```
################《R语言基础课程》####################

###1.R语言介绍####

###1.1R语言历史####
#R是S语言的一种实现。S语言是由 AT&T贝尔实验室开发的一种用来进行数据探索、
#统计分析、作图的解释型语言。最初S语言的实现版本主要是S-PLUS。
#S-PLUS是一个商业 软件，它基于S语言，并由MathSoft公司的统计科学部进一步完善。
#后来Auckland大学的Robert Gentleman 和 Ross Ihaka 及其他志愿人员开发了一个R系统。
#R的使用与S-PLUS有很多类似之处，两个软件有一定的兼容性。

###1.2R的特点####
#1. 有效的数据处理和保存机制。
#2. 拥有一整套数组和矩阵的操作运算符。
#3. 一系列连贯而又完整的数据分析中间工具。
#4. 图形统计可以对数据直接进行分析和显示，可用于多种图形设备。
#5. 一种相当完善、简洁和高效的程序设计语言。它包括条件语句、循环语句、用户自定义的递归函数以及输入输出接口。
#6. R语言是彻底面向对象的统计编程语言。
#7. R语言和其它编程语言、数据库之间有很好的接口。
#8. R语言是自由软件，可以放心大胆地使用，但其功能却不比任何其它同类软件差。
#9. R语言具有丰富的网上资源


###2.Rstudio和R的基本操作####

#2.1查看R语言自带的数据集####
data()

#直接输入数据集的名称，查看这些数据
CO2

#2.2快捷键####
#Ctrl+Enter：运行光标所在行的代码，也可以用来运行鼠标选中区域的代码
#Ctrl+L：清除控制台中的代码
#Ctrl+shift+S：运行代码集中的所有代码
#Ctrl+1：跳转到代码编辑环境
#Ctrl+2：跳转到控制台
#Ctrl+D：删除代码集中光标所在行的代码

#2.3查看数据集的前6行####
head(CO2)

#查看数据集最后的6行
tail(CO2)

#2.4安装包(可以通过命令或者通过图形界面来安装)####
install.packages("ggplot2")

#加载包
library(ggplot2)

#2.5创建向量和矩阵,以及对它们的基本操作####
x1<-c(1,2,3,4,5,6)
x2<-c(2,4,6,8,10,12)

length(x1)

mode(x1)

class(x1)

rbind(x1,x2)

cbind(x1,x2)
```

```
Dmat<-cbind(x1,x2)
class(Dmat)                 #使用c函数结合的变量组成的是矩阵

Dmat2<-rbind(x1,x2)
class(Dmat2)

Dframe<-as.data.frame(cbind(x1,x2))
class(Dframe)

Dmat;Dframe          #矩阵和数据框格式是不一样的，矩阵是由行列组成的，数据框是由记录和变量组成的


a<-c(1:100)

length(a)

cbind(a)

a

#2.6常用的函数####
b<-c(1:20)

mean(b)

sum(b)

max(b)

min(b)

sd(b)

#产生向量
1:5

1:5*2

1:5*2+1

#2.8查看内存中已有的对象####
ls()

#删除当前内存中的所有对象
rm(list = ls())

ls()

#2.9访问向量中的元素####
a<-1:5*2-1

a

a[3]

a[-3]

a[2:4]

a[-(2:4)]

a[1,2,3]    #不能这样访问向量中的元素

a[c(1,2,3)]

a[a<=5]
```

```r
a[a<=3 | a>=7]

a[a>=3 & a<=7]

a[a[2]]

#2.10 seq()函数和rep()函数，用来产生等差数列的函数，调用示例函数####
example("seq")

seq(0, 1, length.out = 11)

seq(stats::rnorm(20))

seq(1, 9, by = 2)

seq(1, 9, by = pi)

seq(1, 6, by = 3)

seq(1.575, 5.125, by = 0.05)

seq(17)

example("rep")

rep(c("a","b","c"),each=4)

rep(c("a","b","c"),4)

#2.11 产生字母序列####
letters[1:30]

#2.12 which()函数####
a<-rnorm(10)
a

which.max(a)

a[which.max(a)]

which(a==a[which.max(a)])

which(a>0)

a[which(a>0)]

#2.13 排序函数####
a<-1:10
a

rev(a)        #反转顺序

a<-c(3,1,5,6,9,2,7,4,6,5)

sort(a)

rev(sort(a))

#2.14 生成矩阵####
a<-c(1:36)

a1<-matrix(a,nrow = 9,ncol = 4);a1

a2<-matrix(a,nrow = 4,ncol = 9);a2

a3<-matrix(a,nrow = 9,ncol = 4,byrow = TRUE);a3
```

```
dim(a1)           #dim函数用来查看矩阵的维度

dim(a2)[1]        #行

dim(a3)[2]        #列
#矩阵运算
t(a1)

a1+a2

a1+a3


#2.15 改变Rstudio的工作界面（通过tools里面的layout设置完成）####

#2.16 设定工作目录，方便载入数据.将数据存放在设定的目录中，载入数据时就不用指定路径了####
setwd("D:/Rstudy/data")
setwd("D:\\Rstudy\\data")

getwd()   #用于知道当前工作目录

#2.17 help()函数 ####
help("seq")
?seq
help(package="ggplot2")  #获取对整个包的帮助文件

#2.18 example()函数 ####
