HW2

휴먼컴퓨터인터페이스

#2. 대화형 프로토타입

제출 : 2018년 04월 29일

소속 : 컴퓨터소프트웨어학과

학번 : 2014726049

성명 : 신 숙

1. 개요

(1) 기능적 요구조건 구현 완성도

항목	여부	비고
1. 수식 입력		
(1) 정수, 실수, 복소수의 표현과 그 기본연산	O	
(2) 벡터, 행렬의 표현과 그 기본 연산	O	
(3) 자주 사용되는 상수 및 함수 지원	O	
2. 결과 출력		
(1) 올바른 입력 => 수식의 결과 값	O	
(2) 잘못된 입력 => 오류 메시지	O	
3. 변수, 함수 정의 및 사용		
(1) 변수 : 최소 3개(예. x,y,z)	O	
(2) 함수 : 최소 2개(예. f, g)	O	

(2) 오픈소스 라이브러리 의존성 요약

기존에 주어진 calc_v2를 이용하여 외부 디자인과 내부 동작방식만을 바꾼 것으로, 외부 라이브러리는 'math.js'만을 사용한다.

(3) 사용성 향상에 기여하는 핵심적인 상호작용 방식 요약

사용성 향상을 위하여 직접 계산기를 사용해 보았을 때, 계산기 화면과 함수 버튼들을 제일 많이 보는데 그 중 특히 함수 버튼을 제일 많이 본다는 것을 알았고, 그 이유가 본인이 원하는 버튼이 어디 있는지 알아내서 써야하기 때문이라는 것을 알았다.

그래서 계산기를 ***직관적***이고, 매우 ***간단하며*** 시선을 주로 한 곳(결과창)에 집중시키고 싶어서, 크게 함수들을 나눠서 사용 대상자가 보기 편할 '한글'로 메뉴를 보이며 함수들을 바로 볼 수 있게 한다.

또한, 함수들의 목록들이 중복으로 켜지는 일이 없도록 하기 위해, 'Close'라는 버튼을 만들어서 실수로 잘못 눌렀다면 끌 수 있도록 하였고, 마지막 글씨를 지우는 'BackSpace'기능을 'Back'이라는 버튼에 추가하였다. 둘은 매우 중요한 기능이기 때문에 크기를 좀 더 키우고 색으로 하이라이트를 주었다.

EV 기능도 AC와 같이 위로 올릴까 생각을 해봤지만, 사용해보고 보았던 거의 대다수의 계산기들이 오른쪽 하단부분에 계산하는 버튼을 두었던 것이 너무 익숙하였다. 그래서 오른쪽 하단에 그대로 두는 대신에 색을 주었고. AC만 Close와 Back 사이에 두도록 하였다. 세 가지 모두 지우는 기능이기에 통일성이 있어 보였다.

2. 본문

(1) 기본 계획과 변경된 부분

기본에는 지금보다 다양하게 여러 함수들을 더 추가하고 함수 각각에 대한 사용 예제들도 출력하려

고 하였으나, 시험기간 등 여러 가지가 한 번에 겹치다 보니 원하는 모든 것을 2번째 과제 안에 구현하기에는 시간이 부족하였다. 더욱이 인터넷프로그래밍 언어와 jquery에 미숙하여서 익히는 것도 힘들었다. 그래서 일단은 기능적 최소 요구조건을 위주로 갖추게 하였으며, 그로 인해 형태도 변하게 되었다.

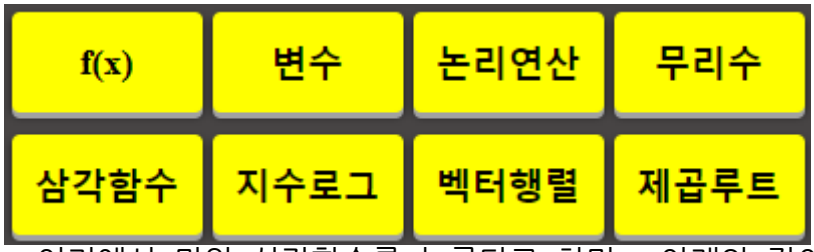
구상 단계에서는 크게 크게 직관적으로 만들려고 했었으나, 실제로 구현을 해 본 결과, 너무 커도 보기 싫고, 빈 공간이 많아도 좋게 보이지 않는다는 것을 깨달았고, 행렬식과 벡터를 구현하는 (,), [,]를 일반 계산 수식들과 같이 함수 내부가 아닌, calc 내부에 표현해 주었다. 그래서 전체적인 외형이 조금 더 많아지고, 앞으로 함수를 더 추가할 것이기 때문에, 어떤 버튼을 누른다면 노란 칸의 함수들의 집합이 다른 것으로 바뀌어 사용가능하게 할 예정이다.

또한, 사용자가 사용할 함수를 클릭했을 때, 함수를 어떻게 그리고 어떤 순서로 써야하는지 예제가 담긴 이미지(혹은 수식)를 보여주고 싶었지만 아직 이미지들과 그 이미지를 어디에 출력시킬지를 정하지 못하여서 기능을 추가하지 못하였다. 이것을 수행하기 위해서는 모든 기능에 대한 이미지를 준비해야하고, 전체에 대한 id를 잡아서 구현해야 하므로 구상 중에 있다.

