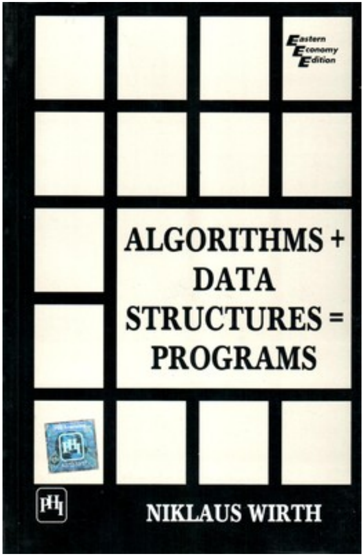
# R programming: condition and loop

Qiang Shen

Jan.9,2018

## Programs


Eastern Economy Edition
ALGORITHMS + DATA STRUCTURES = PROGRAMS
NIKLAUS WIRTH

## Programs

data structures + algorithm = Programs

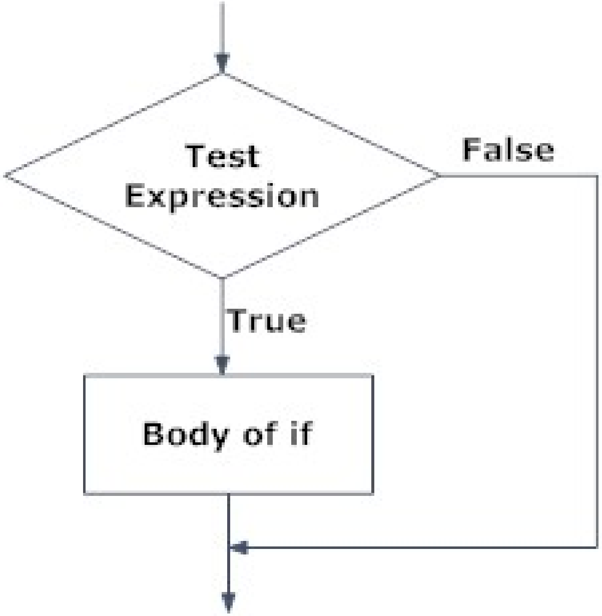
algorithm:

- condition
- loop
- recursion
- function
- ...

## Condition: syntax of if statement

```
if (test_expression) {
   statement
}
```

## Condition: if statement



## Condition: if statement

To judge whether a value is odd or not?

## Condition: if statement

```
num <- 5    ##change num to 6
if (num %% 2 ==1) {   ###remove ==!0
  print("num is even") ##change print to cat
}
if (num %% 2 ==1) print("num is even") ##omit the {}
#double Percent sign
## %%  reminder x mod y ## %/% mode
7 %% 2
7 %/%2
x<-c(1.2,2,4,5,8)
x%%1==0
```

## Condition: syntax of if...else statement

```r
if (test_expression) {
   statement1
} else {
   statement2
}
```

## Condition: if...else statement

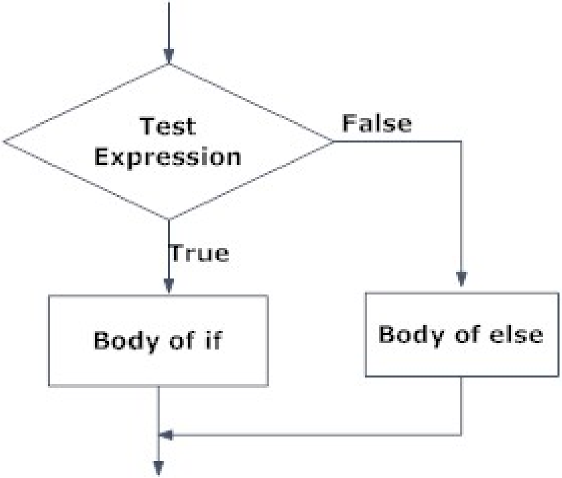


Fig: Operation of if...else statement

## Condition: if...else statement

```r
num <- 5   ###change num to 6
if (num %% 2 != 0) {
  cat(num,"is odd")
  }else{    ###change the position of {}
  cat(num,"is even")
  }
```

## Condition: if...else statement continued

```r
num <- 6   ###change num to 6
if(num %% 2 != 0) cat(num, "is odd")else cat(num,"is even")
y<-if (num %% 2 != 0) 'odd' else 'even'
y
```

