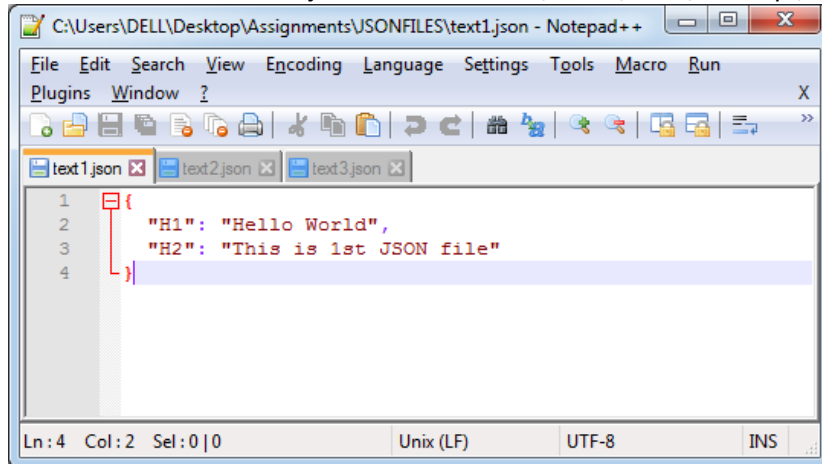


## SESSION 2: Introduction to working with R

### Assignment 2

1. Read multiple json files into a working directory for further converting into a dataset. I have files text1, text2, text3 in the directory json.

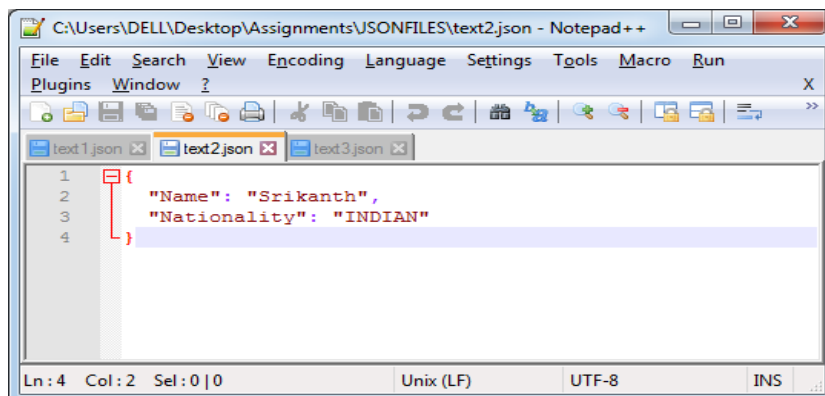
**Answer:** Created these 3 json files under “C:\Users\DELL\Desktop\Assignments\JSONFILES” folder



A screenshot of a Notepad++ window titled "C:\Users\DELL\Desktop\Assignments\JSONFILES\text1.json - Notepad++". The window shows a JSON file named text1.json with the following content:

```
1 {  
2   "H1": "Hello World",  
3   "H2": "This is 1st JSON file"  
4 }
```

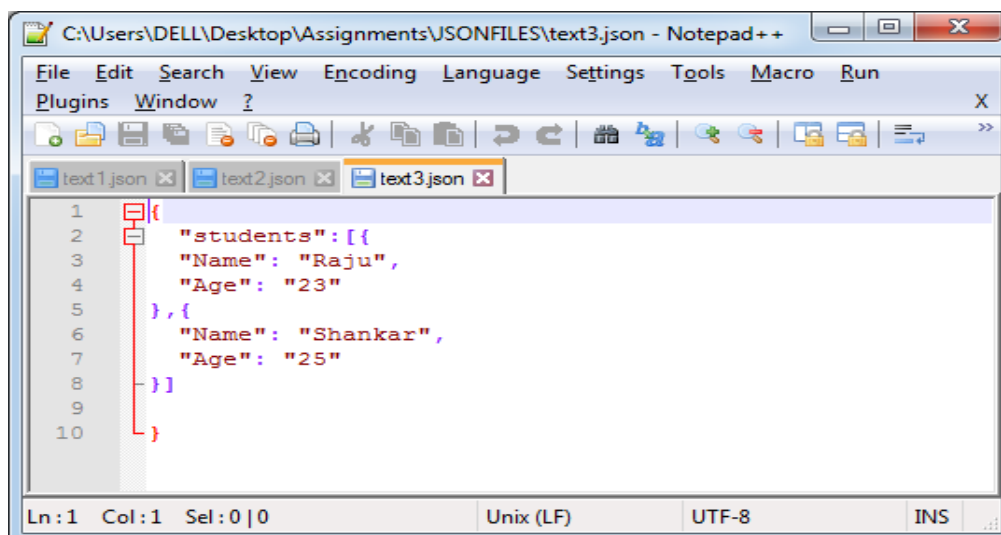
The status bar at the bottom indicates "Ln: 4 Col: 2 Sel: 0 | 0", "Unix (LF)", "UTF-8", and "INS".



