

R 프로그래밍

(7주차)

2016. 04. 16(토)

장운호

(ADP 002-0004)

목차

※ 지난 주 복습

I. which 내장함수의 활용

II. 행렬(Matrix)

III. 배열(Array)

IV. 행렬/배열 활용 연습

※ 벡터 Indexing 개요

벡터의 원소들을 개별적으로, 또는 부분집합으로 다룰 필요가 있을 때, 객체명 옆에 대괄호("[")를 열고, 아래의 Rule에 따른 Index방법을 표기한 후, 대괄호("]")를 닫음으로써 지정이 가능함

[벡터 인덱싱(Indexing) Rule 요약]

- 1) 양의 정수가 사용되면, 해당 위치의 원소를 의미한다.
- 2) 빈칸으로 둔 경우는 모든 원소를 지정하는 것이 된다.
- 3) 음의 정수가 사용되면, 해당 위치의 원소가 제외한다는 의미다.
- 4) 조건식을 넣으면 조건식의 참(TRUE)인 원소가 선택된다.
- 5) 정수로 이뤄진 벡터를 넣으면, 해당 벡터의 위치에 있는 원소를 선택한다.

※ 연산자 우선순위(Precedence)

괄호를 활용하여 우선순위를 적절히 조절하는 것이 중요함.

연산자	설명	적용순서	참조
()			
^	N제곱 계산	오른쪽 → 왼쪽	
-X, +X	음수/양수 기호	왼쪽 → 오른쪽	
:	정수 벡터 생성	from:to	
%%	나머지 계산	"	%in%, %*%
*, /	곱하기, 나누기 계산	"	
+, -	더하기, 빼기 계산	"	
<, >, <=, >=, ==, !=	비교 연산자 (TRUE, FALSE 반환)	"	
!	부정 조건 (상동)	"	
&	And 조건 (상동)	"	&&
	Or 조건 (상동)	"	
->	우측 할당	"	-> >
<-	좌측 할당	오른쪽 → 왼쪽	<<-
=	좌측 할당	오른쪽 → 왼쪽	

※. 주요 내장함수

주요 내장함수의 용도, Argument의 개수 등을 반복적인 연습을 통해 숙달 필요

구분	기능	함수
수식	절대값	abs(x)
	제곱근	sqrt(x)
	N제곱, n제곱근	x^n , $x^{(1/n)}$
	올림/내림	ceiling(x), floor(x)
	지수함수값	exp(x)
	소수점 n자리 반올림	round(x, digits=n)
	자연로그, 상용로그	log(x), log10(x)
	숫자 ↔ 문자열 전환	as.numeric("x"), as.character(x)
문자열	문자열 글자수	nchar("x")
	문자열 일부선택	substr("x", 시작위치, 끝위치)
	단어 붙이기	paste("x", "y", sep=" ")
		paste0("x","y")



I . which 내장함수의 활용

1. which 내장함수의 활용

Which함수는 인수를 논리식으로 받아, 참(TRUE)인 값의 위치 번호(Index)를 정수 또는 나중엔 배우게 될 배열(Array)의 인수 형태로 반환해주는 함수임.

```
> x <- 1:10
```

```
> which(x > 5)
```

```
> which(x[x>5])
```

```
# Error in which(x[x > 5]) : argument to 'which' is not logical
```

```
> x <- c(1,NA,2,NA,3)
```

```
> which(is.na(x))
```

```
> which(!is.na(x))[3]
```


※ 돌발(?) Quiz

다음중 우측의 결과값이 나오지 않는 것은?

[1] 3 5

단, `myVector <- c(-5, -3, -1, 1, 3, 5)`

1. `myVector[myVector > 1]`
2. `myVector[myVector >= 3]`
3. `myVector[!(myVector <= 1)]`
4. `myVector[!myVector <= 1]`
5. `which(myVector > 1)`
6. `myVector[which(myVector > 1)]`
7. `myVector[c(FALSE,FALSE,FALSE,FALSE,TRUE,TRUE)]`
8. `c(myVector[5], myVector[6])`
9. `seq(-5,5,2)[5:6]`
10. `which(myVector[myVector > 1])`



II. 행렬 (Matrix)

1. 행렬 (Matrix)

벡터에 행과 열 속성을 지정하여
표형태로 정리된 배열(rectangular Array)데이터

```
> myMatrix <- matrix(1:20, nrow=4, ncol=5)
```

```
> myMatrix # 열(Column) 우선 원칙 적용
```

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

```
> myMatrix <- matrix(1:19, nrow=4, ncol=5)
```

Warning message:

In matrix(1:19, nrow = 4, ncol = 5) :
data length [19] is not a sub-multiple or
multiple of the number of rows [4]

2. 행렬 관련 주요 내장함수

행렬의 행이름과 열이름을 지정하는 함수, 합산 및 연산 함수 등 행렬관련 내장함수 사용법 숙지가 필요함.

