

[R] R - 기본적인 함수 정리(출력,인덱싱,길이반환,문자열비교 등등)

노트북: [TIL-MY]

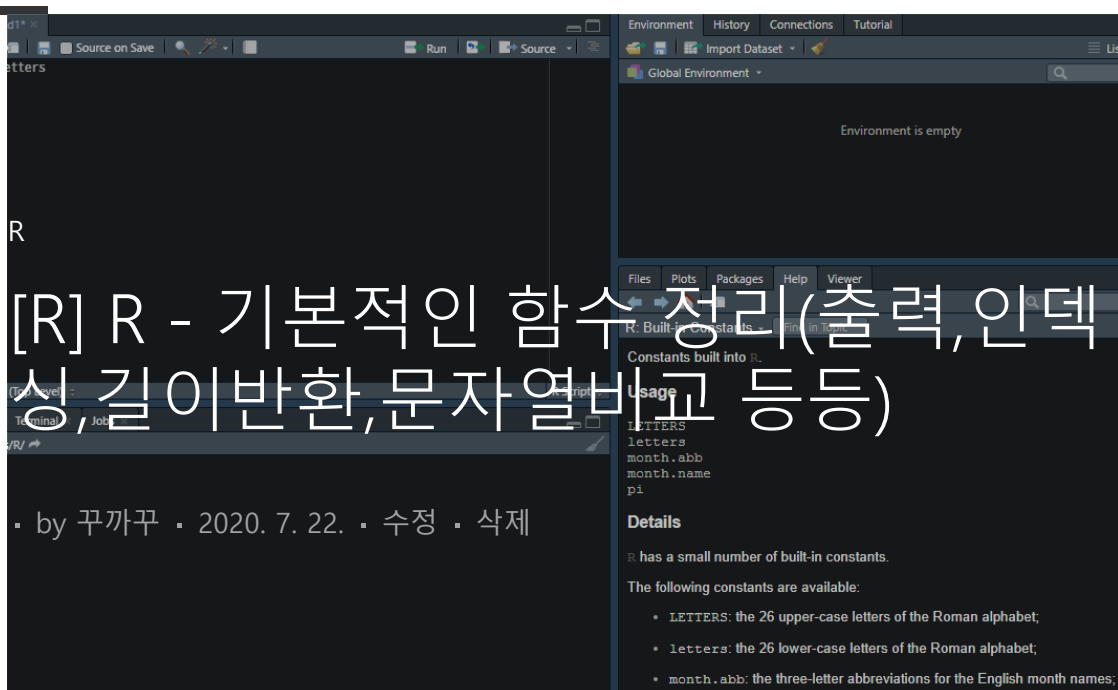
만든 날짜: 2020-07-22 오전 8:42

URL: <https://continuous-development.tistory.com/32>

## 나무늘보의 개발 블로그

홈


태그



#?value - 함수에 대한 사용법

분류 전체보기 

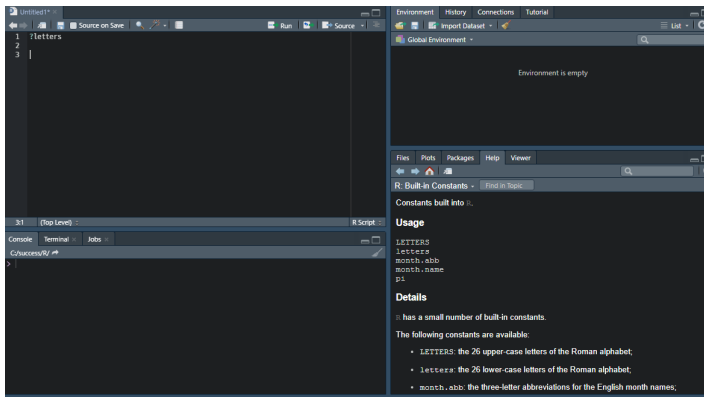
Python

Database 

ASP.NET

Algorithm

Deep learning



## # package - 함수(function) + 데이터셋(dataset)

### 패키지 사용법

```
4 # package란? 함수(function) + 데이터셋(dataset)
5 install.packages("stringr")
6 library("stringr")
7
```

## #print() - 일반적인 출력문

```
[1] 10
> print('섭섭이')
[1] "섭섭이"
> print('네 이눔')
[1] "네 이눔"
>
```

AWS

ETC..

R 

### 공지사항

글 보실 때 주의사항

: 최근글 : 인기글

[R] R  
에...



