In [ ]:
```
#-------------  R LANGAGGE NOTES REVIEW --------------------
#R is popular programming languagee used for statical proggramming and graphhi
```

In [22]:
```
# R syntax
print("hello world")
```

```
[1] "hello world"
```

In [23]:
```
# R variables  ( use this <-)
name <- "yogesh"
age <- 20
#print simply
name
age
```

'yogesh'

20

In [24]:
```
#Concanate Elelments  ( using paste() function to join two or more element)
text <- "Yogesh"
paste("My name is",text)
```

'My name is Yogesh'

In [25]:
```r
#Another Example
var1 <-"R is"
var2 <- "awesome language"
paste(var1,var2)
```

'R is awesome language'

In [26]:
```r
# In R ('+') use only mathmatical operator not join two string
num <- 10
s <- "ten"
num+s
```

```
Error in num + s: non-numeric argument to binary operator
Traceback:
```

In [27]:
```r
#=======R DATATYPE=============
#1. numeric -> 10.5 , 55, 292.000
#2. Integer -> 1L , 55L , 100L ( L is declared is integer
#3. complex -> 4+5i , 3+19i
#4. Character -> "k" , "False", "11.5"
#5. Logical -> True or False
```

In [28]:
```
# Check Data type  -> use class()
x <- 10.5
class(x)
y <- 100L
class(y)
z <- 6+3i
class(z)
```

'numeric'

'integer'

'complex'

In [29]:
```
#-----Type Conversion-----
x <- 10L
y <- 20
class(x)
class(y)
# Convert int to numeric
a <- as.numeric(x)
class(a)
b <- as.integer(y)
class(b)
```

'integer'

'numeric'

'numeric'

'integer'

In [30]:
```r
# R Maths----- ( Built-In Math function)
print("max and min")
max(5,10,-10)
min(5,10,-10)

print("Square")
sqrt(16)

print("absolute")
abs(-292.62)

print("Ceiling and floor function")
# Celling -> Round the numbber upward to nearest
#Floor -> Rounn the number downward to nearest
ceiling(1.4)
floor(1.4)
```

```
[1] "max and min"

10

-10

[1] "Square"

4

[1] "absolute"

292.62

[1] "Ceiling and floor function"
```

2

1

In [31]:
```r
# R String ------
str1 <- "Good, R ,programing ,laguage"
str2 <- "Good, R ,programing ,laguage"

print(str1)
cat(str2)


# To find the lenght of string
nchar(str2)

# To check character present in string
grepl("Good",str1)
grepl("NO",str1)


# iF Define string inside string
# str1 <- "Good, R "programing" laguage"   # ERROR
# print(str1)
str2 <- "Good, R \"programing\" laguage"  # \""\ = escape character
print(str2)
cat(str2)
```

```
[1] "Good, R ,programing ,laguage"
Good, R ,programing ,laguage

28

TRUE

FALSE

[1] "Good, R \"programing\" laguage"
```

```
Good, R "programing" laguage
```

In [32]:
```r
# R Conditional Statements

# If..else
a <- 33
b <- 233
if(a<b){
    print("a is smaller than b")
}else if(a==b){
    print("a is equall to b")
}else{
    print("a is larger than b")
}
```

```
[1] "a is smaller than b"
```

In [33]:
```r
# Logical conditional statement
a <- 200
b <- 33
c <- 500
if(a>b & c>a){
    print("Both are true")
}
if(a>b | c>a){
    print("At least one condition is true")
}
```

```
[1] "Both are true"
[1] "At least one condition is true"
```

In [ ]:

```r
# R While loop
i <- 1
while(i<10){
    print(i)
    i <- i+1
}


# Using Break statement
i <- 1
while(i<6){
    print(i)
     i<- i+1
    if(i==4){
        break
    }
}

# Using Next Statement  (continue)
i <- 0
while(i<7){
    i <- i+1
    if(i==2){
        next
    }
    print(i)
}
```

In [34]:
```r
# R For loop
for(x in 1:5){
    print(x)
}

# print every  items in list
fruit <- list("apple","mango","grapes")
for(i in fruit){
    print(i)
}

#Nested for loop
color <- list("red","pink","orange")
fruit <- list("apple","mango","grapes")
for(i in color){
    for(x in fruit){
        print(paste(i,x))
    }
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] "apple"
[1] "mango"
[1] "grapes"
[1] "red apple"
[1] "red mango"
[1] "red grapes"
[1] "pink apple"
```

```
[1] "pink mango"
[1] "pink grapes"
[1] "orange apple"
[1] "orange mango"
[1] "orange grapes"
```

In [38]:

```r
# R funnction

sum <- function(a,b){
    print(a+b)
}
sum(2,4)
sum(10,20)

# Default Parammeter value

my <-  function(country ="india"){
    paste("I am from",country)
}

my("Australiya")
my()  # Default
```

```
[1] 6
[1] 30
```

'I am from Australiya'

'I am from india'

