* 배열
* 범위 배열 생성
  + range(1,10) : 1부터 10까지 수를 속성으로 가진 배열 생성
* 정렬(1차원 배열)
  + sort() : 순서대로 정렬(오름차순) - 숫자 인덱스인 배열에서 사용
  + natsort() : 자연 정렬 순서로(오름차순) 정렬
  + asort() : 값을 기준으로 정렬 - 숫자가 아닌 인덱스인 배열에서 사용
  + ksort() : 키를 기준으로 정렬 - 숫자가 아닌 인덱스인 배열에서 사용
  + rsort() : 내림 차순으로 정렬
  + arsort() : 값을 기준으로 내림 차순으로 정렬
  + krsort() : 키를 기준으로 내림 차순으로 정렬
* 다차원 배열
  + array\_multisort() : (오름차순 - SORT\_ASC, 내림차순 - SORT\_DESC) 키가 문자열일 때는 이상없이 정렬, 키가 숫자일 때는 인덱스가 다시 생성됨.
* 사용자 정렬
  + usort() : 비교 함수(compare($x, $y))를 필요로 한다.
* 무작위 정렬
  + suffle()
* 역순으로 정렬
  + array\_reverse()
* 파일에서 내용을 배열로 로드
  + file(파일 경로)
* 문자열에서 숫자를 추출하는 방법
  + intval(문자열로 된 숫자만 변환됨.)
* 배열 순환 함수
  + each(), current(), reset(), end(), next(), pos(), prev()
* 배열의 각 요소에 함수 적용하기
  + array\_walk(배열, 사용자 정의 함수, 배열 각 요소에 적용할 데이터) : 배열의 모든 요소를 동일한 방법으로 일괄처리할 때 사용
  + 사용자 정의함수 : my\_function(배열의 값, 배열의 키, 배열 각 요소에 적용할 데이터) 이런식으로 구성한다. 데이터를 사용하기 위해서는 키가 필요가 없더라도 지정해야 한다. 배열의 값에 &를 붙여서 전달하면(예 : &$value) 변수의 값이 아닌 참조가 전달이 되므로 배열의 값을 변경할 수 있다.
* 배열의 요소 수 알아내기
  + count(),
  + sizeof() : count()와 동일, 빈 배열이나 없는 배열이 인자로 전달되면 0을 반환함.
  + array\_count\_values() : 배열에 포함된 고유한 값들이 키, 그 값들의 개수를 값으로 가진 배열을 반환한다.
* 배열을 스칼라 변수로 추출하기
  + extract() : 배열의 키가 변수 이름, 배열의 값이 변수의 값이 되는 변수들을 생성한다. 이미 존재하는 변수명일 경우 어떻게 할 것인지 지정하는 것은 두번째(extract type), 세번째(prefix) 파라미터에서 할 수 있다.
* 문자열 처리와 정규 표현식
  + 문자열 다듬기
    - chop() : rtrim()의 별칭
    - ltrim() / rtrim() : 문자열의 시작/끝부분의 공백문자를 제거
    - trim() : 캐리지 리턴(\n, \r), 수평 탭(\t), 수직 탭(\x0B), 문자열의 끝(\0), 스페이스 제거
  + 문자열 출력의 정형화
    - htmlspecialchars() : 브라우저 출력 문자열 필터링, html에서 특별한 의미를 갖는 문자가 이상없이 출력되도록 그것과 대응되는 html 요소로 변환 (예 : < 를 &lt;로 변환)
    - str\_replace() : 간단하게 하나의 문자열 전체를 다른 것으로 변경하는 경우 사용
    - nl2br() : 모든 줄바꿈 문자(\n)를 html의 <br/>로 변경한다.
      * 사용 예 : <?php echo nl2br(htmlspecialchars($str)) ?>
  + 출력 문자열의 형식 만들기
    - printf() : 정형화된 문자열을 출력
    - sprintf() : 정형화된 문자열을 반환
    - vprintf(), vsprintf() : 두번째 인자에 배열로 전달
    - strtoupper() : 문자열을 대문자로 변경
    - strtolower() : 문자열을 소문자로 변경
    - ucfirst() : 문자열의 첫 글 자가 알파벳이면 대문자로 변경
    - ucwords() : 문자열의 각 단어의 첫 글자가 알파벳이면 대문자로 변경
  + 문자열 함수를 사용해서 문자열을 결합하고 분리하기
    - explode()
    - implode() == join()
    - strtok(문자열, 구분자) : 문자열로부터 한 번에 하나의 조각만 가져온다.  
      첫번째 토큰 : $token = strtok($str, “ “);  
      다음 토큰 : $token = strtok(“ “);  
      포인터 초기화 : $token = strtok($str, “ “);
    - substr() : 문자열에 지정된 시작과 끝 위치 사이의 부분 문자열을 추출
  + 문자열 비교하기(문자열 순서 결정하기)
    - strcmp(str1, str2) : 같으면 0, str1 > str2이면 양수 아니면 음수 – 대소문자 구별
    - strcasecmp(str1, str2) : strcmp()와 동일하지만 대소문자 구별 X
    - strnatcmp() / strnatcasecmp() : 자연적 순서(문자열 “2” < 문자열 “12”)에 따라 문자열 비교
  + 문자열 길이 검사
    - strlen()
      * 이메일의 적법성 검사할 때 간단하게 쓸 수 있다. 이메일은 최소 6글자 이상
  + 부분 문자열의 일치 검사 및 변경
  + 문자열에 포함된 문자열 찾기
    - strstr(검색대상 문자열, 검색할 문자열) : 긴 문자열에 특정 문자열이나 문자가 일치하는 것이 있는지 찾음.  