2020.07.22

[R] R  
- ...



2020.07.22

[Data  
트...



2020.07.21

[Data  
INSEF



2020.07.20

[Database] DDL(데이터 정..

2020.07.19

최근댓글

태그

UPDATE함수,  
DDL,

## #sprintf(type,value) -특정 규칙에 맞게 변환 출력

```
12 #디버깅 print(), paste(), sprintf, cat()
13 print()
14
15
16 #sprintf(type,value)
17
18 # %d -정수
19 # %f -실수
20 # %s -문자
21
22 sprintf("%d",123)
23 sprintf("Number : %d",100)
24 sprintf("Number : %d, String : %s",100,'jsut do it')
25
26 sprintf("%.2f",123.456)
27 sprintf("%5d",123)
28 sprintf("%5d",12345)
29 sprintf("%5d",123456)
```

10:11 (Top Level) R Script

Console Terminal Jobs

```
C:\success\R/
> sprintf("%d",123)
[1] "123"
> sprintf("Number : %d",100)
[1] "Number : 100"
> sprintf("Number : %d, String : %s",100,'jsut do it')
[1] "Number : 100, String : jsut do it"
> sprintf("%.2f",123.456)
[1] "123.46"
> sprintf("%5d",123)
[1] " 123"
> sprintf("%5d",12345)
[1] "12345"
> sprintf("%5d",123456)
[1] "123456"
>
```

## #cat() - 개행(한 칸이 넘어가는 것)을 하지 않는 디버그 형태

```
> myFunc()
append ... Error in myFunc() : 객체 'total'를 찾을 수 없습니다
> #함수 생성
> myFunc <- function() {
+   total <- 0
+   cat("append ...")
+   for(i in 1:10){
+     total <- total +1
+     cat(i," ... ")
+   }
+   cat("End !!","\n")
+   return(total)
+ }
> # 함수 호출
> myFunc()
append ... 1 ... 2 ... 3 ... 4 ... 5 ... 6 ... 7 ... 8 ... 9 ... 10 ... End !!
[1] 10
```

R 정규표현식  
용법,  
Oracle,  
Oracle SQL,  
날짜함수,  
rollback 사용법,  
사용법,  
R 정규표현식,  
DELETE함수,  
SQL, 인스턴스,  
substr, AWS,  
setequal함수,  
설정, names함수,  
R 기본함수,  
commit 사용법,  
paste함수,  
테이블 생성,  
INSERT함수,  
strsplit,  
sql rollback,  
str\_extract\_all,  
R 함수,  
str\_extract,  
sql commit,  
rep함수,  
length함수

전체 방문자

99

Today : 1

Yesterday : 5

## #is.na(value) - 결측 값 확인

```
122
123 #is_na함수로 결측치 확인(NA)
124 sample_na <- NA
125 sample_na
126
127 is.na(sample_na)
128
129 sample_null <- NULL
130 sample_null
131
132 is.na(sample_null)
133
134
```

```
> #is_na함수로 결측치 확인(NA)
> sample_na <- NA
> sample_na
[1] NA
> is.na(sample_na)
[1] TRUE
> sample_null <- NULL
> sample_null
NULL
> is.na(sample_null)
[1] TRUE
```

## #rep(value,each=?) - 반복하는 함수

each는 각각의 값을 반복하게 한다.

```
> rep(1:10,5)
[1] 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7
[40] 8 9 10
> rep(1:10,each=5)
[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 5 5 5 5 5 6 6 6 6 6 7 7 7 7 7 8 8 8 8 8 9 9 9 9 9 10 10 10 10 10
[40] 10 10 10 10 10
```

#seq(from, to, [by]) 함수 - from에서 to까지 출력하는데 by를 통해 간격을 정해 출력

```
#seq(from, to, by ~ 스텝)
seq(1,10)
seq(2,10,2) # 두번째 값을 띄워서 출력해라
seq(1,10,length.out=3) #
seq(1,10,length.out=5) #length.out 값만큼 분할해서출력한다.
```

```
> seq(1,10)
[1] 1 2 3 4 5 6 7 8 9 10
> seq(1,10,2)
[1] 1 3 5 7 9
> seq(2,10,2)
[1] 2 4 6 8 10
> seq(1,10,length.out=3)
[1] 1.0 5.5 10.0
> seq(1,10,length.out=5) #
[1] 1.00 3.25 5.50 7.75 10.00
```