구분	기능	함수
생성	매트릭스 만들기	<code>matrix(x, nrow, ncol, byrow=FALSE)</code>
	벡터를 결합하여 행렬 생성	<code>cbind(x,y); rbind(x,y)</code>
이름 지정	원소(elements)의 이름 지정	<code>name(x)</code>
	행이름 지정	<code>rownames(x)</code>
	열이름 지정	<code>colnames(x)</code>
	Dimension 이름 동시 지정	<code>dimnames(x) <- list(x,y)</code>
합산	행합계	<code>rowSums(x)</code>
	열합계	<code>colSums(x)</code>
매트릭스 연산	매트릭스 곱하기	<code>%*%</code>
	전치행렬 구하기	<code>t(x)</code>
	행렬식 구하기	<code>det(x)</code>
	역행렬 구하기	<code>solve(x)</code>

3. 행렬 Indexing Rule

벡터 Indexing Rule의 “원소”를 “**행 과/또는 열**”로 바꾸면 정확히 일치한다.

- 1) 양의 정수가 사용되면, 해당 위치의 **행과 열**을 의미한다.
- 2) 빈칸으로 둔 경우는 모든 **행 또는 열**이 된다.
- 3) 음의 정수가 사용되는 해당 위치의 **행 또는 열**은 제외한다는 의미다.
- 4) 조건식을 넣으면 조건식의 참(TRUE)인 **행과 열**이 선택된다.
- 5) 정수로 이뤄진 벡터를 넣으면,
해당 벡터의 위치에 있는 **행 또는 열**을 선택한다.

※ 행이나 열의 이름이 지정되어 있으면, 이름 지정 時 지정된 이름을 가진 행 or 열이 선택됨.

참고자료) R과 Knitr를 활용한 데이터 연동형 문서 만들기 (고석범 저, 일부 수정 반영)



Ⅲ. 배열 (Array)

1. 배열 (Array)

An **array** is a systematic arrangement of similar objects, usually in rows and columns. (by Wikipedia)

[한글 Wikipedia의 배열 정의]

컴퓨터 과학에서 **배열**(Array, 配列·排列)은 번호(인덱스)와 번호에 대응하는 데이터들로 이루어진 자료 구조를 나타낸다.

- 일반적으로 배열에는 같은 종류의 데이터들이 순차적으로 저장되어, 값의 번호가 곧 배열의 시작점으로부터 값이 저장되어 있는 곳까지의 상대적인 위치를 나타낸다.
- 대부분의 프로그래밍 언어에서 사용할 수 있는 가장 기초적인 자료 구조로, 기본적인 용도 외에 다른 복잡한 자료 구조들을 표현하기 위해서 또는 행렬, 벡터 등을 컴퓨터에서 표현하는 용도 등으로도 사용된다.

2. 배열 생성 및 Indexing

3차원 이상의 배열 생성을 위해서는 배열 Indexing에 대한 이해와 연습이 필요.

[배열 생성]