In [43]:
```r
# ------Nested function -------
# two way to create
        # * calll function with another function
nested <- function(x,y){
    a <- x+y
    return(a)
}
nested(nested(2,2),nested(10,10))   # (2+2) + (10+10) => 4+20 = 24

        # * write function with a function

outer <-function(x){
    inner <- function(y){
        a < x+y
        return(a)
    }
    return(inner)
}

new <- outer(7)
print(new(7))
```

```
24

[1] 200
```

In [44]:
```r
# factorial function using recursion

fac <- function(n){
    if(n == 0 | n == 1 ){
        return(1)
    }else{
        return(n * fac(n-1))
    }
}

fac(5)
```

```
120
```

In [45]:
```r
#============R Data Structure ==============================
```

In [63]:

```r
#R Vector
      # a list of item which same type like tuple() in python
      # use c()

fruit <- c("apple","banana","pear","almond","juice","mango")
num <- c(1,2,3)
fruit
num

# vector size
paste("length of vector =>",length(fruit))

#sort the vector
sort(fruit)  # it can't change the original vector

#Acces the value
fruit[0]  # index not start at 0
fruit[1]
fruit[c(1,3)]  # acces from 1 to 2

#Change the items
fruit[1] <- "carrot"
fruit
```

'apple' · 'banana' · 'pear' · 'almond' · 'juice' · 'mango'

1 · 2 · 3

'length of vector => 6'

'almond' · 'apple' · 'banana' · 'juice' · 'mango' · 'pear'

'apple'

'apple' · 'pear'

'carrot' · 'banana' · 'pear' · 'almond' · 'juice' · 'mango'

In [70]:
```r
# R List

        # list contain the diffrent type of data
        # use list()

fruit <- list("apple","banana","pear","almond","juice","mango")
fruit
fruit[2]
fruit[2] <- "carrot"
fruit
length(fruit)

# Check present in list or not
"apple" %in% fruit    # return true

# Add item in list
append(fruit,"papaya")  # but it can not add original list
print("original list")
fruit
```

1. 'apple'
2. 'banana'
3. 'pear'
4. 'almond'
5. 'juice'
6. 'mango'

1. 'banana'

1. 'apple'

2. 'carrot'
3. 'pear'
4. 'almond'
5. 'juice'
6. 'mango'

6

TRUE

1. 'apple'
2. 'carrot'
3. 'pear'
4. 'almond'
5. 'juice'
6. 'mango'
7. 'papaya'

```
[1] "original list"
```

1. 'apple'
2. 'carrot'
3. 'pear'
4. 'almond'
5. 'juice'
6. 'mango'

In [ ]:

In [23]:

```r
# R martix

  # A matrix is 2D data set
    # Use matrix()

mat <- matrix(c(1,2,3), nrow=3,ncol=2)
mat
mat2 <- matrix(c(1,2,3,4,5,6), nrow=3,ncol=3)
mat2

m <- matrix(c("apple","orange","pear","grapes","mango","almond","10rs","20rs","40rs"),nrow=3,ncol=3)
m
# acces items
m[1,1]  #1st row 1st column


#Acces more than one row
m[c(1,2),]


#Acces more than onn col
m[,c(1,2)]

#Add row and Column -> use cbind()
new <-cbind(m,c("berry","starwberry"))
new

new1 <- rbind(m,mat2)
new1

#Remove row and col
m <- m[-c(-1),-c(1)]
m

#check item present in matrix
"grapes" %in% m
"apple" %in% m

#size of matix
dim(mat)  # retrun size of row and column
length(m) # return total size of element  (row * col)
```

A

matrix:

3 × 2
of type
dbl

1	1
2	2
3	3

A matrix:
3 × 3 of
type dbl

1	4	1
2	5	2
3	6	3

A matrix: 3 × 3 of type chr

apple	grapes	10rs
orange	mango	20rs
pear	almond	40rs

'apple'

A matrix: 2 × 3 of type
chr

apple	grapes	10rs
orange	mango	20rs

A matrix: 3 × 2 of
type chr

apple	grapes
orange	mango
pear	almond

```
Warning message in cbind(m, c("berry", "starwberry")):
“number of rows of result is not a multiple of vector length (arg 2)”
```

A matrix: 3 × 4 of type chr

apple	grapes	10rs	berry
orange	mango	20rs	starwberry
pear	almond	40rs	berry

A matrix: 6 × 3 of type chr

apple	grapes	10rs
orange	mango	20rs
pear	almond	40rs
1	4	1
2	5	2
3	6	3

'grapes' · '10rs'

TRUE

FALSE

3 · 2

2

In [30]:
```r
#R Array
  # Array can more than 2D data set
    #use array()  and dim  -> to specify the dimension
a <- array(c(1:20),dim = c(4,3,4))
a

#dim = c(row,col,how many dimension we want)

#Access array
a[2,3,4]  # a[row,col,matrix_level]

dim(a)

length(a)


#Loop in array
for(i in a){
    print(i)
}
```

```
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  1  2  3  4  5  6  7
 8  9  10  11  12  13  14  15  16  17  18  19  20  1  2  3  4  5  6  7  8
```

```
6
```

```
 4  3  4
```

```
48
```

