Control structures

Peilin Chen

11/3/2017

If statement

if statement

```
if (condition1){
do1
}
```

if..else statement

```
if (condition1){
do1
}
else{
do2
}
```

nested if

```
if(condition1){
do1
}
else if(condition2){
do2
}
else if(condition3){
do3
}
else
do4
```

```
x=5
if (x%%2==0){
  print ('x is even')
}else {print ('x is odd')}
```

```
## [1] "x is odd"
```

for loop

```
for (i in 1:10){
  j=i^2
  print(j)
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
## [1] 64
## [1] 81
## [1] 100
```

while loop

```
i=1  #inital value
while (i<=10){
  print(i^2)
  i=i+1  #update i to have control of the loop
  }
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
## [1] 64
## [1] 81
## [1] 100
```

examples in a dataset

dataset GDP

```
GDP_table=read.csv('/Users/Penny/Desktop/dataset/MGSA/GDP.csv',header = TRUE)
GDP_table
```

```
##    Year Quarter     GDP
## 1  2004       1 11405.5
## 2  2004       2 11610.3
## 3  2004       3 11779.4
## 4  2004       4 11948.5
## 5  2005       1 12155.4
## 6  2005       2 12297.5
## 7  2005       3 12538.2
## 8  2005       4 12696.4
## 9  2006       1 12959.6
## 10 2006       2 13134.1
## 11 2006       3 13249.6
## 12 2006       4 13370.1
## 13 2007       1 13510.9
## 14 2007       2 13737.5
## 15 2007       3 13950.6
## 16 2007       4 14031.2
## 17 2008       1 14150.8
## 18 2008       2 14294.5
## 19 2008       3 14412.8
## 20 2008       4 14200.3
```

Data slicing get specific rows and columns

```
GDP_table[1,]  #by location index: first row, must be integer
```

```
##   Year Quarter     GDP
## 1 2004       1 11405.5
```

```
GDP_table[,1]  #by location index: first column, must be integer
```

```
##  [1] 2004 2004 2004 2004 2005 2005 2005 2005 2006 2006 2006 2006 2007 2007
## [15] 2007 2007 2008 2008 2008 2008
```

```
GDP_table[,'GDP'] #by column name
```

```
##  [1] 11405.5 11610.3 11779.4 11948.5 12155.4 12297.5 12538.2 12696.4
##  [9] 12959.6 13134.1 13249.6 13370.1 13510.9 13737.5 13950.6 14031.2
## [17] 14150.8 14294.5 14412.8 14200.3
```

```
GDP_table$GDP #by column name
```

```
##  [1] 11405.5 11610.3 11779.4 11948.5 12155.4 12297.5 12538.2 12696.4
##  [9] 12959.6 13134.1 13249.6 13370.1 13510.9 13737.5 13950.6 14031.2
## [17] 14150.8 14294.5 14412.8 14200.3
```

```
GDP_table[GDP_table$Year==2004,]
```

```
##   Year Quarter     GDP
## 1 2004       1 11405.5
## 2 2004       2 11610.3
## 3 2004       3 11779.4
## 4 2004       4 11948.5
```

```
GDP_table[(GDP_table$Year==2004&GDP_table$Quarter==1),]
```

```
##   Year Quarter     GDP
## 1 2004       1 11405.5
```

if we want to get all the data for 1st quarter of the 5 years

```
GDP_table[GDP_table$Quarter==1,]
```

```
##    Year Quarter     GDP
## 1  2004       1 11405.5
## 5  2005       1 12155.4
## 9  2006       1 12959.6
## 13 2007       1 13510.9
## 17 2008       1 14150.8
```

See the that has increment quarterly using for loop:

```
for (i in 1:20){
  if (i!=1)
    {GDP_table$increment[i]=GDP_table$GDP[i]-GDP_table$GDP[i-1]}
}
GDP_table
```

```
##    Year Quarter     GDP increment
## 1  2004       1 11405.5        NA
## 2  2004       2 11610.3     204.8
## 3  2004       3 11779.4     169.1
## 4  2004       4 11948.5     169.1
## 5  2005       1 12155.4     206.9
## 6  2005       2 12297.5     142.1
## 7  2005       3 12538.2     240.7
## 8  2005       4 12696.4     158.2
## 9  2006       1 12959.6     263.2
## 10 2006       2 13134.1     174.5
## 11 2006       3 13249.6     115.5
## 12 2006       4 13370.1     120.5
## 13 2007       1 13510.9     140.8
## 14 2007       2 13737.5     226.6
## 15 2007       3 13950.6     213.1
## 16 2007       4 14031.2      80.6
## 17 2008       1 14150.8     119.6
## 18 2008       2 14294.5     143.7
## 19 2008       3 14412.8     118.3
## 20 2008       4 14200.3    -212.5
```

Or you can use while loop as well:

```
i=1
while (i<20){
  i=i+1
  GDP_table$increment2[i]=GDP_table$GDP[i]-GDP_table$GDP[i-1]
}
GDP_table
```

```
##    Year Quarter     GDP increment increment2
## 1  2004       1 11405.5        NA         NA
## 2  2004       2 11610.3     204.8      204.8
## 3  2004       3 11779.4     169.1      169.1
## 4  2004       4 11948.5     169.1      169.1
## 5  2005       1 12155.4     206.9      206.9
## 6  2005       2 12297.5     142.1      142.1
## 7  2005       3 12538.2     240.7      240.7
## 8  2005       4 12696.4     158.2      158.2
## 9  2006       1 12959.6     263.2      263.2
## 10 2006       2 13134.1     174.5      174.5
## 11 2006       3 13249.6     115.5      115.5
## 12 2006       4 13370.1     120.5      120.5
## 13 2007       1 13510.9     140.8      140.8
## 14 2007       2 13737.5     226.6      226.6
## 15 2007       3 13950.6     213.1      213.1
## 16 2007       4 14031.2      80.6       80.6
## 17 2008       1 14150.8     119.6      119.6
## 18 2008       2 14294.5     143.7      143.7
## 19 2008       3 14412.8     118.3      118.3
## 20 2008       4 14200.3    -212.5     -212.5
```

next is used to skip an iteration of a loop

```
for (i in 1:20){
  if (i%%2==0){
    next  #skip all the even numbers
  }
  print(i)
}
```

```
## [1] 1
## [1] 3
## [1] 5
## [1] 7
## [1] 9
## [1] 11
## [1] 13
## [1] 15
## [1] 17
## [1] 19
```

break is to stop a loop immediately

```
for (i in 1:20){
  print(i)
  if(i>10){
    break  #stop after first 10 iterations
  }
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
## [1] 11
```

Summary

```
1.if and else: test a condition and act on it
2.for: iteration for a fixed number of times
3.while: iteration on a condition
4.next: skip an iteration for a loop; similar to continue
5.break: break the execution of a loop
```