## #인덱싱으로 값 가져오기

```
157
158 #indexing[]
159 seq_vec02[5]
160
```

```
158 #indexing[]
159 seq_vec02[5]
160
161 seq_vec02[length(seq_vec02)-4]
162 seq_vec02[30]
163 |
164
165
166
163:1 (Top Level) ↗
```

```
> seq_vec02[5]
[1] 13
> seq_vec02[length(seq_vec02)-4]
[1] 88
> seq_vec02[30]
[1] 88
```

## #인덱싱 조건식 예제

```

165 # 인덱스에서 조건식을 활용할 수 있다..
166 # AND = & , OR = "|"
167 |
168 ##인덱스 번지가 30 이하인 데이터만출력하려면?
169 seq_vec02[seq_vec02<30]
170
171
172
173 ##인덱스 번지가 10이상이고 30이하인 데이터만출력하려면?
174 seq_vec02[seq_vec02>10 & seq_vec02<30]
175
176 ##인덱스 번지가 10이상 이거나 30 이하인 데이터만출력하려면?
177 seq_vec02[seq_vec02>10 | seq_vec02 <30]
178
179 ##인덱스 번지가 홀수인 데이터만 출력하려면?
180 seq_vec02_odd <- seq_vec02[seq(1,length(seq_vec02),2)]
181 seq_vec02_odd
182

```

```

> seq_vec02[seq_vec02>10 | seq_vec02 <30]
[1] 1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100
> seq_vec02[seq_vec02<10 & seq_vec02<30]
[1] 1 4 7
> seq_vec02[seq_vec02>10 & seq_vec02>30]
[1] 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100
> seq_vec02[seq_vec02>10 | seq_vec02 <30]
[1] 1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100
> seq_vec02_odd <- seq_vec02[seq(1,length(seq_vec02),2)]
> seq_vec02_odd
[1] 1 7 13 19 25 31 37 43 49 55 61 67 73 79 85 91 97
>

```

## #round 함수 -반올림

```

184 #round(vlaue,value(반올림할 위치) - 반올림 하는 함수
185 round_vc <- c(10.234,11.3467)
186 round(round_vc,2)
187
188 round_vc02 <- 123.234
189 round(round_vc02,-1)|
190

```

## #names 함수 -- 칼럼 이름 할당

```

192
193 #names - 각각의 셀의 컬럼이름을 할당할 수 있다.
194
195 data_x <- c(1,2,3)
196 cols <-c('lim','park','cho')
197
198 names(data_x) <- cols
199 data_x
200
201 names(data_x)
202 names(data_x)[2]
203
204 data_x["lim"]
205
206 data_x[c("lim","park")]
207
208

```

## #Vector Indexing

```

210 ##Vector Indexing(인덱스는 1)
211 #벡터내의 데이터 접근 방법
212
213 index_vec <- c(1, 3, 5, 7, 9)
214
215 index_vec[2]
216
217 index_vec
218
219 index_vec[5:3] #리버스 효과
220
221 index_vec[length(index_vec):3] # 길이를통해 마지막 값을 가져올수있다.
222
223 index_vec[c(1,3)] # 내가원하는 인덱스 번지만 취할 수 있다.
224

```

205:1 (Top Level) ▾

Console Terminal × Jobs ×

C:/success/R/ ↗

```

[1] 9 7 5
> index_vec[c(1,3)] # 내가원하는 인덱스 번지만 취할 수 있다다
[1] 1 5
> ##Vector Indexing(인덱스는 1)
> #벡터내의 데이터 접근 방법
>
> index_vec <- c(1, 3, 5, 7, 9)
>
> index_vec[2]
[1] 3
>
> index_vec
[1] 1 3 5 7 9
>
> index_vec[5:3] #리버스 효과
[1] 9 7 5
>
> index_vec[length(index_vec):3] # 길이를통해 마지막 값을 가져올수있다.
[1] 9 7 5
>
> index_vec[c(1,3)] # 내가원하는 인덱스 번지만 취할 수 있다.
[1] 1 5
>

```

## #특정 요소 제외

```
225
226 # 특정요소 제외
227 index_vec[-1] #첫번째 값이 없어진다.
228 index_vec[c(-1,-5)] # 하나의 이상의 요소값을 뺄때는 vector로 묶어줘야 한다.
229
230
231
```

```
> index_vec[-1] #첫번째 값이 없어진다.
[1] 3 5 7 9
> index_vec[c(-1,-5)] # 하나의 이상의 요소값을 뺄때는 vector로 묶어줘야 한다.
[1] 3 5 7
> |
```