example("seq")
example("plot")
example("hist")

#搜索自己需要的函数
help.search("multivariate normal")

#2.19 数据框。dataframe####
##数据框就是一个数据表格，一行表示一个记录，一列表示一个变量

x1<-c(1,2,3,4,5,6,7,8,9)
x2<-c(2,4,6,8,10,12,14,16,18)
x<-data.frame(x1,x2);x

x=data.frame('重量'=x1,'运费'=x2);x

x$"费率" <- x$运费/x$重量 #数据框可以利用已有的变量产生新的变量并存储于当前数据框中

names(x)     #查看数据框中的变量

str(x)       #查看数据框中数据的定义

#我们在载入数据后，要使用上面的两个函数来检查数据载入是否成功，并检查数据的定义



###2.20 列表 list（）####
x1<-1                    #单个数字，其实是一个只有一个元素的一维向量

x2<-c(1,2,3,4)           #有四个元素的一维向量

x3<-c("a","bc","d")      #字符向量

x4<-matrix(1:36,nrow = 9,ncol = 4)   #矩阵，vector

x5<-data.frame(a=c(1,2,3,4),b=c(2,3,4,5),c=c(3,4,5,6))

list_01<-list(x1=x1,x2=x2,x3=x3,x4=x4,x5=x5)        #产生一个数据框

list_01$x3
```

```
###3.数据导入与导出####

#3.1 先把R内置的CO2数据集导出，然后再练习导入操作####
write.table(CO2,file = "二氧化碳.txt")
write.csv(CO2,file = "二氧化碳.csv")

#3.2 导入逗号分割的文本文件####
read.table(file = "二氧化碳.txt",header = TRUE)

#最好给指定一个名称#
carbon<-read.table(file = "二氧化碳.txt",header = TRUE)

#3.3 导入csv格式的数据####
carbon_csv<-read.csv(file = "二氧化碳.csv",header = TRUE)

#3.4 通过剪贴板读入数据并写入文件####
setwd("d:/Rstudy/data")
clipboard<-read.table("clipboard",header = TRUE)
clipboard

write.table(clipboard,file = "clipboard.txt")

#3.5 导入空格分割的文本文件数据####
setwd("d:/Rstudy/data")
lowtem<-read.table(file = "lowtemperature.prn",header = TRUE)
names(lowtem);str(lowtem)

lowtem$fyear<-factor(lowtem$year,c(2013,2014,2015),labels = c("2013年","2014年","2015年"))
#此处设计重新编码变量的操作

head(lowtem)

#3.5.1 读取数据框中的变量####

#1)很多函数中有data参数，可以指定数据框，然后在函数内部直接访问数据框中的变量
boxplot(lowtem~year,data = lowtem)
#2）使用美元符号$
boxplot(lowtem$lowtem~lowtem$year)
#3)使用attach函数，不推荐使用，很容易出错，特别是在同一个R代码集中写很多不同程序的时候
attach(lowtem)
boxplot(lowtem~year)
detach(lowtem)
###3.6 数据子集操作####

#1)先明确分类变量有几个类别
unique(lowtem$year)
#2)选择数据子集
lowtem_15<-lowtem[lowtem$year==2015,]
plot.ts(lowtem_15$lowtem)
#3)多个条件用逻辑连接符号进行连接
lowtem_1501<-lowtem[lowtem$year==2015&lowtem$day=="1/1",]
lowtem_1501
#4）排序操作

#按照日最低气温进行排序，这个排序可能没有实际意义，这里只是演示操作的方便

orderlowtem<-lowtem[order(lowtem$lowtem),]   #对行的操作
View(orderlowtem)

#5）合并数据子集

#加入三年的气温数据是分开的，我们要将他们合并到一个数据集中
#（1）先读入数据
setwd("D:/Rstudy/data")
hightem<-read.table("clipboard",header = TRUE)          #通过剪贴板读入数据

write.table(hightem,file = "hightem.txt",quote = FALSE)  #将这个数据保存以便于下次使用
```

```
lowtem<-read.table(file = "lowtemperature.prn",header = TRUE)
#（2）然后进行合并

colnames(hightem)<-c("day","year","hightem")   #由于变量名称大小写不一致，对最高温度的数据集变量名称进行重命名

tem<-merge(hightem,lowtem,by=c("year","day"))

tem$day<-as.Date(tem$day)

#(3)合并后有一个问题，时间序列被打乱了
#暂时解决办法是把文件写入csv文件，用excel重新编辑日期，并排序以后重新导入

tem_new<-read.csv(file = "tem.csv",header = TRUE)
head(tem_new)

#3.7 读入Excel格式的文件####
install.packages("RODBC")   #注意有些包只在特定版本中能够运行，更新问题
