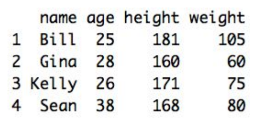
Session 4 - dplyr

R: Please re-write the following expression using the pipe (`%>%`) operator.

```
mean(is.na(age))
```

R: suppose you have a data frame `patients` with the following contents:

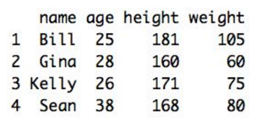


```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you are only interested in the `height` and `weight` columns.

Using the pipe operator and appropriate dplyr functions, create a data frame `heights_weights` which contains the `height` and `weight` columns of `patients`.

R: suppose you have a data frame `patients` with the following contents:



```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you'd like to de-identify the data set.

Using the pipe operator and appropriate dplyr functions, create a data frame `patients_deidentified` which contains all the same columns as `patients` except `name`.

R: suppose you have a data frame `patients` with the following contents:

```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you're only interested in patients younger than 30 years old.

Using the pipe operator and appropriate dplyr functions, create a data frame `patients_less_than_30` which contains only the rows of `patients` whose `age` is less than `30`.

R: suppose you have a data frame `patients` with the following contents:

```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you're interested in patients by the name Bill or Sean.

Using the pipe operator and appropriate dplyr functions, create a data frame `bill_or_sean` which contains only the rows of `patients` whose `name` is either `Bill` or `Sean`.

R: suppose you have a data frame `patients` with the following contents:

```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you'd like to sort the patients by age.

Using the pipe operator and appropriate dplyr functions, create a data frame `patients_by_age` which contains the same data as `patients` but is sorted in ascending order, by `age`.

R: suppose you have a data frame `patients` with the following contents:

```
   name age height weight
1  Bill  25    181    105
2  Gina  28    160     60
3 Kelly  26    171     75
4  Sean  38    168     80
```

Assume you'd like to add column containing the body mass index (BMI) of each patient. The BMI is the weight (in kg) divided by the height (in m) squared.

Using the pipe operator and appropriate dplyr functions, modify the data frame `patients` to add a column `bmi` which is equal to `weight / (height / 100) ^ 2`

R: suppose you have a data frame `patients` with the following contents:

```
   name age    sex height weight  bmi
1  Bill  25   male    181    105 32.1
2  Gina  28 female    160     60 23.4
3 Kelly  26 female    171     75 25.6
4  Sean  38   male    168     80 28.3
```

Let's say we're interested in the mean and standard deviation of the BMI of the patients, broken down by sex.

Using the pipe operator and appropriate dplyr functions, do the following:

1. Group the `patients` data by `sex`, then
2. Create a data frame with the following summary statistics:
    1. `n`: the number of cases in each group
    2. `bmi_mean`: the mean of `bmi` of each group
    3. `bmi_sd`: the standard deviation of `bmi` of each group

    The resulting table should look like this:

```
     sex     n bmi_mean   bmi_sd
   <chr> <int>    <dbl>    <dbl>
1 female     2     24.5 1.555635
2   male     2     30.2 2.687006
```

R: suppose you have the following two data frames:

```
> bmi_data                > patients
  id age    sex  bmi        id  name
   2  28 female 23.4         1  Bill
   3  26 female 25.6         2  Gina
   4  38   male 28.3         3 Kelly
   1  25   male 32.1         4  Sean
```

Let's say we want to augment `bmi_data` with a column `name` containing the patient name, identified by the column `id` that is present in both tables.

Join `bmi_data` and `patients` to add a new column `name` to `bmi_data`, using `id` as the *key* to join on. Use the pipe operator where appropriate. Overwrite `bmi_data` with the result.