A screenshot of a Notepad++ window titled "C:\Users\DELL\Desktop\Assignments\JSONFILES\text2.json - Notepad++". The window shows a JSON file named text2.json with the following content:

```
1 {  
2   "Name": "Srikanth",  
3   "Nationality": "INDIAN"  
4 }
```

The status bar at the bottom indicates "Ln: 4 Col: 2 Sel: 0 | 0", "Unix (LF)", "UTF-8", and "INS".



A screenshot of a Notepad++ window titled "C:\Users\DELL\Desktop\Assignments\JSONFILES\text3.json - Notepad++". The window shows a JSON file named text3.json with the following content:

```
1 {  
2   "students": [{  
3     "Name": "Raju",  
4     "Age": "23"  
5   }, {  
6     "Name": "Shankar",  
7     "Age": "25"  
8   }]  
9 }  
10 }
```

The status bar at the bottom indicates "Ln: 1 Col: 1 Sel: 0 | 0", "Unix (LF)", "UTF-8", and "INS".

Installed the package rjson

Set the working directory to the folder path where all the Json files are present

```
setwd("C://Users//DELL//Desktop//Assignments//JSONFILES")
```

Stored all the json files name to vector x using list.files function passing "\*.json" as the pattern

```
x <-list.files(pattern="*.json")
```

using lapply function we are storing the content of json file one by one in dataframe l

```
l<-lapply(x,function(x) fromJSON(file=x))
```

```
install.packages("rjson")
```

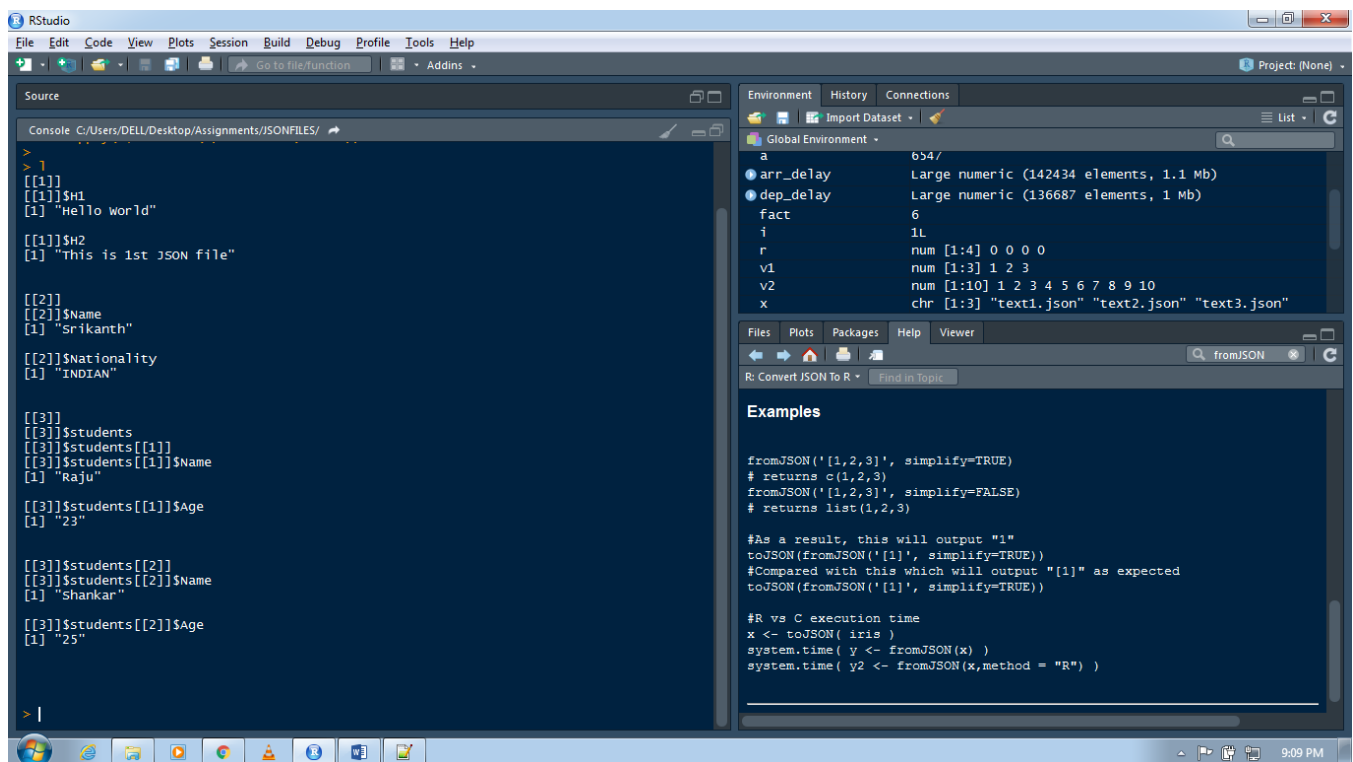
```
library(rjson)
```

```
setwd("C://Users//DELL//Desktop//Assignments//JSONFILES")
```

```
x <-list.files(pattern="*.json")
```

```
l<-lapply(x,function(x) fromJSON(file=x))
```

```
l
```