library(RODBC)

xls<-odbcConnectExcel("wechat.xls")


###4.常用的函数####

###4.1 tapply函数(对某数值变量依据另外一个分类变量求某值)####
setwd("D:/Rstudy/data")

tem_new<-read.csv(file = "tem.csv",header = TRUE)  #读入数据文件
head(tem_new)
str(tem_new)

tapply(tem_new$lowtem, tem_new$year, mean)   #计算最低气温均值

tapply(tem_new$hightem, tem_new$year, mean)  #计算最高气温均值

tapply(tem_new$lowtem, tem_new$year, sd)     #计算最低气温方差

###4.2 sapply()函数和lapply()函数（对多个变量求某函数的结果）####

sapply(tem_new[,4:5],mean)   #输出为向量

lapply(tem_new[,4:5],mean)   #输出为列表

###4.3 summary()函数 用来输出汇总信息####
a<-summary(tem_new)

b<-summary(tem_new$year)

c<-summary(c(1,2,3,4,5,6,7,8,9))

d<-lm(tem_new$hightem~tem_new$lowtem)

e<-summary(d)

class(a);class(b);class(c);class(d);class(e)  #summary函数的输出非常丰富,是一种列表

e$coefficients

###4.4 table函数 （单变量的频数统计，多分类变量的交叉分析）

example("table")  #直接调用example函数来查看它的用法

#4.4.1简单的频数统计####
rpois(100, 5)
table(rpois(100, 5))
```

```
#4.4.2 查看表的结构(可用于对数据框中两个分类变量进行交叉分析)####
View(warpbreaks)
with(warpbreaks, table(wool, tension))

###5.绘图工具####
#5.1 基本绘图函数plot（）####
View(cars)
help(cars)     #行驶速度与刹车距离

head(cars)

plot(cars)

plot(cars$dist~cars$speed)   #指定y轴表示刹车距离

plot(cars$speed~cars$dist)   #指定x轴表示刹车距离

#5.2 为了方便将两幅图画在一起进行对比####


windows()                    #用单独的窗口显示图形以获得更好的显示效果，注意windows命令在前

par(mfrow=c(2,2))            #c(2,2)表示生成一个2*2的矩阵

plot(cars$dist~cars$speed)

plot(cars$speed~cars$dist)

#5.3 函数的参数####

plot(cars$dist~cars$speed, # y~x
     main="刹车距离与车速之间的关系",      # 画标题
     xlab="Speed (miles per hour)",        #X坐标轴标题
     ylab="Distance travelled (miles)",    #Y坐标轴标题
     xlim=c(0,30),                         #设置X轴范围为从0到30
     ylim=c(0,140),                        #设置Y轴范围为从0到140
     xaxs="i",                             #设置X轴风格internal
     yaxs="i",                             #设置Y轴风格internal
     col="red",                            #设置“散点”的颜色为红色
     pch=19)                               #设置散点的形状为实心圆点

#5.4绘制线图####
#使用微信公众号用户数据
setwd("d:/Rstudy/data")

num<-read.csv(file = "wechat.csv")
head(num)

windows()
plot(num$NetGrowth~as.Date(num$date),
     type="l",
     mian="每日净增长变化",
     xlab="日期",
     ylab ="净增长人数",col="red" )

#5.5 低水平绘图函数lines()####
lines(num$NetGrowth~as.Date(num$date),col="red")

example("lines")

plot(cars, main = "Stopping Distance versus Speed")

lines(stats::lowess(cars))   #此段代码为散点图加上光滑的曲线


#5.6 柱形图####
#使用数据集BOD
```

```
View(BOD)

#使用基础绘图包
barplot(BOD$demand,names.arg = BOD$Time)
#先输出频数表，然后在用条形图绘制
table(mtcars$cyl)
barplot(table(mtcars$cyl))

#改变图形的方向
windows()
par(mfrow=c(1,2))
barplot(BOD$demand,names.arg = BOD$Time)
barplot(BOD$demand,names.arg = BOD$Time,horiz = TRUE)


###5.7.绘制直方图####
windows()
par(mfrow=c(1,2))
hist(mtcars$mpg)