In [44]:
```r
#R data Frame
     # Data display in a table format
     #Use data.frame()

frame <- data.frame(Training=c("strength","stamina","other"),pulse=c(100,150,200),Duration=c(10,20,15))
frame

#Summary tthe data
print("Summary of data frame")
summary(frame)

#Access the item from data frame
frame[1]
frame[["Training"]]
frame $ pulse  #$ is same as [[""]]

#Add row and column
new <- rbind(frame,c("power",110,110))  # add in row
new

new1 <- cbind(frame,stable=c(1200,2300,2370))
new1
```

A data.frame: 3 × 3

Training	pulse	Duration
<chr>	<dbl>	<dbl>
strength	100	10
stamina	150	20
other	200	15

```
[1] "Summary of data frame"
```

```
   Training             pulse        Duration
 Length:3           Min.   :100   Min.   :10.0
 Class :character   1st Qu.:125   1st Qu.:12.5
 Mode  :character   Median :150   Median :15.0
                    Mean   :150   Mean   :15.0
                    3rd Qu.:175   3rd Qu.:17.5
                    Max.   :200   Max.   :20.0
```

A

data.frame:

3 × 1

Training
<chr>
strength
stamina
other

'strength' · 'stamina' · 'other'

100 · 150 · 200

A data.frame: 4 × 3

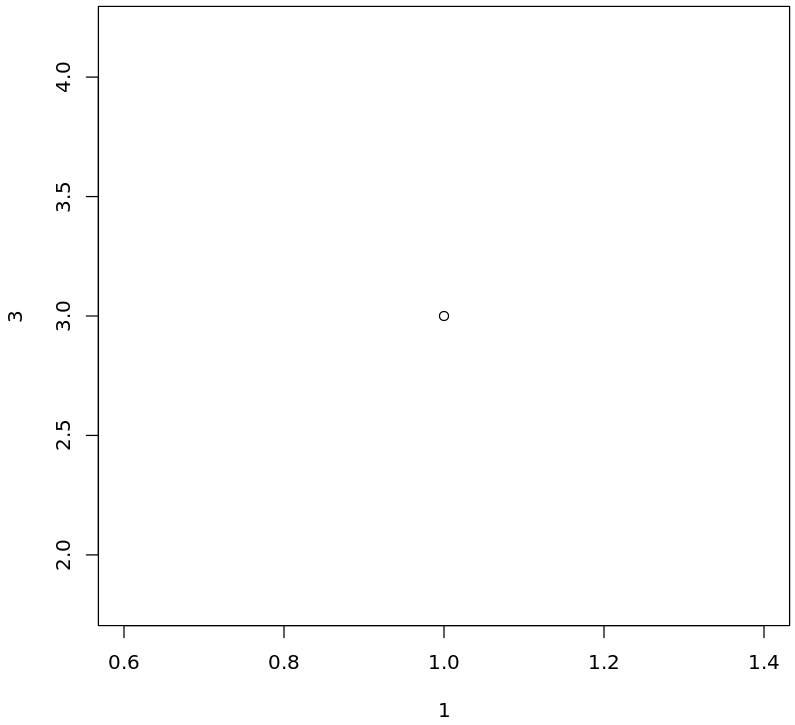
Training	pulse	Duration
<chr>	<chr>	<chr>
strength	100	10
stamina	150	20
other	200	15
power	110	110