## Nested if...else statement

```
if ( test_expression1) {
   statement1
} else if ( test_expression2) {
   statement2
} else if ( test_expression3) {
   statement3
} else
   statement4
```

## Nested if...else statement

```r
num <- 3
if (num < 0) {
   print(paste0(num," is a Negative number"))
} else if (num > 0) {
   print(paste0(num, " is a positive number"))
} else
   print(paste0(num, " is Zero"))
```

## Condition: syntax of ifelse() function

```r
ifelse(test_condition, true_value, false_value)
```

## Example of ifelse() function

```r
a = c(5,7,2,9)
ifelse(a %% 2 == 0,"even","odd")
```

```
[1] "odd"  "odd"  "even" "odd"
```

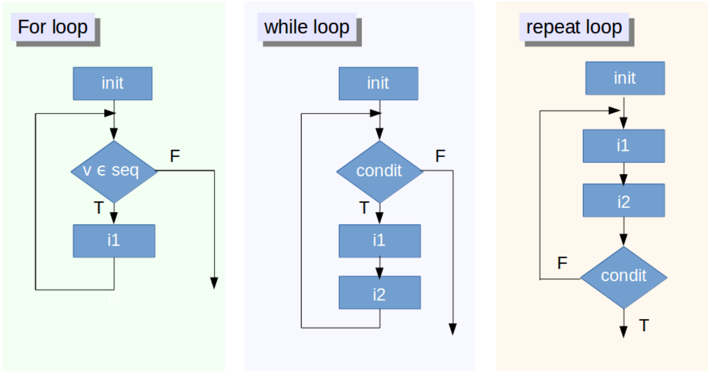
## Condition: switch()

```r
centre = function(x, type) {
  switch(type,
         mean = mean(x),
         median = median(x),
         trimmed = mean(x, trim = .1))
}
a = rnorm(10)
centre(x=a, type="mean")
centre(x=a, type="median")
centre(x=a, type="trimmed")
```

## Loop

-Replicate execution

- for
- repeat
- while



## Loop: for

```r
for (counter in vector) {commands}
```

## loop: for

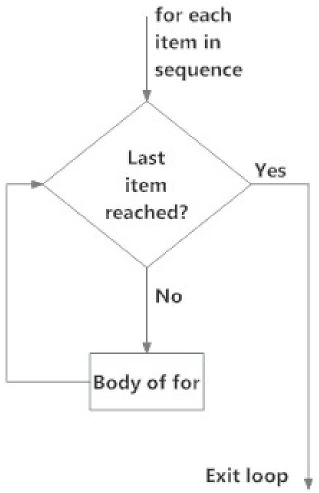


Fig: operation of for loop

## loop: for example 1 for “for”

```
1    January
2    February
3    March
4    April
5    May
6    June
7    July
8    August
9    September
10   October
11   November
12   December
```

## loop: for example 1 for “for”

```r
for (i in 1:12){
  cat(i,month.name[i],sep="\t")
  cat('\n')
#   print(paste(i,month.name[i],sep=":"))
}
```

## GUASS

-Anecdotes: 1+2+3+....+100

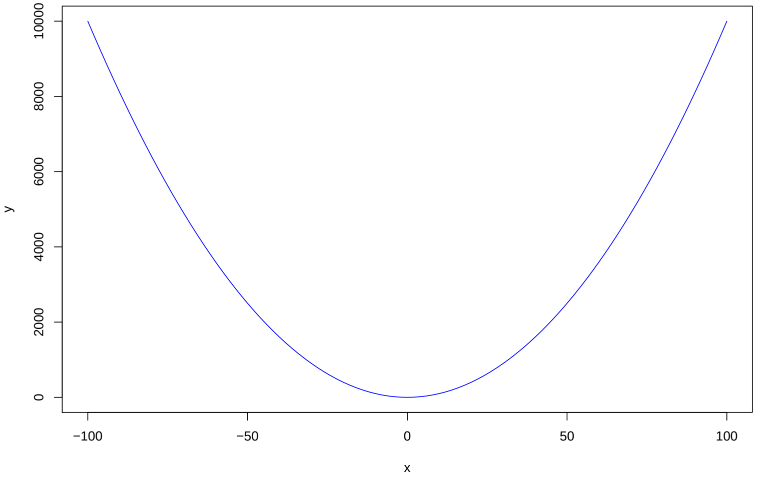
## GUASS

```r
x<-0    ###note the importance of assign an initial value t
for (i in 1:100){  ## add if (i%%2!=0){}
 if (i%%2!=0)
  x<-x+i
}
x
```

```
[1] 2500
```

```r
# sum(seq(1,100,by=2))
```