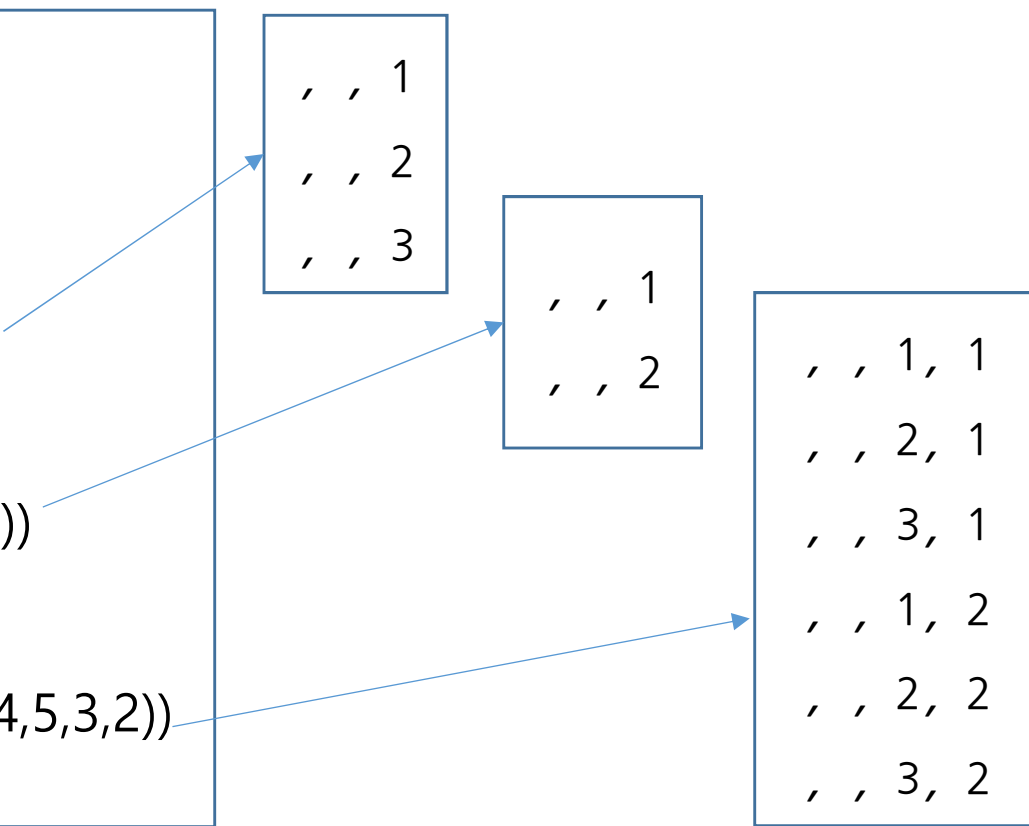
```
array(data = NA,  
      dim = length(data),  
      dimnames = NULL)
```

```
myArray <- array(60:1, dim=c(4,5,3))  
myArray
```

```
shortArray <- array(60:1, dim=c(4,5,2))  
shortArray
```

```
fourDimArray <- array(1:120, dim=c(4,5,3,2))  
fourDimArray
```

[배열 Indexing]



,	,	1
,	,	2
,	,	3

,	,	1
,	,	2

,	,	1, 1
,	,	2, 1
,	,	3, 1
,	,	1, 2
,	,	2, 2
,	,	3, 2

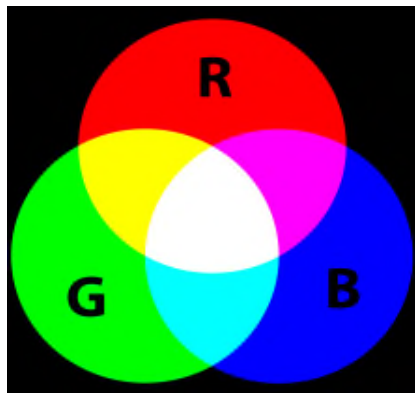


IV. 행렬 / 배열 활용 사례

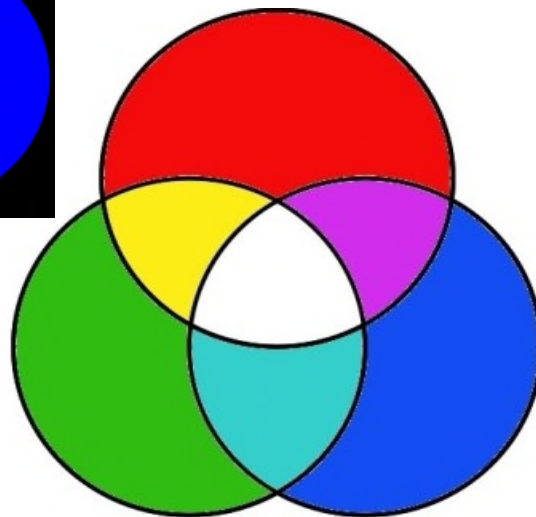
1. 색광의 3원색과 색료의 3원색

3원색 각각의 강도를 0에서 255사이의 숫자 혹은 다양한 수치의 형태로 표시가 가능함.