## #length, nrow, NROW (길이 반환 함수)

```
231
232 #길이
233 length(index_vec) #일반적인 길이를 반환해주는 함수
234 nrow(index_vec) #프레임형식에서 행의 갯수를 리턴해주는 함수 현재는 행이 없어서 안나온다.
235 NROW(index_vec) #이같은 경우에는 셀을 행으로 변환해서 리턴해준다. 그래서 값이 나온다.
236
```

```
> #길이
> length(index_vec)
[1] 5
> nrow(index_vec)
NULL
> NROW(index_vec)
[1] 5
> |
```

## # %in% 연산자(해당 값이 있는지를 확인하고 true/false를 반환한다.)

```
239 # %in% 연산자 - 해당 값이 있는지 true/false로 출력해준다.
240
241 bool <- "a" %in% c("A","b","c")
242 bool
243
244 bool <- "A" %in% c("A","b","c")
245 bool
246
```



```

> bool ← "a" %in% c("A","b","c")
> bool
[1] FALSE
> bool ← "A" %in% c("A","b","c")
> bool
[1] TRUE

```

## #setdiff, union, intersect (집합 간의 비교)

```

248 #setdiff() - 차집합, union() - 합집합, intersect() - 교집합
249
250 setdiff( c("a","b","c"), c("a","b")) # 차집합
251
252 union(c("a","b","c"),c("a","b")) #합집합
253
254 intersect(c("a","b","c"),c("a","b")) #교집합
255

```

```

> setdiff( c("a","b","c"), c("a","b"))
[1] "c"
> union(c("a","b","c"),c("a","b"))
[1] "a" "b" "c"
> intersect(c("a","b","c"),c("a","b"))
[1] "a" "b"

```

## #setequal(집합 간의 비교)

```

256
257 #집합간의 비교 setequal()
258
259 setequal(c("a","b","c"), c("a","b"))
260 setequal(c("a","b","c"), c("a","b","c"))
261

```

## #vector 예제

```
263 ##벡터 예제
264 #100에서 200으로 구성된 벡터 sampleVec를 생성한다음
265 #각 문제를 수행하는 코드를 작성하고 답을 구하시오!!
266
267 sampleVec <- c(100:200)
268 sampleVec
269
270 #문1) 10번째 값을 출력하세요
271 sampleVec[10]
272
273 #문2) 끝에서 10개의 값을 잘라내어 출력하세요
274 sampleVec[length(sampleVec):10]
275
276 #문3) 홀수만 출력하세요
277 sampleVec[seq(2,length(sampleVec),2)]
278
279 #문4) 3의 배수만 출력하세요
280 sampleVec[seq(3,length(sampleVec),3)]
281
282 #문5) 앞에서 20개의 값을 잘라내고 sampleVec.head 변수에 저장하고 출력하세요
283
284 sampleVec.head <- head(sampleVec,20)
285
286 #문6) sampleVec.head 변수에서 5번째 값을 제외하고 출력
287
288 sampleVec.head[c(-5)]
289
290 #문7) sampleVec.head 변수에서 5,7,9번째 값을 제외하고 출력
291 sampleVec.head[c(-5,-7,-9)]
292
```

```
> sampleVec <- c(100:200)
> sampleVec
[1] 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134
[35] 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169
[71] 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
>
> #문1) 10번째 값을 출력하세요
> sampleVec[10]
[1] 109
>
> #문2) 끝에서 10개의 값을 잘라내어 출력하세요
> sampleVec[length(sampleVec):10]
[1] 200 199 198 197 196 195 194 193 192 191 190 189 188 187 186 185 184 183 182 181 180 179 178 177 176 175 174 173 172 171 170 169 168 167 166
[36] 165 164 163 162 161 160 159 158 157 156 155 154 153 152 151 150 149 148 147 146 145 144 143 142 141 140 139 138 137 136 135 134 133 132 131
[71] 130 129 128 127 126 125 124 123 122 121 120 119 118 117 116 115 114 113 112 111 110 109
>
> #문3) 홀수만 출력하세요
> sampleVec[seq(2,length(sampleVec),2)]
[1] 101 103 105 107 109 111 113 115 117 119 121 123 125 127 129 131 133 135 137 139 141 143 145 147 149 151 153 155 157 159 161 163 165 167 169
[96] 171 173 175 177 179 181 183 185 187 189 191 193 195 197 199
>
```

```
> #문4) 3의 배수만 출력하세요
> sampleVec[seq(3,length(sampleVec),3)]
[1] 102 105 108 111 114 117 120 123 126 129 132 135 138 141 144 147 150 153 156 159 162 165 168 171 174 177 180 183 186 189 192 195 198
>
> #문5) 앞에서 20개의 값을 잘라내고 sampleVec.head 변수에 저장하고 출력하세요
> sampleVec.head <- head(sampleVec,20)
>
> #문6) sampleVec.head 변수에서 5번째 값을 제외하고 출력
> sampleVec.head[c(-5)]
[1] 100 101 102 103 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
>
> #문7) sampleVec.head 변수에서 5,7,9번째 값을 제외하고 출력
> sampleVec.head[c(-5,-7,-9)]
[1] 100 101 102 103 105 107 109 110 111 112 113 114 115 116 117 118 119
>
```

```

294
295 ##월별 결석생 수 통계가 다음과 같을때
296 #이 자료를 absent 벡터에 저장하시오
297 #결석생 수를 값으로하고 , 월 이름을 값의 이름으로 한다.)
298
299 ?month.name
300 absent ← c(10,8,14,15,9,10,15,12,9,7,8,7)
301
302 names(absent) ← month.name
303
304 #문1) 5월(MAY)의 결석생 수를 출력하시오
305
306 absent[5]
307
308 #문2) 7월(JUL), 9월(SEP)의 결석생 수를 출력하시오
309
310 absent[c(7,9)]
311
312 #문3) 상반기 (1~6월)의 결석생 수의 합계를 출력하시오
313 sum(absent[c(1:6)])
314
315 #문4) 하반기(7~12월)의 결석생 수의 평균을 출력하시오
316 mean(absent[c(7:12)])
317