## Loop: example 2 for “for”



## Loop: example 2 for “for”

```
value0=100 #change 100 to 100000
x = seq(-value0,value0,by=2)
y=NULL
##possible mistake 1   # value.cubed=NULL
for (i in 1:length(x)){
y[i]<-x[i]^2
##possible mistake 2  # value.cubed[i]<-value[i]^3
}
plot(x,y,type='l',col='blue')
##curvature for y=x^2
# lines(value,value.cubed,col='blue')
# summary(value.squared)
```

## Loop: example 2 for “for”

```
start=Sys.time()
value = seq(1,100000,by=2)   ###the disadvantage of R.
value.squared=NULL ###value.squared<-rep(NA,length(value))
length(value)
for (i in 1:length(value)){
value.squared[i]<-value[i]^2
}
# value.squared<-!is.na(value.squared)
summary(value.squared)
end<-Sys.time()
end-start
```

## sapply

```
rm(list=ls())
value.squared<-sapply(seq(1,1000,by=2),function(x) x^2)
```

## Data frame example

- if
- for

```r
load('data/mydata.rdata')
head(mydata)
```

```
  ID age Sex Weight Height Married  Race
1  1  26   1    132     60       0 White
2  2  65   0    122     63       2 White
3  3  15   1    184     67       2 White
4  4   7   1    145     59       0 Black
5  5  80   0    110     64       0 Black
6  6  43   1     NA     NA       0 Black
```

```r
dim(mydata)
```

```
[1] 10  7
```

## data frame example: no vectorization

```r
load('data/mydata.rdata')
mydata$agegroup<-1
if (mydata$age>=20)
  mydata$agegroup<-3
```

## method 1: using “for” loop.

```
load('data/mydata.rdata')
# head(mydata)
mydata$agegroup<-0
for (i in 1:10){
  if (mydata$age[i]<=10) {
      mydata$agegroup[i]<-1
  }else if (mydata$age[i]>10 & mydata$age[i]<20){
      mydata$agegroup[i]<-2
  }else if (mydata$age[i]>=20) {
    mydata$agegroup[i]<-3
}
}
head(mydata)
```

## method 2: ifelse for data frame

```r
load('data/mydata.rdata')
head(mydata)
mydata$agegroup<-ifelse(mydata$age>10,1,0)
head(mydata)
```

## nested ifelse.

```r
load('data/mydata.rdata')
mydata$agegroup<-ifelse(mydata$age<=10,1,
ifelse(mydata$age>10&mydata$age<20,2,3))
head(mydata)
```

```
  ID age Sex Weight Height Married  Race agegroup
1  1  26   1    132     60       0 White        3
2  2  65   0    122     63       2 White        3
3  3  15   1    184     67       2 White        2
4  4   7   1    145     59       0 Black        1
5  5  80   0    110     64       0 Black        3
6  6  43   1     NA     NA       0 Black        3
```

## method 3: [] to substibute ifelse

```
load('data/mydata.rdata')
     mydata$agegroup[mydata$age<=10]<-1
     mydata$agegroup[mydata$age>10&mydata$age<20]<-2
     mydata$agegroup[mydata$age>20]<-3
head(mydata)
```

```
  ID age Sex Weight Height Married  Race agegroup
1  1  26   1    132     60       0 White        3
2  2  65   0    122     63       2 White        3
3  3  15   1    184     67       2 White        2
4  4   7   1    145     59       0 Black        1
5  5  80   0    110     64       0 Black        3
6  6  43   1     NA     NA       0 Black        3
```

## Loop: while statement

```
while (condition) {
  statements
}
```

## Loop: while statement

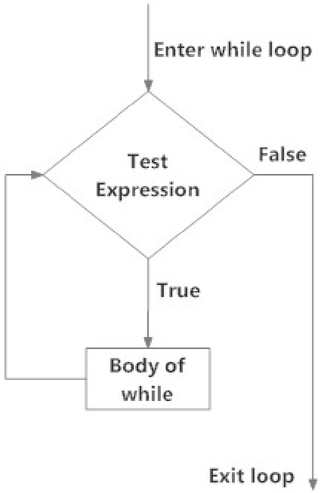


Fig: operation of while loop

## Loop: while statement

caution of infinite loop.

```r
i = 0
while (i <10) {
  i = i + 1
  print(i)
}
```