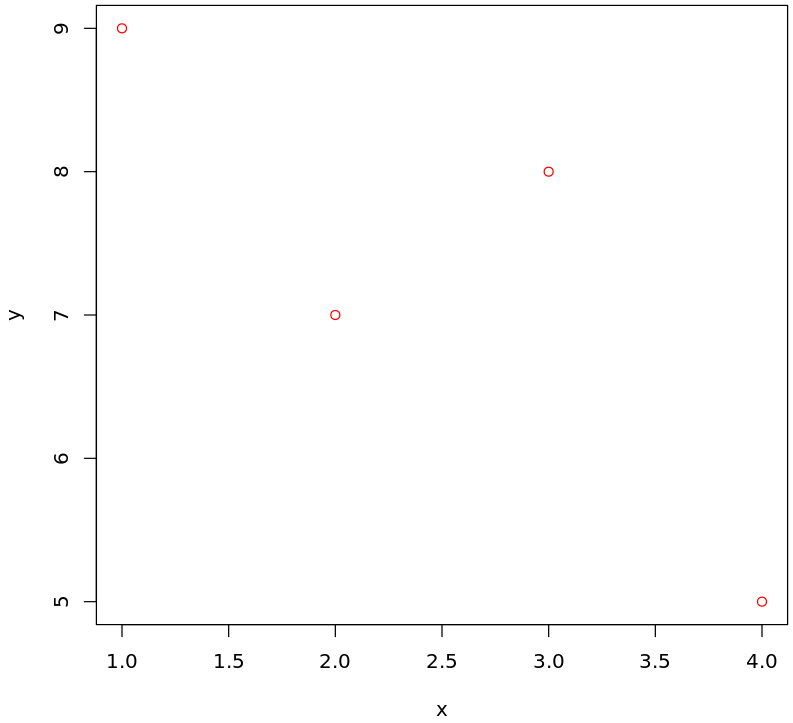
A data.frame: 3 × 4

Training	pulse	Duration	stable
<chr>	<dbl>	<dbl>	<dbl>
strength	100	10	1200
stamina	150	20	2300
other	200	15	2370

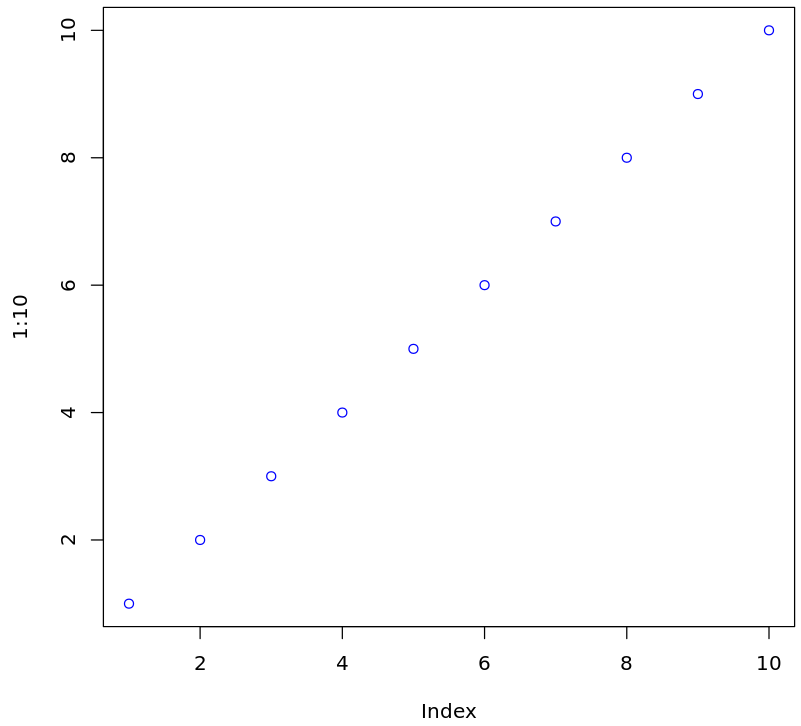
In [45]:
```
#=============R Graphics===============
```

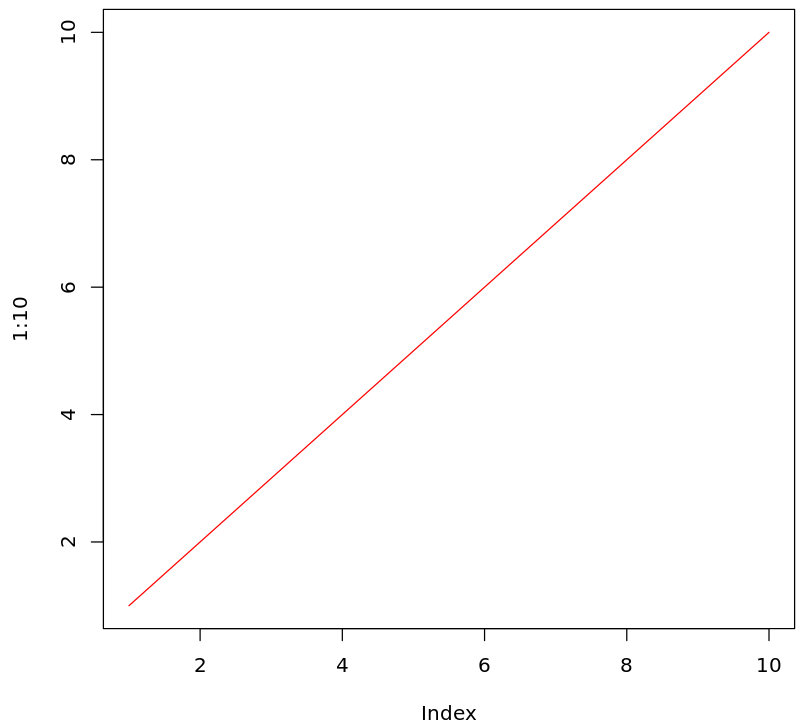
In [51]:
```r
#R plotting
    #use plot(x-axis,y-axis)

plot(1,3)

#To draw more point
x <- c(1,2,3,4)
y <- c(9,7,8,5)
plot(x,y, col="red")
```

