#벡터, 행렬, 데이터프레임, 리스트
> x1<-c(1,2,3); x2<-c(4,5,6); x3<-c('a','b','c')
> A1<-matrix(1:6, nrow=2); A2<-matrix(1:6, nrow=2, byrow=TRUE)
> C1<-data.frame(c(1,2,3), c(4,5,6)); C2<-data.frame(x1,x2)
> D1<-list(x2, x3); D2<-unlist(D1)

> x1; class(x1)
> x3; class(x3)
> A1; class(A1)
> C1; class(C1)
> D1; class(D1)
> D2; class(D2)
> A1[1,]
> A2[2,]
> t(A1[,1:2])*A2[,1:2]
> t(A1[,1:2])%*%A2[,1:2]
> C1; C2
> C2[[1]]; C2[,1]; C2$x1
> C2[C2$x1>2,]
> D1[[1]]
> D2[[2]][2:3]

점수(x)가 60 이상이면 pass, 그렇지 않으면 fail으로 출력하는 if-else문
> x=82
> if(x>=60) {print("pass")} else {print("fail")}

#학생수=5명, 점수가 80이상이면 "pass", 아니면 "fail"
> x<-c(90,50,85,100,70)
> ifelse(x>=80, "pass", "fail")

x=c(10,12,15) 인 경우 x[i]를 출력하는 for문
> x<-c(10,12,15)
> for(i in 1:3){ print(x[i])}

x=c(10,12,15) 인 경우 x[i]/3의 나머지를 출력하는 for문
> x<-c(10,12,15)
> for(i in 1:3){ print(x[i]%%3)}

#x=(1,2,3,4,5), y=(1,3,5,9,11), z=(2,4,6,8,10), 두개의 산점도(선(xz, 빨강), 점(xy)) 겹쳐 그리기
> x<-1:5, y<-c(1,3,5,9,11), z<-c(2,4,6,8,10)
> plot(x,y)
> lines(x,z,col='red')

#x=(1,2,3,4,5), y=(1,3,5,9,11), z=(2,4,6,8,10), 두개의 산점도(점(xz, 빨간), 점(xy)) 겹쳐 그리기
```
> plot(x,y)
> points(x,z, col='red')
```

#x=(10,15,20,22,27,33,34,34,35,37,45,46), boxplot, histogram, 줄기-잎 그래프 그리기
```
> x<-(10,15,20,22,27,33,34,34,35,37,45,46)
> boxplot(x)
> hist(x, right = FALSE, breaks=c(10,20,30,40,50))
> stem(x)
```

#사용자 정의함수 eg1 ~ eg6
eg1) 함수를 실행하면 "Hello !" 가 프린트되는 이름이 'greet' 라는 함수 만들기
```
> greet<-function(){
>         print("Hello !")
      }
greet()
```

eg2) 함수에 x를 입력하여 실행하면 'x가 7의 배수인지 아닌지'의 결과가 구해지는 이름이 'seven' 이라는 함수 만들기
```
> seven<-function(x){
>     if(x%%7==0){cat(x,"=7의 배수")
>     } else {cat(x,"=7의 배수아님") }
> }
> seven(114)
```

eg3) 함수에 x를 입력하여 실행하면 'x<0 이면 0, x≥0이면 x'의 결과가 구해지는 이름이 'relu' 라는 함수 만들기
```
> relu<-function(x){
>     if (x<0) {return(0)
>     } else {return(x)}
> }
> relu(-2.1); relu(1.5)
```

eg4) 함수에 x1,x2을 입력하여 실행하면 '100*(x1+x2)'의 결과가 구해지는 이름이 'hundr' 이라는 함수 만들기
```
> hundr<-function(x1,x2){
>     x=100*(x1+x2)
>     return(x)
> }
> hundr(1, 3)
> y=hundr(2.1, 2.4) # hundr(2.1, 2.4) 결과를 y에 저장
```

eg5) 숫자 x를 입력하여 실행하면. x-7, x+7이 구해지는 이름이 'seven2' 이라는 함수 만들기.

```
> seven2<-function(x){
>   y=numeric(2)
>   y[1]=x-7;   y[2]=x+7
> return(y)
> }
> seven2(11)
```

eg6) 함수형 수치형벡터 x를 입력하여 실행하면 평균과 분산이 구해지는 이름이 'summr' 이라는 함수 만들기

```
> summr<-function(x){
>     y=numeric(2) # 크기가 2인 벡터 만들기.
>     y[1]=mean(x); y[2]=var(x)
>       return(y)
> }
> x<-c(1,3,5,7,8)
> y=summr(x)
> y
```

```
# 도수분포표:  범주형자료: 10명의 선호도 결과 : (저,중,상)=(L, M, H)
> x<-c('L' ,'H', 'M', 'M', 'L', 'H', 'H', 'M', 'H', 'M')
> table(x) # 도수
> prop.table(table(x)) # 상대도수
```

```
#기초통계
> x<-c(10,15,45,34,20,46,33,34,35,37,22,27)
> fivenum(x)
> summary(x)
```

```
#apply 함수
> A<-matrix(1:12, ncol=4); A
> apply(A,1,mean) # mean, min, max, sd
> apply(A,2,mean)
> B=A; B[2,3]=NA; B
> apply(B,1,mean)
> apply(B,1,mean, na.rm=T)
```