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

## Loop: while statement: example 1

```r
data<-data.frame(ID=paste0('a',1:10),
                 num=c(0.1,0.2,0.3,0.4,0.3,0.5,0.6,0.7,0.8,0.5))
for (i in 1:9) {
  cat(paste0(i,":",data[i,'num']<data[i+1,'num']))
  cat("\n")
}
```

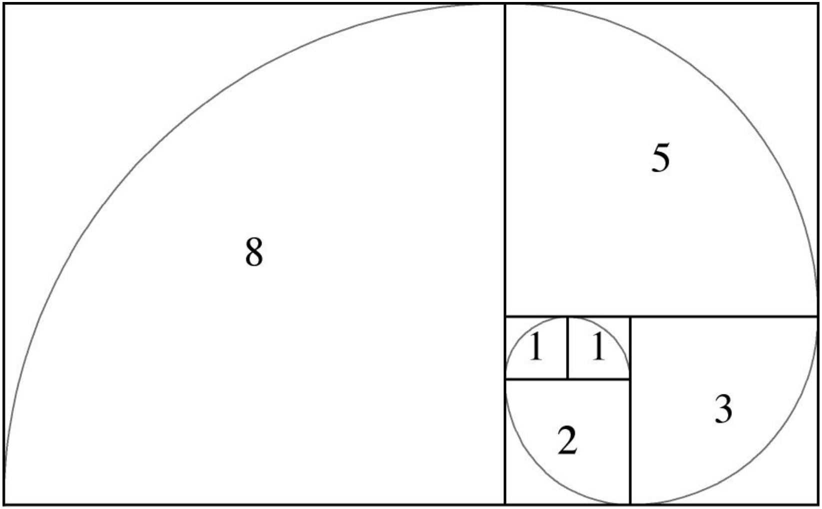
```
1:TRUE
2:TRUE
3:TRUE
4:FALSE
5:TRUE
6:TRUE
7:TRUE
8:TRUE
9:FALSE
```

## Loop: while statement: example 1

```r
i=0
while (i<9) {
  i=i+1
  cat(paste0(i,":",data[i,'num']<data[i+1,'num']))
  cat("\n")
  # print(i)
}
```

```
1:TRUE
2:TRUE
3:TRUE
4:FALSE
5:TRUE
6:TRUE
7:TRUE
8:TRUE
9:FALSE
```

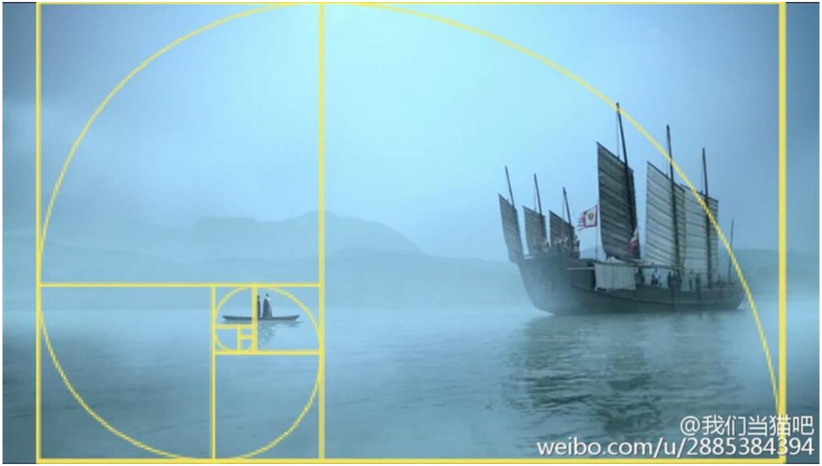
## Fibonacci seqence



## Fibonacci seqence

## Fibonacci seqence

Nirvana in Fire(琅琊榜)



## Loop example with “for”: Fibonacci

```
myseq<-NULL
myseq[1] = 0
myseq[2] = 1
for (i in 3:12) {
  myseq[i] = myseq[i-2] + myseq[i-1]
}
myseq
```

```
 [1]  0  1  1  2  3  5  8 13 21 34 55 89
```