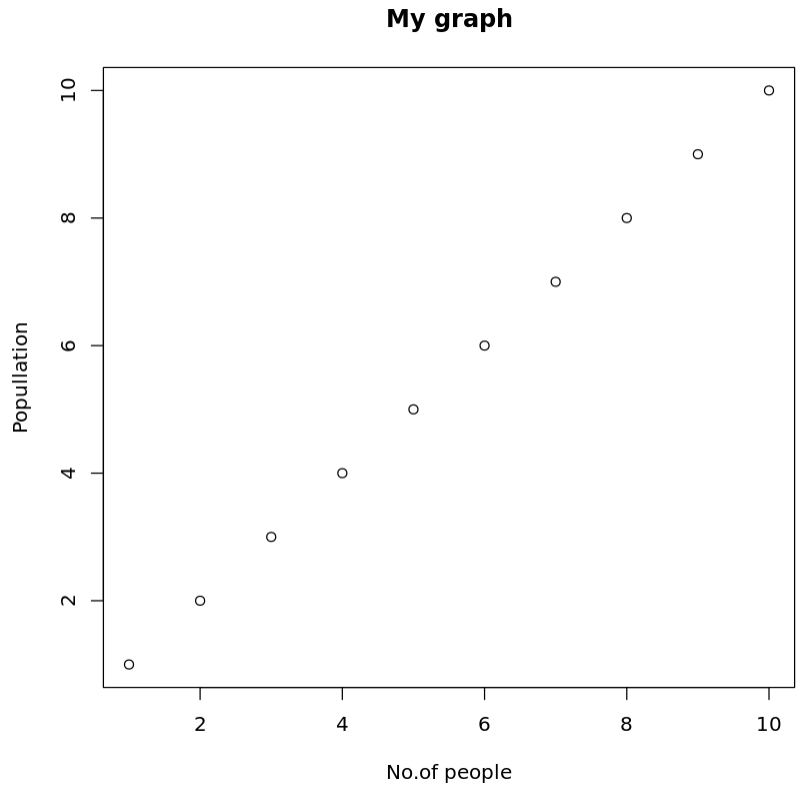



y
9
8
7
6
5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
x

In [56]:
```
#Sequence of point
plot(1:10, col="blue")

# Draw a line graph
plot(1:10,type="l", col="red")

#Labeling
plot(1:10,main="My graph",xlab="No.of people",ylab="Popullation")
```