      일치하는 문자열이 있으면 그 문자열부터 끝까지의 문자열을 반환함. 없으면 false를 반환함.
    - strchr() : strstr()과 동일하지만 문자열에 포함된 문자 하나를 찾는데 사용됨.
    - strrchr() : strstr()과 유사하지만 검색할 문자열과 마지막으로 일치하는 문자열부터 검색대상 문자열 끝까지의 문자열을 반환한다.   
      또 하나의 문자만 검색할 수도 있다. (검색할 문자열로 전달된 문자열의 첫번째 문자)
    - stristr() : strstr()과 동일하지만 대소문자를 구별하지 않음.
  + 문자열 위치 찾기
    - strpos(검색대상 문자열, 검색할 문자열, 오프셋=0) : strstr()과 유사하게 동작. 부분 문자열 반환대신 검색대상 문자열에 포함된 검색할 문자열이 발견된 첫번째 위치를 숫자로 반환한다.   
      strstr()보다 실행속도가 더 빠르기 때문에 strops()를 더 권장.   
      오프셋은 검색을 시작할 위치를 지정.   
      검색결과가 없으면 false를 반환.   
      위치 0과 false는 혼란을 야기할 수 있으므로 비교할 때 === 연산자를 사용.
    - strrpos() : strpos()와 거의 같지만 검색대상 문자열에서 마지막으로 발견된 검색할 문자열의 위치를 반환.
  + 문자열 변경하기
    - str\_replace(needle, new\_needle, haystack) : haystack에 나타나는 모든 needle 문자열을 찾아서 new\_needle로 변경하고 새로운 버전의 haystack 문자열을 반환한다.
    - substr\_replace(string, replacement, start[, length]) : 문자열안에서의 위치를 기준으로 특정의 부분 문자열을 찾아 변경한다.   
      string 문자열의 일부분을 replacement 문자열로 변경한다. 변경될 부분은 start 인자의 값과 length 인자(생략가능)의 값에 따라 결정된다.   
      length를 0으로 지정하면 기존 문자열을 덮어쓰지 않고 start에 지정된 위치에 replacement 문자열이 삽입된다.   
      양수면 변경해야 하는 문자 개수를 나타낸다.   
      음수면 문자 변경이 끝낼 지점을 나타내며, 문자열의 끝으로부터 그 값만큼 떨어진 곳을 나타낸다.
* 정규 표현식
  + - PHP 5.3 버전부터는 POSIX 스타일 사용이 안됨, Perl 방식만 사용됨.
  + 정규 표현식 함수로 부분 문자열 찾기
    - int preg\_match( pattern, subject[, matches[, flags=0[, offset=0]]] )  
      subject 문자열을 검색하여 pattern의 정규 표현식과 일치하는 것을 찾는다. 만일 일치하는 것이 발견되면, 검색 결과가 $matches 배열에 저장된다. $matches[0]에는 패턴 전체에 대해 일치되는 문자열이 저장된다. 그 다음의 배열 요소들은 패턴의 각 부분 표현식에 일치되는 문자열을 저장한다.  
      일치 된 것이 발견되면 1, 발견 안되면 0, 에러가 생기면 false를 반환한다.  
      0과 false 혼동을 막기 위해 === 연산자를 사용해야 한다.
  + 정규 표현식 함수로 부분 문자열 변경하기
    - mixed preg\_replace(pattern, replacement, subject[, limit=-1[, &count]])  
      subject 문자열에서 pattern을 검사한 후 일치되면 replacement 문자열로 변경한다.  
      limit 매개변수에는 최대 변경 횟수를 지정한다. 기본 값은 무한정의 의미를 갖는 -1  
      count 매개변수를 지정하면 전체 변경 횟수가 해당 변수에 저장된다.
  + 정규 표현식 함수로 문자열 분리하기
    - array preg\_split(pattern, subject[, limit=-1[, flags=0]])  
      pattern과 일치되는 부분 문자열로 subject 문자열을 분리한 후 각 부분 문자열을 배열에 넣어서 반환한다.
* 코드 재사용과 함수 작성
  + require()와 include()
    - 동일한 기능, 실행 실패시  
      require() : 중대한 에러를 발생시켜서 스크립트의 실행을 중단시킴.  
      include() : 경고만을 발생시키고 스크립트 실행은 계속하도록 해줌.
  + require\_once()와 include\_once()
    - 가져오는 파일이 한 번만 포함되도록 해준다.  
      잘못해서 같은 함수 라이브러리를 두 번 포함시키는 바람에 그 안에 있는 함수가 중복 정의되어 에러를 유발시키는 것을 막아준다.   
      하지만 실행속도는 require(), include()가 더 빠르다.
  + readfile()
    - 포함하는 파일에 php문이 없고 일반 텍스트나 html만 있을 때는 require() 대신 사용.  
      파일의 내용을 파싱하지 않고 포함만 시켜준다.