Converting the list to dataframe in below script

```
df<-as.data.frame(do.call("cbind", l))
```

```
df
```

```
> df<-as.data.frame(do.call("cbind", l))
> df
```

	v1	v2	v3
H1	Hello world	Srikanth Raju, 23, Shankar,	25
H2	This is 1st JSON file	INDIAN Raju, 23, Shankar,	25

2. Parse the following JSON into a data frame

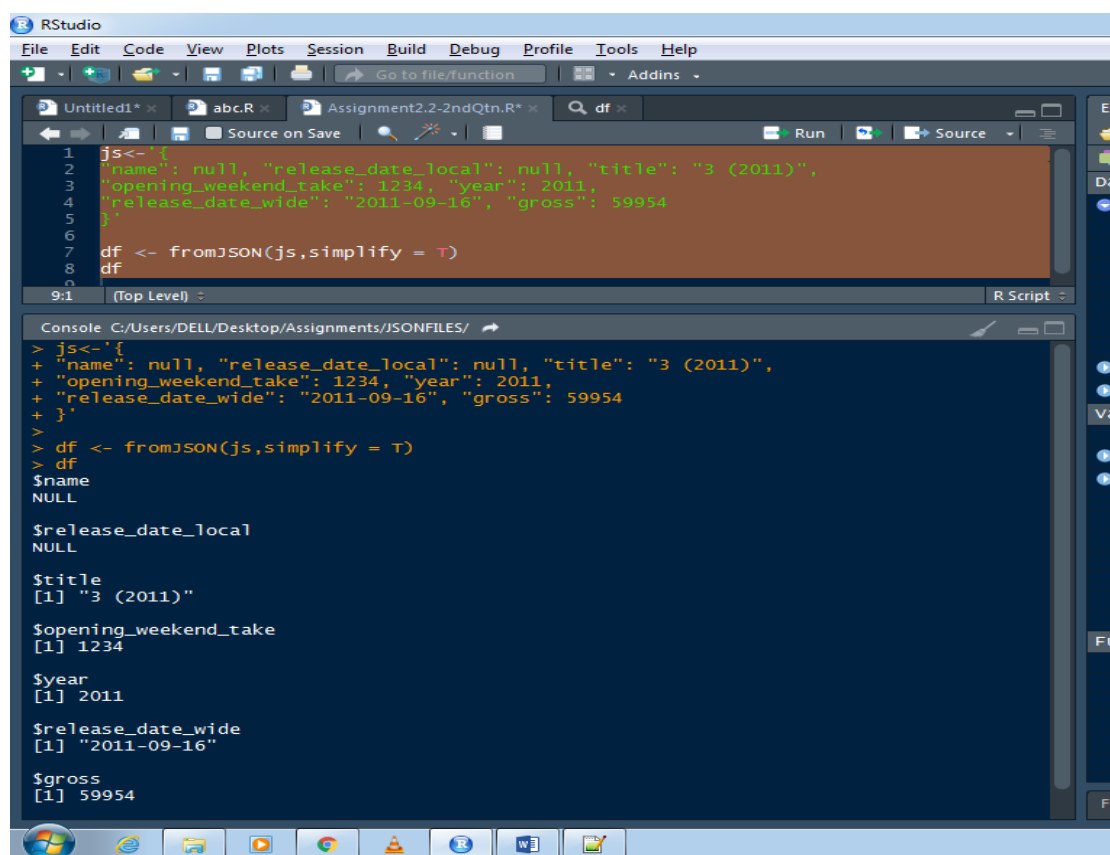
```
js<-'{
"name": null, "release_date_local": null, "title": "3 (2011)",
"opening_weekend_take": 1234, "year": 2011,
"release_date_wide": "2011-09-16", "gross": 59954
}'
```