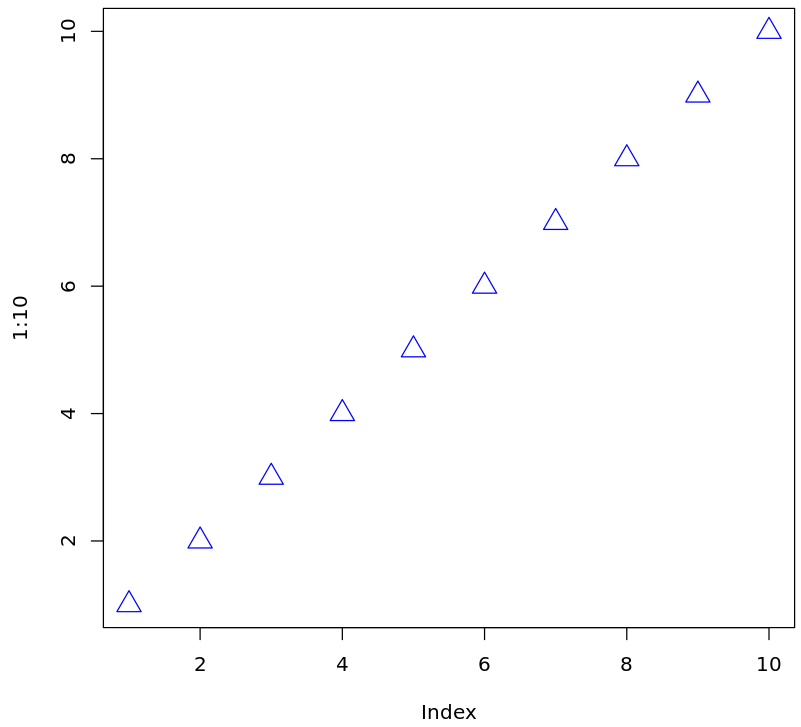



1:10
2
4
6
8
10
Index
2
4
6
8
10

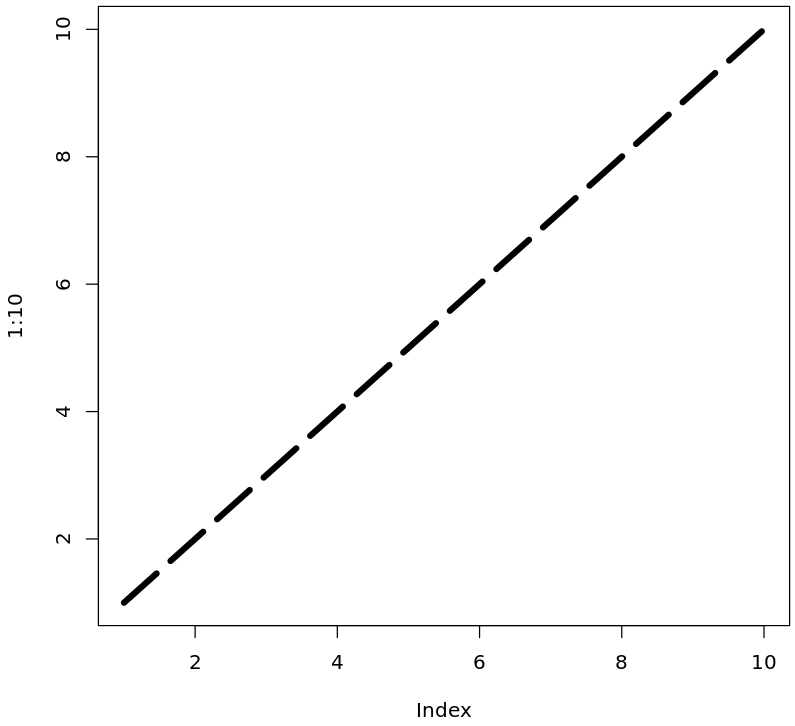


In [ ]:

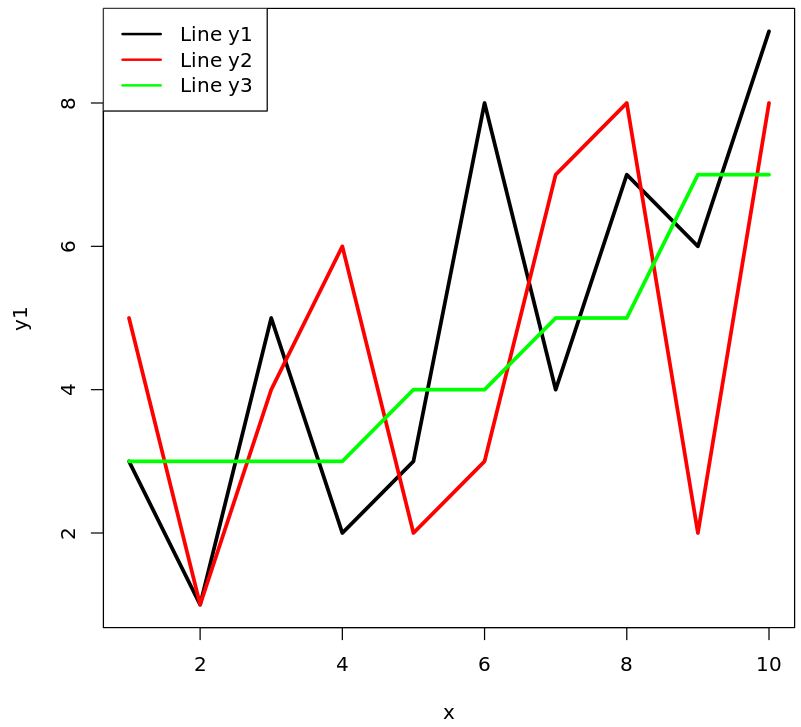
In [1]:
```
# change shape and size
#size -> cex=1(default),0.5(samller),2(larger)
#shape -> pch=0 to 25 type of shapes
plot(1:10,cex=2,pch=24,col="blue")
```



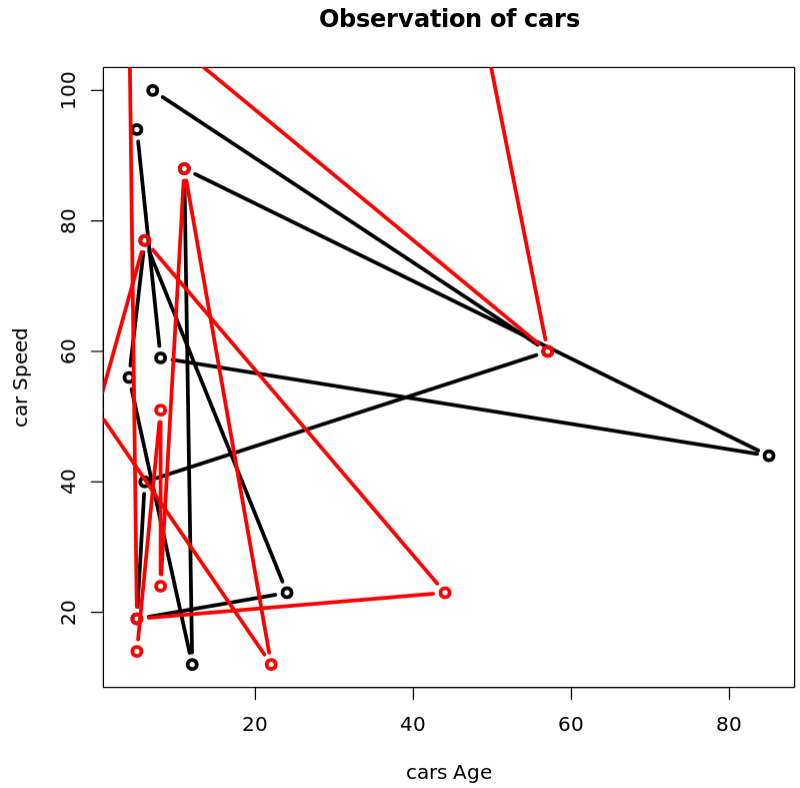
In [4]:
```
#Change line width and style
plot(1:10,type="l",lwd=5,lty=5)

#width:  lwd=?
# Style: lty=0 to 6
        # 0-> remove line
        # 1-> display solid line(default)
        # 2-> dashed line
        # 3-> dotes line
        # 4-> dot-dashed line
        # 5-> long dashed line
        # 6-> two dashed line
```