```

288:23 (Top Level) ↕

Console

Terminal ×

Jobs ×

C:/success/R/ ↗

```

> names(absent) ← month.name
> #문1) 5월(MAY)의 결석생 수를 출력하시오
> absent[5]
May
9
> #문2) 7월(JUL), 9월(SEP)의 결석생 수를 출력하시오
> absent[c(7,9)]
July September
15          9
>
> #문3) 상반기 (1~6월)의 결석생 수의 합계를 출력하시오
> sum(absent[c(1:6)])
[1] 66
>
> #문4) 하반기(7~12월)의 결석생 수의 평균을 출력하시오
> mean(absent[c(7:12)])
[1] 9.666667

```

## #논리형 벡터 , 문자형 벡터

```

324 #논리형 벡터, 문자형 벡터
325 c(T, F, TRUE, FALSE)
326
327 c(T, F, T) | c(TRUE, TRUE, FALSE)
328
329 c(F, T) | c(TRUE, TRUE, FALSE) # 객체의 길이가 안맞아서 안된다
330
331 !c(T,F,T) # !는 역을 나타낸다.
332
333 xor(c(T, F, T), c(TRUE, TRUE, FALSE)) #xor 역을나타내서 앞에서 true일 때 false 면 true를 나타낸다.
334
335 (randomNum ← runif(3)) # runif는 난수 발생을 한다.
336
337 (0.25 < randomNum) & (randomNum < 0.75)
338
339 any(randomNum > 0.8) # 요소의 값중 하나라도 true이면 true를 반환한다.
340 all(randomNum < 0.8) # 모든 요소가 true일 때만 true를 반환한다.
341

```

```

> #논리형 벡터, 문자형 벡터
> c(T, F, TRUE, FALSE)
[1] TRUE FALSE TRUE FALSE
>
> c(T, F, T) != c(TRUE, TRUE, FALSE)
[1] TRUE TRUE TRUE
>
> c(F, T) != c(TRUE, TRUE, FALSE) # 객체의 길이가 안맞아서 안된다
[1] TRUE TRUE FALSE
경고메시지(들):
In c(F, T) != c(TRUE, TRUE, FALSE) :
  두 객체의 길이가 서로 배수관계에 있지 않습니다
>
> !c(T, F, T) # !는 역을 나타낸다.
[1] FALSE TRUE FALSE
>
> xor(c(T, F, T), c(TRUE, TRUE, FALSE)) #xor 역을나타내서 앞에서 true일 때 false 면 true를 나타낸다.
[1] FALSE TRUE TRUE
>
> (randomNum <- runif(3)) # runif는 난수 발생을 한다.
[1] 0.2465945 0.8500172 0.6784686