**Answer:**

```
js<-'{
"name": null, "release_date_local": null, "title": "3 (2011)",
"opening_weekend_take": 1234, "year": 2011,
"release_date_wide": "2011-09-16", "gross": 59954
}'
```

```
df <- fromJSON(js)
```

```
df
```

The screenshot shows the RStudio interface. The script editor contains the following code:

```
1 js<- '{
2   "name": null, "release_date_local": null, "title": "3 (2011)",
3   "opening_weekend_take": 1234, "year": 2011,
4   "release_date_wide": "2011-09-16", "gross": 59954
5 }'
6
7 df <- fromJSON(js,simplify = T)
8 df
```

The console shows the output of the code execution:

```
> js<- '{
+ "name": null, "release_date_local": null, "title": "3 (2011)",
+ "opening_weekend_take": 1234, "year": 2011,
+ "release_date_wide": "2011-09-16", "gross": 59954
+ }'
>
> df <- fromJSON(js,simplify = T)
> df
$name
NULL

$release_date_local
NULL

$title
[1] "3 (2011)"

$opening_weekend_take
[1] 1234

$year
[1] 2011

$release_date_wide
[1] "2011-09-16"

$gross
[1] 59954
```

Converting list to dataframe

```
df<-as.data.frame(do.call("cbind", df))
```

```
df
```

```
> df<-as.data.frame(do.call("cbind", df))
> df
  title opening_weekend_take year release_date_wide gross
1 3 (2011)             1234 2011      2011-09-16 59954
```

3. Write a script for variable binning using R.

**Answer:** Bins are created on continuous and categorical variables

Let's create a vector **age** as below

```
age <- c(4,7,5,9,1,10,15,18,19,3,16,10,16,12,22,2,23,16,17)
```

Vector **age** contains various elements of different age group.

To group them or to categorize them under certain range we use cut function

```
cut(age, c(1,5,10,15,25))
```

```
> cut(age, c(1,5,10,15,25))  
[1] (1,5] (5,10] (1,5] (5,10] <NA> (5,10] (10,15] (15,25] (15,25] (1,5]  
[11] (15,25] (5,10] (15,25] (10,15] (15,25] (1,5] (15,25] (15,25] (15,25]  
Levels: (1,5] (5,10] (10,15] (15,25]
```

Here we are grouping the vector contents into below bin or group or categories

[1,5]

[5,10]

[10,15]

[15,25]

So the above vector values fall under these bins using the **data.frame()** along with **cut()**

```
age <- c(4,7,5,9,1,10,15,18,19,3,16,10,16,12,22,2,23,16,17)
```

```
data.frame(age, bin=cut(age, c(1,5,10,15,25), include.lowest=TRUE))
```

```
> age <- c(4,7,5,9,1,10,15,18,19,3,16,10,16,12,22,2,23,16,17)  
> data.frame(age, bin=cut(age, c(1,5,10,15,25), include.lowest=TRUE))  
  age bin  
1   4 [1,5]  
2   7 (5,10]  
3   5 [1,5]  
4   9 (5,10]  
5   1 [1,5]  
6  10 (5,10]  
7  15 (10,15]  
8  18 (15,25]  
9  19 (15,25]  
10  3 [1,5]  
11 16 (15,25]  
12 10 (5,10]  
13 16 (15,25]  
14 12 (10,15]  
15 22 (15,25]  
16  2 [1,5]  
17 23 (15,25]  
18 16 (15,25]  
19 17 (15,25]
```

Here we can see each age is falling into respecting bin.