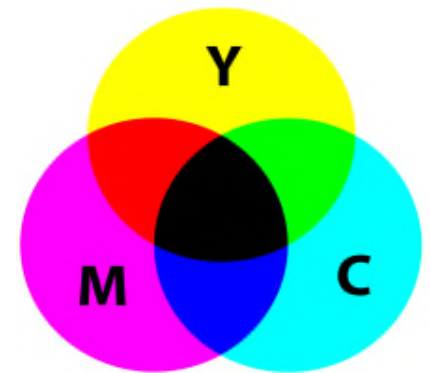
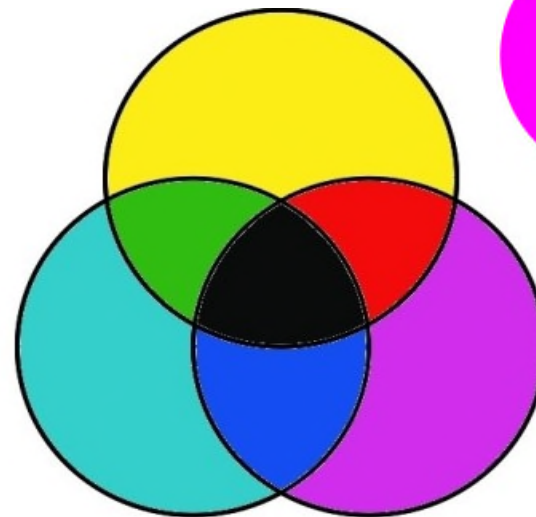
※ 각각의 pixel에 RGB 3가지 숫자가 조합된 3차원 배열(Array)로 이미지 표시 가능