```

```

> (randomNum <- runif(3)) # runif는 난수 발생을 한다.
[1] 0.2465945 0.8500172 0.6784686
>
> (0.25 < randomNum) & (randomNum < 0.75)
[1] FALSE FALSE TRUE
>
> any(randomNum > 0.8) # 요소의 값중 하나라도 true이면 true를 반환한다.
[1] TRUE
> all(randomNum < 0.8) # 모든 요소가 true일 때만 true를 반환한다.
[1] FALSE

```

## #문자열 비교

```

342
343 #문자열 비교|
344 c("a","b","c","d","e")
345 strVec <- c("H","S","T","N","O")
346 strVec[1] > strVec[5]
347 strVec[3] > strVec[5]
348
349

```

## #paste - 여러 개의 문자열을 합쳐서 반환해주는 함수

```

350 #paste() - 여러개의 문자열을 합쳐서 반환해주는 함수
351 paste("May 1","help you?")
352
353 |
354 ?month.abb
355 month.abb
356
357 paste(month.abb,1:12,c("st","nd","rd",rep("th",9)))
358
359 paste("/usr","local","bin",sep="/")
360
361 paste("/usr","local","bin",sep=",")
362
363 paste("/usr","local","bin",sep=" ")
364
365 paste("/usr","local","bin",sep="")
366
367

```

```

[1] "May 1, help you?"
>
>
> ?month.abb
> month.abb
[1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
>
> paste(month.abb,1:12,c("st","nd","rd",rep("th",9)))
[1] "Jan 1 st" "Feb 2 nd" "Mar 3 rd" "Apr 4 th" "May 5 th" "Jun 6 th" "Jul 7 th" "Aug 8 th" "Sep 9 th" "Oct 10 th" "Nov 11 th"
[12] "Dec 12 th"
>
> paste("/usr","local","bin",sep="/")
[1] "/usr/local/bin"
>
> paste("/usr","local","bin",sep=",")
[1] "/usr,local,bin"
>
> paste("/usr","local","bin",sep=" ")
[1] "/usr local bin"
>
> paste("/usr","local","bin",sep="")
[1] "/usrlocalbin"
>

```

```

368 (seqVec ← paste(1:4))
369 seqVec[2]
370 class(seqVec)
371
372 paste(seqVec, collapse = "jslim")
373 paste(seqVec, collapse="")|
374

```

```

> (seqVec ← paste(1:4))
[1] "1" "2" "3" "4"
> seqVec
[1] "1" "2" "3" "4"
> seqVec[1]
[1] "1"
> class(seqVec)
[1] "character"
> paste(seqVec, collapse = "jslim")
[1] "1jslim2jslim3jslim4"
> seqVec[2]
[1] "2"
> paste(seqVec, collapse="")
[1] "1234"

```

## #str\_locate\_all - 문자열 위치

```

490
491
492 #문자열 위치
493 str_locate_all(stringLength,'섭섭')
494 class(str_locate_all(stringLength,'섭섭'))
495

```

```
C:/success/R/ ↗
> str_locate_all(stringLength, '섭섭')
[[1]]
      start end
[1,]    19  20

> class(str_locate_all(stringLength, '섭섭'))
[1] "list"
> |
```

## #특수문자 제외

```
497 #특수문자 제외 - 특수문자는 \  이거 두개와 특수문자를 붙여 쓴다.
498 num <- "$123,466"
499 tmp <- str_replace_all(num, "\\$!\\," , "")
500 class(tmp)
501
```

```
496
497 #특수문자 제외 - 특수문자는 \  이거 두개와 특수문자를 붙여 쓴다.
498 num <- "$123,466"
499 tmp <- str_replace_all(num, "\\$!\\," , "")
500 tmp
501 class(tmp)
502
```

## # as.타입 (형변환)

```
502
503 #형변환
504 #as.numeric(tmp)
505 data <- as.numeric(tmp)
506 data * 2
507
```

```
[1] "character"
> tmp
[1] "123466"
> ?as
> #형변환
> #as.numeric(tmp)
> data <- as.numeric(tmp)
> data * 2
[1] 246932
> |
```



### 'R' 카테고리의 다른 글

[R] R에서 사용되는 정규표현식(Regex) 표현 방법과 함수를 통한 사용 예제 (0) 08:37:37

[R] R - 기본적인 함수 정리(출력,인덱싱,길이반환,문자열비교 등등) (0) 00:19:35

## 태그

length함수

names함수

paste함수

print

R 기본함수

R 함수

rep함수

seq함수

setequal함수

관련글



[R] R에서 ...

댓글 0