# Loop example with “while”: Fibonacci

```
myseq[1] = 0
myseq[2] = 1
i = 2
currentVal = 1
while (currentVal < 500) {
  myseq[i+1] = currentVal
  currentVal = myseq[i] + myseq[i+1]
  i = i+1
}

myseq
```

```
 [1]   0   1   1   2   3   5   8  13  21  34  55  89 144 23
```

## Loop next

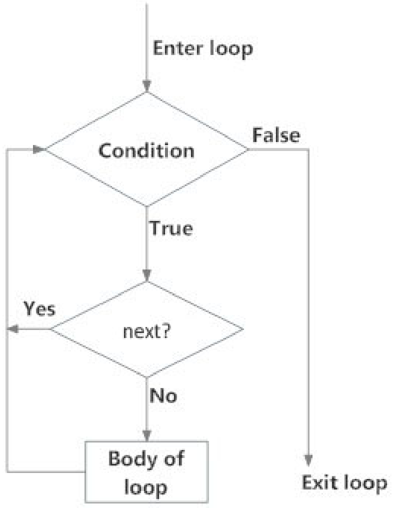


Fig: flowchart of next

## Loop:next

- `next` jump out of the current loop
- example: print value 1,2,3,4,6,7,8,9,10

```
for (i in seq(1,10)) {
  if (i == 5) {
    next
  }
  print(i)
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 6
[1] 7
[1] 8
[1] 9
```

## loop break

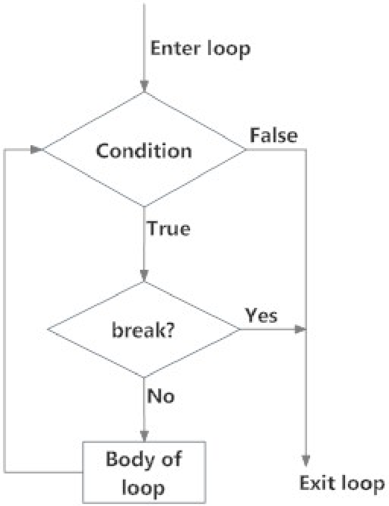


Fig: flowchart of break

## Loop: break

break exit the loop:

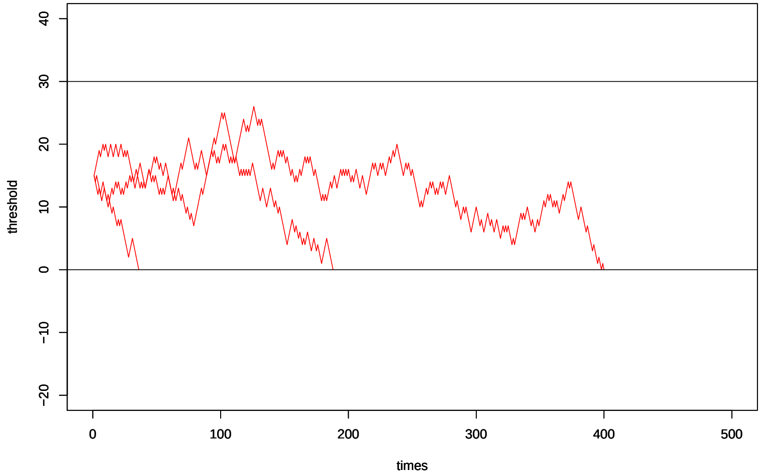
```
val = 0
i = 0
while(TRUE) {
  i = i + 1
  val = val + rnorm(1)
  if (abs(val) > 3) {
    break
  }
}
val
```

```
[1] -3.491233
```

```
i
```

```
[1] 27
```

## Example: random walk



## random walk

```
###random walk
for (s in 1:3){
z<-15;i=1;data<-NULL
while(z>=0&&z<=30){
                    data=rbind(data,cbind(i,z))
                    coin <- rbinom(1, 1, 0.5)
                    if(coin == 1) { ## random walk
                      z<-z+1
                    }else{
                      z<-z-1
                    }
                    i=i+1
}
data<-as.data.frame(data)
par(new=TRUE)
plot(data$i,data$z,type="l",col='red',xlim=c(0,500),ylim=c
}
```

## summary

condition

- if
- if...else
- nested if
- ifelse
- switch

loop

- for
- while
- next,break

## summary

- speed up the loop
- vectorization of if for data frame