| 문항번호 | | 1 | 난이도 | 하 |
| --- | --- | --- | --- | --- |
| 유형 | | 구현 | 예상 소요 시간 | 30분 |
| 문제 | 문자열 S는 숫자와 4칙연산 기호로 이루어진 수식입니다.  해당 수식을 4칙연산 계산 순서에 맞춰 계산하는 함수를 작성하세요.  단, 소수점 자리는 2번째 자리까지 표현합니다.  입력  사칙 연산이 가능한 String 문자열  출력  소수점 2자리까지 표현된 사칙 연산 결과값  예를들어, S가 “2\*3+5/6\*3+15” 인 경우 결과는 23.50 입니다. | | | |
| 매개변수  &  리턴타입 | 매개변수 : String S  리턴타입 : float | | | |
| 지원자  초기 코드 | * python   def solution(S):  answer = 0  return answer;   * javascript   function solution(S) {  var answer = 0;  return answer;  } | | | |
| 모범  답안 | * python   import re  def solution(S):  arr = [ s for s in S ]  numbers = re.findall(r'\d+', S)  operators = re.findall(r'[+\-\*/]', S)  temp = float(numbers[0])  result = 0  for i in range (len(operators)):  if operators[i] == '\*':  temp \*= float(numbers[i+1])  elif operators[i] == '/':  temp /= float(numbers[i+1])  elif operators[i] == '+':  result += temp  temp = 0  temp += float(numbers[i+1])  elif operators[i] == '-':  result += temp  temp = 0  temp -= float(numbers[i+1])  result += temp  return '%0.2f' % result   * javascript   function solution(S){  var arr = S.split("");  var numbers = S.match(/\d+/gi);  var operators = S.match(/[+\-\*/]/gi);  var temp = parseFloat(numbers[0]);  var result = 0;  for(var i = 0; i < operators.length; i++){  switch (operators[i]) {  case '\*':  temp \*= parseFloat(numbers[i+1]);  break;  case '/':  temp /= parseFloat(numbers[i+1]);  break;  case '+':  result += temp;  temp = 0;  temp += parseFloat(numbers[i+1]);  break;  case '-':  result += temp;  temp = 0;  temp -= parseFloat(numbers[i+1]);  break;  }  }  result += temp;  return result.toFixed(2);  } | | | |
| 테스트  케이스 | 입력 S : “2-6-7\*8/2+5”  출력 : -27.00  입력 S : “2\*3+5/6\*3+15”  출력 : 23.50  입력 S : “1+2\*3/4-5”  출력 : -2.50  입력 S : “1\*2\*3\*⅘”  출력 : 4.80  입력 S : “3-2\*5-11”  출력 : -18.00 | | | |