가산 혼합

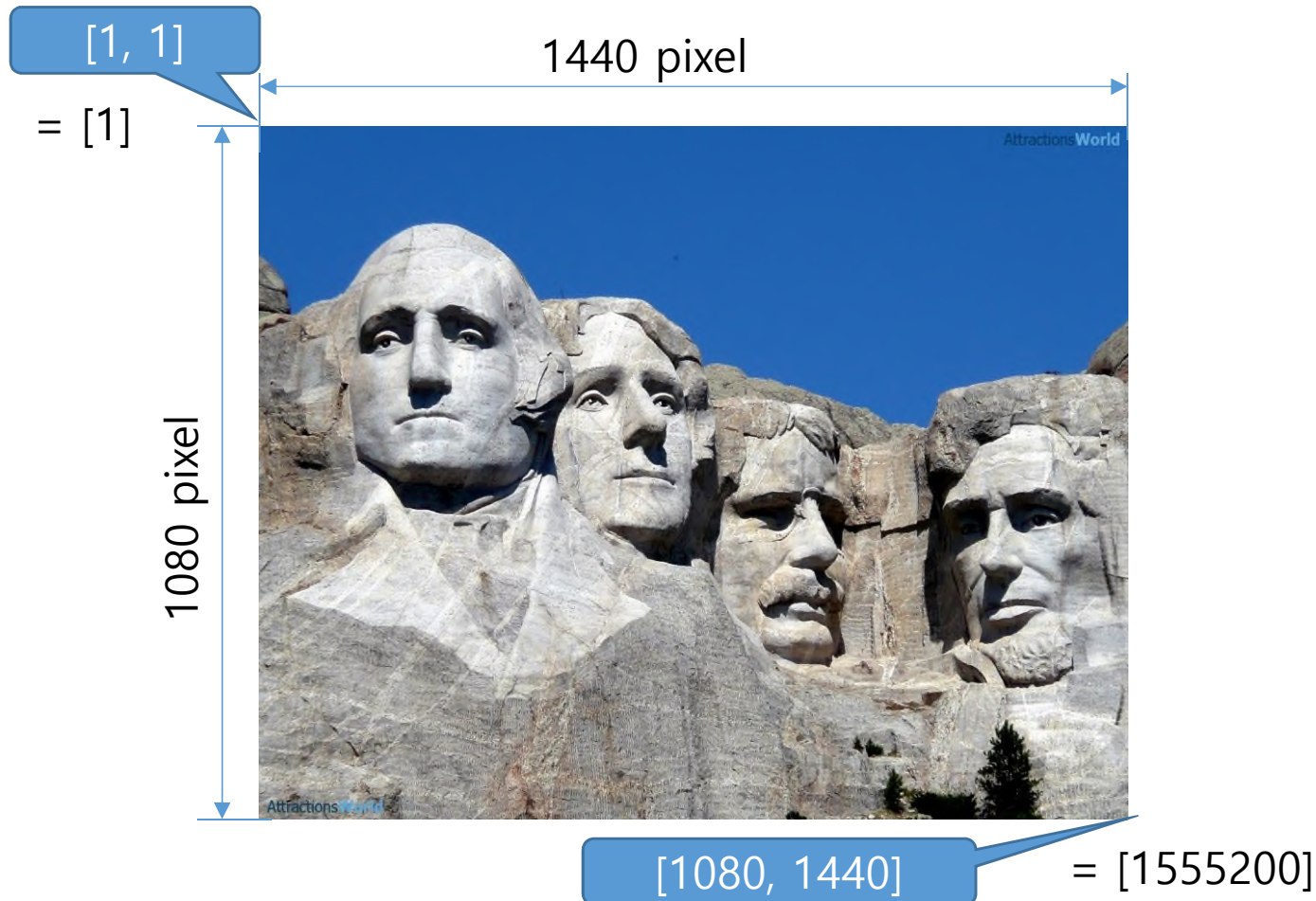


감산 혼합



2. Image 파일

이미지(Image) 파일은 pixel을 행과 열로 배치한 것으로 볼 수 있어, 본질적으로 매트릭스이며, 따라서 일종의 배열(Array)이라고도 볼 수 있음.



3. R Colors

R에서 color의 표시는 0에서 255까지의 숫자외에 16진수로된 색번호를 활용하여서도 나타낼 수 있음.

R colors

자료) <http://research.stowers-institute.org/efg/R/Color/Chart/ColorChart.pdf>

1	white	#FFFFFF	255	255	255
2	aliceblue	#F0F8FF	240	248	255
3	antiquewhite	#FAEBD7	250	235	215
4	antiquewhite1	#FFEFD8	255	239	219
5	antiquewhite2	#EEDFCC	238	223	204
6	antiquewhite3	#CDC0B0	205	192	176
7	antiquewhite4	#8B8378	139	131	120
8	aquamarine	#7FFFD4	127	255	212
9	aquamarine1	#7FFFD4	127	255	212
10	aquamarine2	#76EEC6	118	238	198
11	aquamarine3	#66CDAA	102	205	170


51	chartreuse4	#458B00	69	139	0
52	chocolate	#D2691E	210	105	30
53	chocolate1	#FF7F24	255	127	36
54	chocolate2	#EE7621	238	118	33
55	chocolate3	#CD661D	205	102	29
56	chocolate4	#8B4513	139	69	19
57	coral	#FF7F50	255	127	80
58	coral1	#FF7256	255	114	86
59	coral2	#EE6A50	238	106	80
60	coral3	#CD5B45	205	91	69
61	coral4	#8B3E2F	139	62	47

4. jpeg 패키지

```
install.packages("jpeg")  
library(jpeg)  
mtrushBMP <- readJPEG("mount-rushmore-national-memorial.jpg")  
str(mtrushBMP)  
class(mtrushBMP)  
dim(mtrushBMP)  
range(mtrushBMP[, , 1])  
max(mtrushBMP)  
min(mtrushBMP)
```

max()와 min() 결과로 보면 RGB를 0과 1사이의 수치값으로 나타내는 것으로 예상됨.
따라서 일부를 지우려면, 모두 1로 바꾸면 가능할 것으로 예상됨. (지운다 -> "하얀색으로 만든다"와 같은 의미로 이해함)

4. jpeg 패키지



```
mtrushBMP[100:200,100:200,1] <- 1 # Red
mtrushBMP[100:200,100:200,2] <- 1 # Green
mtrushBMP[100:200,100:200,3] <- 1 # Blue

# writeJPEG함수의 target 아규먼트로 이미지 파일 저장 가능
writeJPEG(mtrushBMP, target="mtrushJPG.jpg")
```

※ working director에 저장된 jpg이미지를 탐색기로
더블클릭하여 확인 가능.

