Format:

In order to receive full credit for the practice problems, you need to submit an R script file. All the non-R codes need to be placed after the # signs. You also need to write your name and *Assignment 2* on top of the page. Please make sure to include your name as part of the file name.

To complete this assignment, you need to use what you've learned from this course. You might *not* receive credits if you use the functions that are not used in this course.

You can only ask questions for clarification purposes on the Discussion Board. That is to say, you can't ask questions about how to answer a specific question or help you to debug your program.

Data

You need to load Assignment2.RData to complete this assignment:

```
> load("Assignment2.RData")
> ls()

[1] "chol"    "patient" "sbp"
```

Problems 1: 5 points

You will use the data frame, `sbp`, for this problem.

```
> head(sbp)

      id visit sbp
1   125F     1 122
2 13000M     1  NA
3 13120M     1 116
4 13260M     1 122
5 13480M     1  NA
6 13520M     1 132
```

This dataset contains SBP (Systolic Blood Pressure) measurements for each patient. Some patients were measured once and some were measured more than once. The description of each variable is described below:

VARIABLE	DESCRIPTION
id	Patient ID
visit	The visiting time
sbp	Systolic blood pressure

The gender of each patient can be identified from the last field of the ID variable with 'M' for Male or 'F' for Female. The gender field can be in either upper or lower cases. Some of the IDs start with an 'x' and some don't. For example, 'x13260M' and '13260M' refer to the same person.

Write a function, `getMaxSBP`, that takes one argument, which is the name of the input data. The function returns a data frame that contains only one observation for each patient with the following variables:

newID : is created by using the numerical fields of the ID variable. For example, if ID is 'x13260M' or '13260M', then NEWID is '13260'.

Sex : is created by using the last field of the ID variable, with values equals 'M' or 'F' (all in upper cases).

maxsbp : is the maximum SBP for each patient. If an SBP is measured greater than 300, then the SBP value should be considered as invalid. In this case, the SBP is considered as the missing value. Some patients might only contain one missing value for this variable.

maxvisit : is the visiting time that corresponds to the maximum SBP when the patient was measured.

For example, consider the following scenarios:

- If a patient was measured three times with
  - SBP = 120 when visit = 1
  - SBP = 142 when visit = 2
  - SBP = 132 when visit = 3

  then MAXSBP = 142 and MAXVISIT = 2
- If a patient was measured three times with
  - SBP = 120 when visit = 1
  - SBP = 305 when visit = 2
  - SBP = 132 when visit = 3

  then MAXSBP = 132 and MAXVISIT = 3
- If a patient was measured three times with
  - SBP = 120 when visit = 1
  - SBP = NA when visit = 2
  - SBP = 132 when visit = 3,

  then MAXSBP = 132 and MAXVISIT = 3
- If a patient was measured once with SBP equals either NA or a value greater than 300, then MAXSBP = NA and MAXVISIT = 1
- If a patient was measured twice with
  - SBP = NA when visit = 1
  - SBP = 320 when visit = 2

  then MAXSBP = NA and MAXVISIT = 1 or 2

The Resulting data set looks like the one below:

```
> getMaxSBP (sbp)

   newID Sex maxvisit maxsbp
1    125   F        1    122
2  13000   M        1     NA
3  13120   M        1    116
4  13260   M        1    122
5  13480   M        2    124
6  13520   M        1    132
7  13580   M        3    124
8   1750   F        1    120
9   2000   F        1     NA
10 21300   F        1    120
11 21525   F        1    124
12 24950   F        2    130
13  4050   F        1     NA
14  4250   F        1     NA
15  4300   F        1    124
16  5200   F        1    116
17  5925   F        2    118
18  7200   F        1     NA
19  8120   M        1    126
20  9020   M        1    120
21  9380   M        1    128
22  9450   F        2    118
23  9500   F        1    124
24  9775   F        1    122
25  9800   M        2    124
26  9900   M        2    122
```

Problems 2: 5 points

You will use the data frame, `chol`, for this problem.

```
> head(chol)

     sex age chol  tg    ht     wt sbp dbp vldl hdl ldl      bmi
id.2   M  60  137  50 68.25 111.75 110  70   10  53  74 2.399066
id.3   M  26  154 202 82.75 184.75  88  64   34  31  92 2.698040
id.4   M  33  198 108 64.25 147.00 120  80   22  34 132 3.560993
id.5   F  27  154  47 63.25 129.00 110  76    9  57  88 3.224547
id.6   M  36  212  79 67.50 176.25 130 100   16  37 159 3.868313
id.7   F  31  197  90 64.50 121.00 122  78   18  58 111 2.908479
```