hist(mtcars$mpg,breaks=10)   #控制分区数目

###5.8绘制箱线图####
#使用数据集ToothGrowth
View(ToothGrowth)
#使用plot函数时，当x轴为分类变量，y轴为数值型变量时，默认输出箱线图
plot(ToothGrowth$supp,ToothGrowth$len)

###5.9 绘图设备(输出图形的格式)####

?device    #查看可以使用的图形设备

setwd("d:/Rstudy/image")
pdf("boxplot.pdf")          #在图形设备中输出图形名称
plot(ToothGrowth$supp,ToothGrowth$len)
dev.off()                   #关闭当前图形设备，相当于保存该图形

png("boxplot.png")
plot(ToothGrowth$supp,ToothGrowth$len)
dev.off()

pdf("综合图.pdf")   #在你希望将所有图形输出到一个文件中时，这样做,不支持PNG格式
hist(mtcars$mpg)
hist(mtcars$mpg,breaks=10)
plot(ToothGrowth$supp,ToothGrowth$len)

dev.off()

#图形设备设置

dev.cur()  #查看当前正在使用的图形设备

dev.set(3) #把当前正在使用的图形设别设置为其它设备

#5.10 其他常用图形####
example("pie")    #查看饼图的使用方法和示例

example("plot.ts")#时间序列图示例

#5.11 绘图参数####

setwd("d:/Rstudy/data")
wec<-read.csv(file = "wechat.csv",header = TRUE)

head(wec)
windows()
plot.ts(wec$NetGrowth,col="red")
```

```
points(wec$NetGrowth,bg="skyblue",color="red",pch=21)
abline(h=mean(wec$NetGrowth),col="blue")
title("公众号日净增关注人数变化")
box()

#5.12 生成时间序列的函数####

ts(1:10, frequency = 4, start = c(1959, 2)) #生成时间序列，季度，起始

print( ts(1:10, frequency = 7, start = c(12, 2)), calendar = TRUE)
                                              #星期 日历
gnp <- ts(cumsum(1 + round(rnorm(100), 2)),
          start = c(1954, 7), frequency = 12)

z <- ts(matrix(rnorm(300), 100, 3),
        start = c(1961, 1), frequency = 12)

class(z);head(z);plot(z)


###6.R语言与统计####

###7.循环与自定义函数####
#7.1 for循环####

example("for")

for(i in 1:5)  print(1:i)    #循环遍历每个数，每取出一个数，就打印一个向量

for(n in c(2,5,10,20,50)) {  #循环遍历向量中的没一个数
  x <- stats::rnorm(n)       #生成含有n个数的正态分布，n是上面循环遍历的那个数
  cat(n,":", sum(x^2),"\n")  #对应每一个数n，计算生成的正态分布数组的平方和
}

f <- factor(sample(letters[1:5], 10, replace = TRUE));f  #抽样，生成字母序列

for(i in unique(f)) print(i)#依次取出序列中的每一个独有的值

#7.2 while语句####
a[1]<-1;i<-1

while(a[i]<100){
  i=i+1;
  a[i]<-a[i-1]+3
}

#7.3 自定义函数####

myfun_cv<-function(x){ #函数在R中也是一个对象
  cv<-sd(x)/mean(x)    #函数的语句体用来计算变异系数
  return(cv)           #函数执行完毕后返回cv值，该值就是变异系数
}

#测试该函数
a<-c(1,2,5,8,9,6)      #生成一个向量
myfun_cv(a)            #调用自定义的函数来计算变异系数

#7.4 通过循环和自定义函数来验证中心极限定理####
myfun<-function(a){
  x<-1:100             #先生成一个1到100的序列，后面可以更改这些值，相当于覆盖掉原来的值
  x<-data.frame(x)
  a<-data.frame(a)

  for(i in 1:100){        #设置循环，循环抽取100个样本，并将计算出来的均值赋值给数据框中的x变量
    c<-a[sample(nrow(a),1000),]
    m=mean(c)
    x$x[i]<-m
```

```
  }

  windows(1280,720);par(mfrow=c(1,2))
  plot(density(a$a),main = "这是原来的分布")
  plot(density(x$x),main = "这是抽取的样本的均数的分布")
}

###7.4 .1正态分布####
a<-rnorm(10000,0,1)
myfun(a)
###7.4.2指数分布####
b<-rexp(100000,1)
myfun(b)
###7.4.3t分布####
c<-rt(1000,3)
myfun(c)
###7.4.4F分布####
d<-rchisq(100000,1)
myfun(d)

###8.数据处理####

###8.1把数据框变长####
```