat, gdp=gdp)
  return(new)
}
```

Manipulating data and simple plots

1. Make a new column in the gapminder data frame that contains population in units of millions of people.

```
gapminder$pop_millions <- gapminder$pop / 1e6
```

2. On a single graph, plot population, in millions, against year, for all countries. Do not worry about identifying which country is which.

```
ggplot(gapminder, aes(x = year, y = pop_millions)) +

 geom_point()
```

3. Repeat the exercise, graphing only for China, India, and Indonesia. Again, do not worry about which is which.

```
countryset <- c("China","India","Indonesia")

ggplot(gapminder[gapminder$country %in% countryset,],

       aes(x = year, y = pop_millions)) +

  geom_point()
```

More plotting

1. Using **ggplot2**, plot a scatter plot of **GDP** against **life expectancy**.

```
ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +

  geom_point()
```

2. Now modify the plot to show how **life expectancy** has changed over **time**.

```
ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp)) + geom_point()
```

3. Now colour code to the points to show **continent**.

```
ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp, color=continent)) +

  geom_point()
```

2. Scatter plot might not be the best option for showing change over time. Change it to a **line plot**. Hint: you probably want to consider a new argument **'group'**.

```
ggplot(data = gapminder, mapping = aes(x=year, y=lifeExp, group=country, color=continent)) +

  geom_line()
```