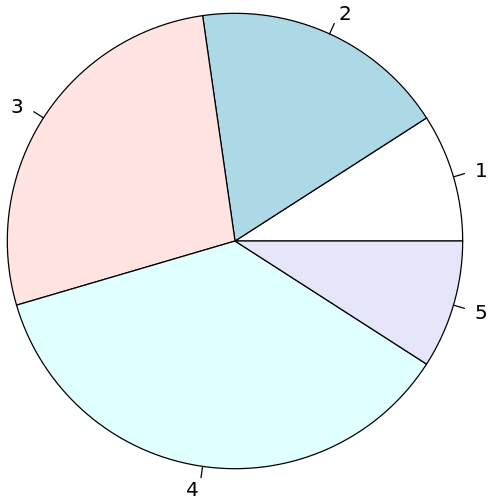
In [36]:
```r
#Multiple lines
x <- 1:10
y1 <- c(3, 1, 5, 2, 3, 8, 4, 7, 6, 9)
y2 <- c(5, 1, 4, 6, 2, 3, 7, 8, 2, 8)
y3 <- c(3, 3, 3, 3, 4, 4, 5, 5, 7, 7)
plot(x, y1, type = "l",lwd=3)                                  # Draw first line
lines(x, y2, type = "l", col = "red",lwd=3)                    # Add second line
lines(x, y3, type = "l", col = "green",lwd=3)                  # Add third line

#Add Legend
legend("topleft",legend = c("Line y1", "Line y2", "Line y3"),
       col = c("black", "red", "green"),lty = 1,lwd=2)
```



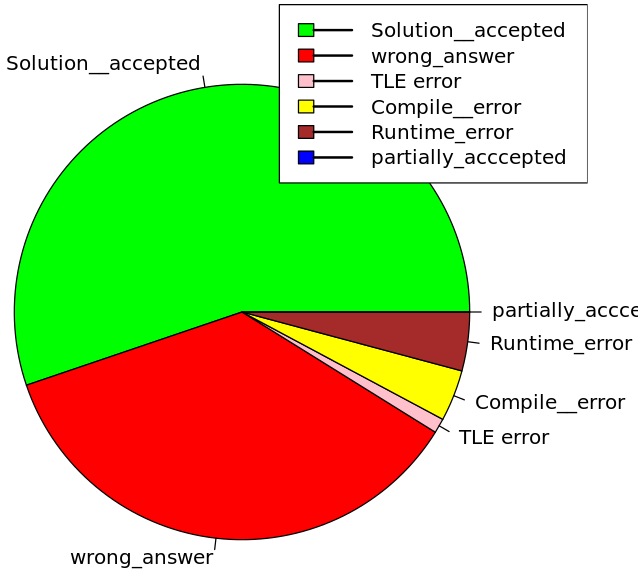
In [46]:
```r
#Show Observation of cars
x <- c(5,8,85,11,12,4,6,24,5,6,57,7)
y <- c(94,59,44,88,12,56,77,23,19,40,60,100)

x1 <- c(5,8,8,11,22,0,6,44,5,1,57,7)
y1 <- c(14,51,24,88,12,51,77,23,19,400,60,110)
plot(x,y,main="Observation of cars",xlab="cars Age",ylab="car Speed",type="b",lwd=3)

#To comapare two chart using points()
points(x1,y1,type="b",col="red",lwd=3)
```