4. jpeg 패키지

100 200



4. jpeg 패키지

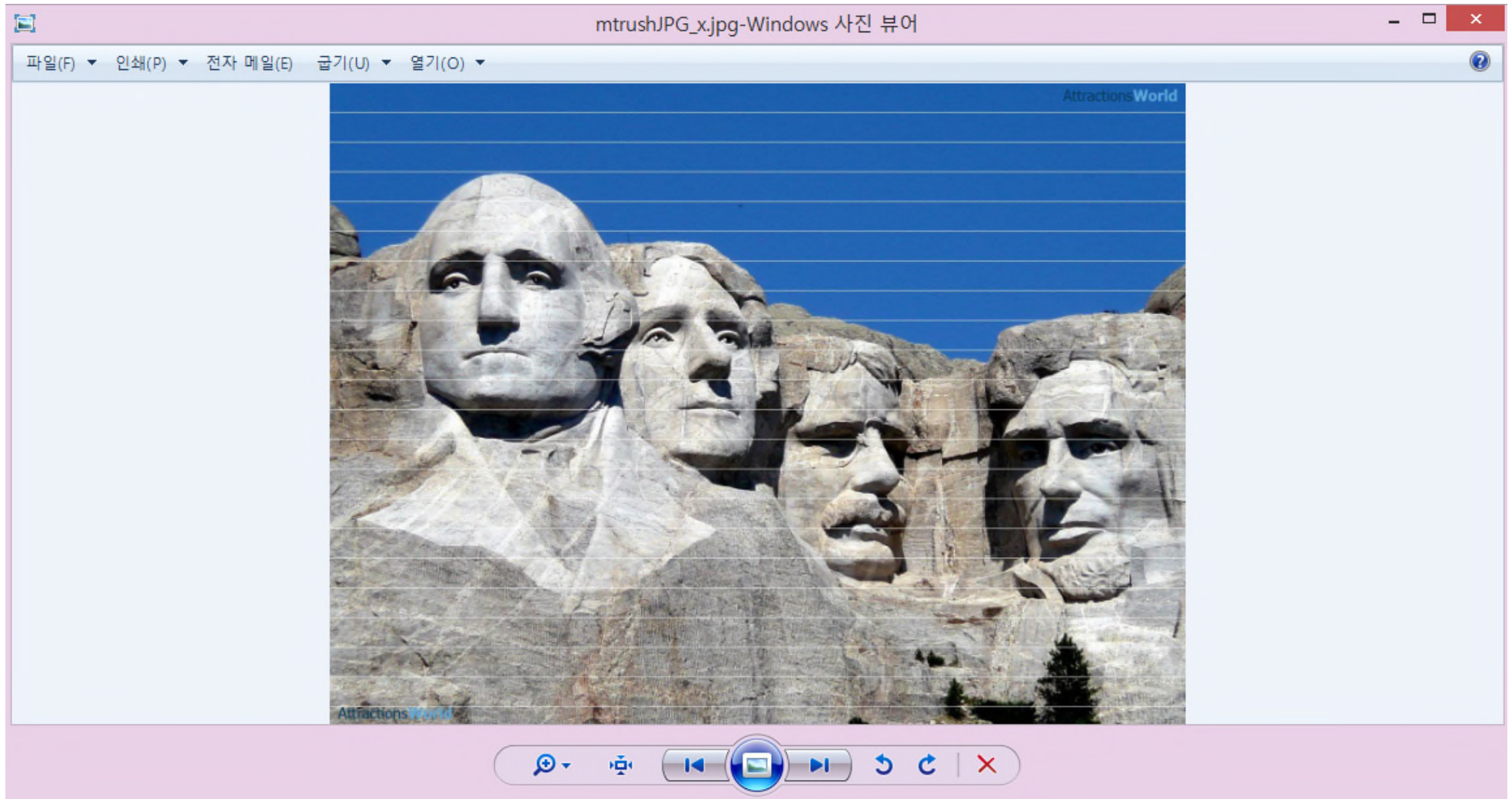
##흐름제어 연습 with jpeg package-----

```
mtrushBMP <- readJPEG("mount-rushmore-national-memorial.jpg")
```

```
for (i in 1:dim(mtrushBMP)[1]) {  
  if ( i %% 50 == 0) {  
    mtrushBMP[i,,1] <- 1  
    mtrushBMP[i,,2] <- 1  
    mtrushBMP[i,,3] <- 1  
  }  
}
```

```
writeJPEG(mtrushBMP,target="mtrushJPG_x.jpg")
```

4. jpeg 패키지



4. jpeg 패키지

```
mtrushBMP <- readJPEG("mount-rushmore-national-memorial.jpg")
```

```
for (y in 1:dim(mtrushBMP)[2]) {  
  if ( y %% 50 == 0) {  
    mtrushBMP[,y,1] <- 1  
    mtrushBMP[,y,2] <- 1  
    mtrushBMP[,y,3] <- 1  
  }  
}
```

```
writeJPEG(mtrushBMP,target="mtrushJPG_y.jpg")
```

4. jpeg 패키지



End of Document.

감사합니다.