To test the correlation between two continuous variables, you can use the `cor.test` function. For example:

```
> age_chol <- cor.test(chol$age, chol$chol)
> str(age_chol)
```

```
List of 9
 $ statistic  : Named num 3.13
  ..- attr(*, "names")= chr "t"
 $ parameter  : Named int 188
  ..- attr(*, "names")= chr "df"
 $ p.value    : num 0.00202
 $ estimate   : Named num 0.223
  ..- attr(*, "names")= chr "cor"
 $ null.value : Named num 0
  ..- attr(*, "names")= chr "correlation"
 $ alternative: chr "two.sided"
 $ method     : chr "Pearson's product-moment correlation"
 $ data.name  : chr "chol$age and chol$chol"
 $ conf.int   : atomic [1:2] 0.0829 0.3538
  ..- attr(*, "conf.level")= num 0.95
 - attr(*, "class")= chr "htest"
```

To extract the correlation coefficient and the correponding p-value, you can write the following:

```
> age_chol$estimate

      cor
0.2226466

> age_chol$p.value

[1] 0.002018013
```

Write a function named `myCorTest`, which is used to calculate the pairwise correlation between one variable with a list of given variables. This function takes three arguments:

`dat` : the name of the data frame, such `chol`.

`mainVar` : a character vector of length 1 that contains the name of a continuous variable. For example: `"wt"`. You will calculate the correlation between this variable with each of the variables in the third argument.

`varlist` : a character vector contains one or more values. This argument contains the names of continuous variable.

The function will return a data frame that contains the correlation coefficient and the corresponding p-value between each pair. For example, here are some sample results that are based on the `myCortest` function:

```
> myCortest (chol, "wt", "age")

    var1 var2         R            p
age   wt  age 0.6660014 5.631448e-26

> myCortest (chol, "wt", c("age", "chol", "tg", "ht"))
```

```
     var1 var2          R            p
age    wt  age 0.66600144 5.631448e-26
chol   wt chol 0.06076105 4.049710e-01
tg     wt   tg 0.29461701 3.688497e-05
ht     wt   ht 0.85583311 2.705222e-56
```

> myCortest (chol, "bmi", c("sbp", "dbp", "vldl", "hdl", "ldl"))

```
     var1 var2           R            p
sbp   bmi  sbp  0.14927952 3.877523e-02
dbp   bmi  dbp  0.42636371 6.997094e-10
vldl  bmi vldl  0.41033688 4.107925e-09
hdl   bmi  hdl -0.11984422 9.956239e-02
ldl   bmi  ldl  0.03449137 6.366170e-01
```

Problems 3: 5 points

You will use the data frame, `patient`, for this problem.

> patient

```
   ID GLUC TGL HDL LDL  HRT MAMM SMOKE
1   A   88  NA  32  99    Y <NA>  ever
2   B   NA 150  60  NA <NA>   no never
3   C  110  NA  NA 120    N <NA>  <NA>
4   D   NA 200  65 165 <NA>  yes never
5   E   90 210  NA 150    Y <NA> never
6   F   88  NA  32 210 <NA>  yes  ever
7   G  120 164  NA  NA    Y  yes  <NA>
8   H  110 170  70 188 <NA> <NA>  ever
9   I   NA 190  NA 190    N   no  <NA>
10  J   90  NA  75  NA <NA>  yes never
```

In Assignment 1, we wrote a function called `table1`, which calculates the descriptive statistics for numeric variables. For this problem, you need to write a function that calculates the descriptive statistics for both numeric and categorical variables (factors).

For numeric variables, you will calculate mean (MEAN), median (MEDIAN), stardard deviation(SD), and count the number of missing values (NMiss), like you did in Assignment1.

For character variables, you will need to tabulate the count within each level of the variable and count the number of missing values.

This function takes only three argument,

`dat` : the name of the data frame, such `patient`.

`numvar` : a character vector contains one or more values. This argument contains the names of numeric variable. Set the default value to `NULL`. If you don't specify any values for this argument, which is the NULL value, then no need to calculate the statistics for the numeric variables.