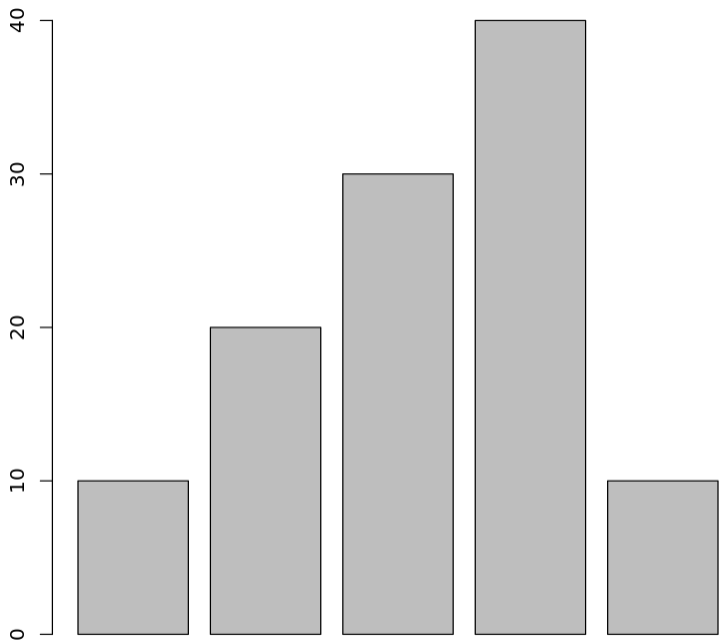
In [54]:
```r
#R Pie chat
x <- c(10,20,30,40,10)
pie(x)

#Codechef pie chart
x <-c(106,69,2,7,8,0)
c <-c("green","red","pink","yellow","brown","blue")
l <- c("Solution__accepted","wrong_answer","TLE error","Compile__error","Runtime_error","partially_acccepted"
pie(x,label=l,main="Codechef Submission Answer",col=c)

legend("topright",legend = l,fil = c,lwd=2)
```

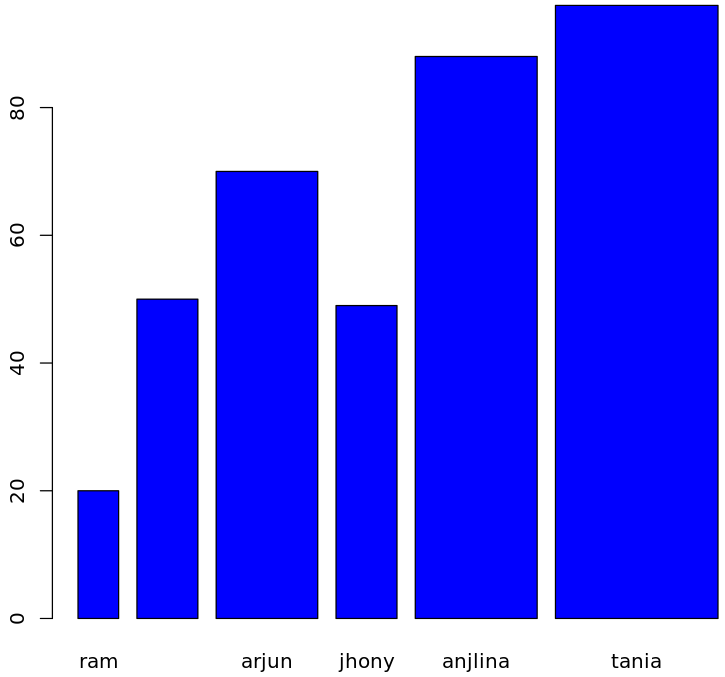


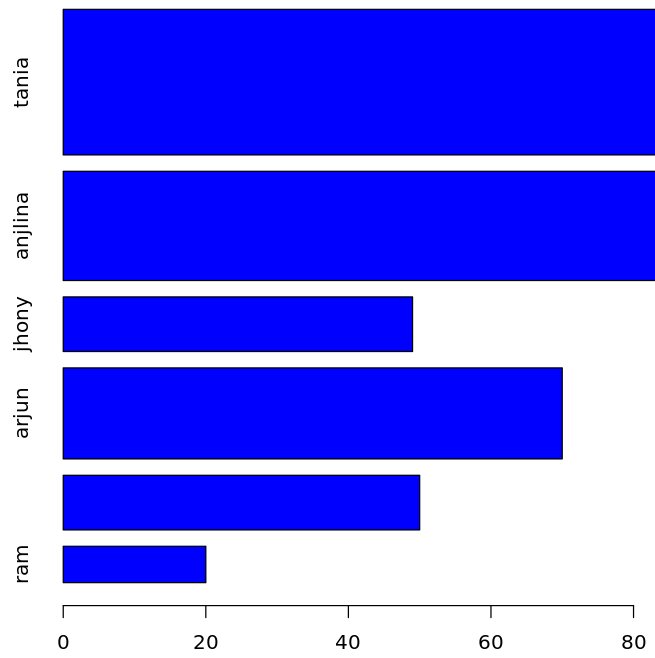
Codechef Submission Answer



In [59]:
```r
#R Bar chart
x <- c(10,20,30,40,10)
barplot(x)

#Marks bar chart
name <- c("ram","shyam","arjun","jhony","anjlina","tania")
mark <- c(20,50,70,49,88,96)
width <- c(2,3,5,3,6,8)
barplot(mark,names.arg =name, col="blue",width=width)

#Horizontal Bar chart
barplot(mark,names.arg =name, col="blue",width=width,horiz=TRUE)
```




0
20
40
60
80
ram
arjun
jhony
anjlina
tania



In [ ]: