텍스트와 날짜

텍스트

- 문자 벡터의 원소들
- 한 개의 문자도 될수 있고, 여러 개의 문자로 구성 될 수도 있다
- nchar() : 텍스트가 몇 개의 문자로 구성돼있는지 확인
    - 공백도 문자로 취급함
    - length() 함수는 주어진 문자 벡터의 길이를 반환함
- 정렬 : sort() 함수
    - 내림차순 정렬 : 옵션 decreasing = TRUE
- 텍스트에 포함된 문자를 소문자 또는 대문자로 일괄 변경
    - 대문자 : toupper(), 소문자 : tolower()
- 텍스트 분할 / 결합
    - strsplit() : 분할 - 첫번째 인수에 텍스트, 두번째 인수에 구분자
        - 그냥 빈 따옴표를 구분자로 하면 글자 단위로 분할
        - **리스트** 형식으로 출력(각 원소는 분할된 텍스트로 구성된 벡터 -> unlist()하거나 인덱싱을 통해 벡터로 변환하면 벡터형식으로 추출
        - 텍스트 타입의 벡터를 split 하게되면, 각 텍스트별로 원소가 분할된다
    - paste() : 결합
        - 인수로 주어진 텍스트 벡터들을 차례대로 결합해 하나의 새로운 텍스트를 생성
        - 실수로 텍스트들을 c()로 묶으면 하나의 텍스트 벡터로 인식하기 때문에 무용지물
        - 텍스트 결합 시 텍스트 간에 한 개의 공백을 삽입 -> 공백말고 다른걸 삽입하고 싶으면 sep 인수에 원하는 구분자를 입력
        - 결합해야 할 데이터가 텍스트가 아니면 내부적으로 이들을 문자형의 텍스트로 변환 -> 비문자형 데이터나 수식의 실행 결과도 텍스트 형식으로 결합가능
        - 인수로 벡터 여러개를 받을 수 있으며, 이 상황에서는 각각의 대응되는 인덱스에 있는 텍스트끼리 결합해서 반환, 재사용 법칙도 적용
            - heroes <- c("Batman", "Captain America", "Hulk") paste(heroes, "wants", "to", "fly") -> [1] "Batman wants to fly" "Captain America wants to fly" [3] "Hulk wants to fly"
        - collapse 인수 : 텍스트 결합을 통해 생성된 텍스트들을 다시 하나로 연결하는 구분자 -> 반환되는 벡터 내의 모든 원소들이 합쳐서 하나의 값으로 출력되길 원할 때 사용 --> 공백으로 지정하면 띄어쓰기 하듯이 연결됨
            - sep 인수와 동시에 쓰이면, 먼저 sep에 따라서 결합이 된 후, 이후 collapse에 따라서 하나의 텍스트로 변환이 됨
        - 변수와 함께 다소 긴 텍스트를 결합할때는 불편하고, 가독성도 떨어짐
    - outer() 함수: paste와 함께 사용하면 두 개의 텍스트 집합으로부터 가능한 모든 텍스트 쌍의 조합 반환
        - 원래는 두 집합에 대해 가능한 모든 순서쌍의 곱을 수행
        - 세번째 인수인 FUN에 함수를 지정해 곱셈 대신 다른 작업을 수행 할 수 있음

        - 네번째 인수는 세번째에 지정한 함수의 인수를 입력
        - 결과를 행렬 형식으로 출력, 벡터 형식이 필요하면 as.vector() 함수를 이용
        - 순서를 고려하지 않은 중복되지 않는 텍스트 조합만을 보고싶으면 대각선 아래부분을 제거
            - lower.tri() 함수는 대각선 아래부분을 추출 -> 이에 대한 **반대 집합을** 구하면 됨
    - sprintf() 함수 : 텍스트 내의 %s가 각 변수에 대응, 순서가 일치해야 함 - %s는 텍스트 형식 포맷 출력
        - printf 같은 느낌
        - 변수가 벡터로 주어지면, 벡터연산 방식으로 벡터 내 원소가 처리됨
            - 벡터의 길이는 다른 벡터 길이의 배수여야 함
        - %s 이외에도 %f, %e, %d도 지원
- 중복되지 않는 단어 추출
    - unique() 함수 이용
        - 적용하기전에 tolower() / toupper()로 대문자/소문자를 통합해주는게 좋음
- 텍스트로부터 특정 위치에 있는 문자열 추출 : substr() / substring() 함수
    - substr() : start, stop 위치를 인수로 줌
        - 시작 위치, 끝 위치를 벡터로 지정 할 수 있음
    - substring() : start 위치만 인수로 줌(종료 위치 인수 생략 가능)
    - 텍스트 벡터가 인수러 주어지면 벡터 연산 적용, 각 원소별로 수행, 이 값들의 벡토로 반환
- 특정 문자열을 포함하고 있는 텍스트 찾기 : grep() 함수
    - 첫번째 인수 pattern에서 찾고자 하는 문자열 지정
    - 두번째 인수로 지정된 텍스트 벡터에서 해당 문자열을 포함하고 있는 원소의 위치 반환
    - value = TRUE를 지정해주면 인덱스 필요없이 결과를 바로 알려줌
    - 두 개 이상의 단어로 이루어진 텍스트를 찾을려면 공백이 포함된 문자열을 찾으면 됨
- 텍스트 내의 특정 문자열 찾아서 다른 문자열을 바꾸려면 sub() / gsub()
    - sub() : 일치하는 **처음 문자열**만 새로운 문자열로 바꿈
    - gsub() : 일치하는 **모든 문자열**을 새로운 문자열로 대체
        - 첫번째 인수(pattern)에는 찾고자 하는 문자열 지정
        - 두번째 인수(replacement)에는 대체하고자 하는 문자열 지정
        - 세번째 인수(x) 대상이 되는 전체 텍스트 지정
        - 빈 텍스트로 대체하는 작업으로 문자열을 삭제하는 효과를 얻을 수 있습니다
    - fixed =TRUE 옵션이면 문자열 그대로 취급, FALSE(기본값)면 정규표현식으로 해석
        - strsplit, grep, sub, gsub 등 다 사용가능
    - 정규표현식
        - 대괄호 [] : 각 문자와 하나라도 일치하는 문자열을 검색
        - | : OR조건을 만족하는 패턴 찾기
        - 소괄호 () : 패턴을 그룹화(분리)
        - ? : ?앞의 문자가 0또는 1회 나타나는 패턴(최대 1회)
        - * : * 앞의 문자가 0또는 1회 이상 반복되는 패턴(최소 0회)
        - + : + 앞의 문자가 1회 이상 반복되는 패턴 (최소 1회)
        - ^ : 뒤에 나오는 문자로 시작하는 문자열
        - $: 앞에 나오는 문자로 끝나는 문자열
        - . : 임의의 문자 하나
        - [[:alnum:]] : 알파벳과 숫자 조합

        - [[:alpha:]] : 알파벳
        - [[:digit:]] : 숫자
        - [[:punct:]] : 문장부호 및 특수문자
        - [[:space:]] : 공백
        - \w, \d, \s는 각각 단어, 숫자, 공백을 나타냄

날짜

- 현재 날짜 : Sys.Date()
    - R은 이 객체를 텍스트 형태로 변환해 출력
- date() : 현재의 날짜와 시간
- 현재의 시스템 시간 : Sys.time()
- 텍스트 데이터로부터 날짜를 생성하는 가장 간단한 방법 : as.Date()
    - 첫번째 인수로 주어진 텍스트
    - 두번째 인수로 주어진 날짜 형식(default = yyyy-mm-dd)
        - 아니라면 따로 format인수에 지정
        - %m : 두자리 숫자로 표현한 월, %d : 일, %y : 네자리로 표현한 연도
    - 이 방식으로 객체를 생성하면, 날짜 정보는 나타낼 수 있지만 시간을 표현하지는 못함
- format() 함수 : 날짜를 다양한 형식으로 출력
    - 인수를 받아서 지정한 형식으로 출력
    - 첫번째 인수 Date 객체
    - 두번째 인수 format
- 어떤 날짜의 요일 : weekdays() 함수
    - 첫번째 인수 Date 객체
    - 날짜 벡터를 인수로 받을 수 있으며 각 날짜에 대응되는 요일이 벡터 형식으로 출력
- 날짜 객체에 더하기나 빼기를 해서 새로운 날짜 객체 생성 가능
- 연속 된 날짜 데이터 생성 : seq() 함수
    - 연속된 숫자를 생성하는 것과 동일한 방식으로 연속된 날짜 데이터 생성
    - from, to, by를 지정해줌 - 여기서 by 1은 하루를 의미
    - length.out도 지정가능(일수)
    - by는 day(s), week(s), month(s)로 세부지정 가능
- months() 함수, quaters() 함수 : 주어진 날짜에 대한 월과 분기를 출력
- 날짜 관련 함수들의 출력 결과는 현재 사용중인 컴퓨터의 로케일 설정에 따름
    - 한글 윈도우를 사용할 때는 한글형식으로 날짜를 출력함
        - sys.getlocale()에서 확인
- 날짜와 함께 추가적으로 시간 정보를 저장하려면 as.POSIXct() 함수 / as.POSIXlt()
    - POSIXct() : 1970년 1월 1일 이후 경과된 초의 개수를 숫자 벡터 형식으로 저장
        - 메모리를 덜 필요함 -> 효율적
        - as.integer() 함수를 적용하면 정수로 변환가능
        - format 인수에서 %H, %M, %S도 사용가능
    - POSIXlt() : 년,월,일,요일,시간,분,초 등의 날짜 및 시간 요소를 리스트로 저장
        - 시간요소를 조금 더 쉽게 추출가능

        - $기호와 요소 이름을 사용해 정보 추출 : sec, min, hour, mday, mon, year, wday, yday
        - 연도를 추출할때는 반드시 1900을 더해야 정상값이 나옴
        - 월을 추출할때는 반드시 1을 더해야 정확한 월이 나옴
    - 얘네는 초단위로 날짜정보가 저장되기 때문에 산술연산도 초단위임
        - 몇 시간, 몇 주를 수동으로 계산해서 구할수있음
    - strptime() : 텍스트로 주어진 날짜와 시간을 POSIXlt객체로 나타낼 수 있음
        - 변환된 결과는 POSIXlt 객체와 동일하게 취급
- ISOdate() 함수 : 인수로 주어진 세 데이터(년,월,일)을 합쳐서 하나의 POSIXct 객체를 생성
    - 시간 정보가 필요없을 때는 그냥 바로 as.Date()로 시간정보 제거
    - 벡터형식으로 저장된 년월일 데이터를 한꺼번에 날짜 데이터로 변환 가능
        - 날짜 데이터를 대량으로 생성해야 할때 매우 유용
- 1970년 이전의 날짜는 음수로 저장, 1970년 1월 1일 이후 날짜는 경과된 일수로 저장 -> 줄리안 날짜
    - julian() 함수를 이용해서 줄리안 날짜 구하기 가능
        - 인수 = Date객체
- difftime() 함수 : 좀 더 다양한 방식으로 기간 간격 계산
    - 초, 분, 시, 일, 주 단위로 기간 계산 가능
        - units = "단위" 인수로 가능
    - 첫번째와 두번째 인수가 비교할 날짜들
- 날짜 객체간에 비교 연산자를 이용해서 선후 관계도 파악 가능(부등호)