`charvar` : a character vector contains one or more values. This argument contains the names of categorical variable. Set the default value to `NULL`. If you don't specify any values for this argument, which is the NULL value, then no need to calculate the statistics for the categorical variables.

The returned result is a list with length of 2. Each component of the list is either a data frame that contains the statistics or a NULL value. The first component of the list contains the descriptive statistics for the numeric variables and the second components of the list contains the counts for the character variables. The format of the returned list looks like the one below:

```
> table1 (dat=patient, numvar=c("TGL", "HDL", "LDL"), charvar=c("HRT", "MAMM"))

$numericStats
  varName      MEAN MEDIAN       SD NMiss
1     TGL 180.66667  180.0 23.03620     4
2     HDL  55.66667   62.5 19.00175     4
3     LDL 160.28571  165.0 40.06126     3

$FactorStats
  varName group count
1     HRT     N     2
2             Y     3
3         NMiss     5
4    MAMM    no     2
5           yes     4
6         NMiss     4

> table1 (dat=patient, numvar="LDL", charvar=c("HRT", "MAMM","SMOKE"))

$numericStats
  varName     MEAN MEDIAN       SD NMiss
1     LDL 160.2857    165 40.06126     3

$FactorStats
  varName group count
1     HRT     N     2
2             Y     3
3         NMiss     5
4    MAMM    no     2
5           yes     4
6         NMiss     4
7   SMOKE  ever     3
8         never     4
9         NMiss     3

> table1 (dat=patient, numvar=c("HDL", "LDL"))

$numericStats
  varName      MEAN MEDIAN       SD NMiss
1     HDL  55.66667   62.5 19.00175     4
```

```
2     LDL 160.28571  165.0 40.06126     3

$FactorStats
NULL
```

> table1 (dat=patient, charvar=c("HRT", "MAMM","SMOKE"))

```
$numericStats
NULL

$FactorStats
  varName group count
1     HRT     N     2
2             Y     3
3         NMiss     5
4    MAMM    no     2
5           yes     4
6         NMiss     4
7   SMOKE  ever     3
8         never     4
9         NMiss     3
```

Problems 4: 5 points

This problem concerns missing data imputation. You will use the same data frame, `patient`, for this problem. In the `patient` data frame, there are many missing values for both numeric and character variables.

Write a function named `impute`, which is used to replace the missing value with some meaningful values. For a numeric variable of the given data frame, you will replace the missing value(s) with the median value of the numeric variable. For a character variable, you will replace the missing value(s) with the value of the highest frequency of the character variable.

The `impute` function takes two arguments:

`dat` : the name of the data frame.

`varlist` : a character vector that contains the names of the variables you want to impute. Set the default value to NULL. If you don't provide the variable names, which is the NULL value, the function will impute all the variables of the data frame.

For example, here are some sample results that are based on the `impute` function:

> impute (dat=patient)

```
   ID GLUC TGL  HDL LDL HRT MAMM SMOKE
1   A   88 180 32.0  99   Y  yes  ever
2   B   90 150 60.0 165   Y   no never
```

```
3    C  110 180 62.5 120   N  yes never
4    D   90 200 65.0 165   Y  yes never
5    E   90 210 62.5 150   Y  yes never
6    F   88 180 32.0 210   Y  yes  ever
7    G  120 164 62.5 165   Y  yes never
8    H  110 170 70.0 188   Y  yes  ever
9    I   90 190 62.5 190   N   no never
10   J   90 180 75.0 165   Y  yes never
```

```
> impute (dat=patient, varlist=c("ID", "GLUC", "TGL", "HDL"))

   ID GLUC TGL  HDL
1    A   88 180 32.0
2    B   90 150 60.0
3    C  110 180 62.5
4    D   90 200 65.0
5    E   90 210 62.5
6    F   88 180 32.0
7    G  120 164 62.5
8    H  110 170 70.0
9    I   90 190 62.5
10   J   90 180 75.0
```

```
> impute (dat=patient, varlist=c("LDL", "HRT", "MAMM"))

   LDL HRT MAMM
1   99   Y  yes
2  165   Y   no
3  120   N  yes
4  165   Y  yes
5  150   Y  yes
6  210   Y  yes
7  165   Y  yes
8  188   Y  yes
9  190   N   no
10 165   Y  yes
```

```
> impute (dat=patient, varlist="HRT")

   HRT
1    Y
2    Y
3    N
4    Y
5    Y
6    Y
7    Y
8    Y
9    N
10   Y
```