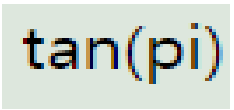
(2) 사용자 인터페이스의 구성 요소 및 사용 방법

기본 수식들(*, /, +, - 등)의 사용방법은 본래 계산기와 같이 수와 수식(ex. 8 * 5)을 누르고 EV를 누르면 계산이 된다.

더 심화된 수식(삼각함수, 논리연산, 루트 등)을 이용하기 위해서는 계산기 중간에 있는 아래의 노란 함수기능들을 누르면 된다.



여기에서 만약 삼각함수를 누른다고 하면, 아래와 같이 결과화면에 내가 선택할 수 있는 함수들과 함께, 스크롤이 보여지게 된다. 사용방법은 우측 하단과 같이, 내가 원하는 함수(ex. tan)를 누르고 괄



호(( )를 넣어주어서 사용한 뒤 닫아주어야 한다. 다만 범위는 지정해 줄 수 없어서, tan는

만약 위에서 함수를 잘못 클릭했다고 했을 때 사용하지 않고 끄기 위해서는 다시 올바른 함수버튼을 클릭하던지 혹은 주황색의 Close 버튼을 누르면 위의 result 창에 있는 함수 목록들이 닫히게 된다.

AC는 흔히 아는 Clear의 역할을 하며 'Back'은 사용자가 수식을 잘못 썼을 경우, Clear가 아닌 한 칸 뒤로 가기(***BackSpace***)의 역할을 한다.

무리수 같은 경우는 pi, e, i를 상수처럼 사용할 수 있어서 그냥 수처럼 사용이 가능하며(ex. pi/2), 논리 연산은 같은지, 다른지, 큰지, 작은지 등 Return 값이 True나 False가 나오게 되는 함수이다(ex. ***7 != 7 EV=> False***).

변수는 a, b, w, x, y, z 의 변수들에 '='을 이용하여 값을 넣어줄 수 있으며(ex. ***x = 7, x EV => 7***), 변수들을 이용해서 계산을 한다던지 f(x) 버튼에 있는 함수들을 사용 가능하다.

삼각함수는 하이퍼볼릭 같이 잘 안 쓰더라도 막상 쓰려고 하는데 없으면 또 불편하기 때문에 math.js에서 사용가능한 모든 삼각함수를 전부 등록해 놓았다. 하이퍼볼릭 함수들, 역함수, 역수 등 바로바로 사용하는 것이 기본을 이용해서 바꾸는 것보다는 좋아보였다. 삼각함수들은 모든 삼각함수(sin부터 coth까지)는 괄호를 열어서 1가지의 인자만 사용한다.(ex. ***coth(pi)***)

지수로그에서 exp는 밑이 e인 지수함수로서 ***exp(10)***과 같이 사용이 가능하며, log는 ***log(8)*** 과 같이 사용이 가능하며 밑이 e이다. 바꾸고 싶다면, ','를 이용하여 구분해서 밑이 10인 log100을 구하고 싶다면, ***log(100, 10)***으로 사용가능하다.

벡터는 바깥에 기본 수식들과 같이 있는 ***[,]***를 이용하면 사용가능하고, 행렬인 det는 대괄호([,])를 2개 이용하여, 사용가능하다. cross는 외적으로 ***cross([1,2,3],[1,2,3])***같이 사용가능하고, dot는 내적으로 ***dot([1,2,3], [1,2,3])***와 같이 사용이 가능하다.

제곱루트에서 ^는 제곱으로 2의 4승을 구하고 싶다면, ***2^4***처럼 사용하면 된다. sqrt는 밑이 2인 루트이며, ***sqrt(8)***과 같이 사용이 가능하다.

f(x) 버튼에 있는 함수들('f(x), g(x), h(x)')은 모두 사용방법이 같다. 예로, ***f(x) = x^2 + 3*x + 2*** 와 같이 표현이 가능하고, 이 식을 EV누른 뒤에 ***f(7)***을 누르게 된다면, 앞서 미리 등록해 놓은 f(x) 함수에 7이라는 값이 들어가서 수식을 푼 값이 나오게 된다. f(x) 내부의 x는 다른 변수들로 대체 가능하다.

(3) 특징적인 상호작용 방식들에 대한 세부 구현 방법

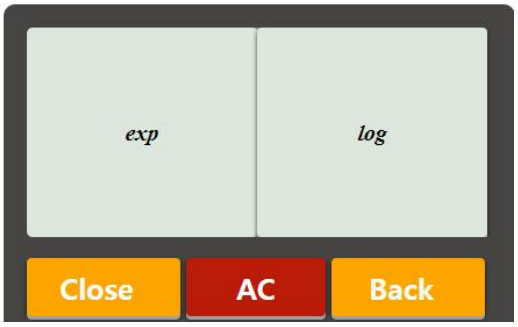
먼저 result 내부화면에 값들을 함수 버튼들을 띄우기 위해서 result div아래에 div를 하나 더 생성하였다. 본래 의중은 result안에 div를 넣어서 함수 목록들을 띄우려고 했는데, 값 출력하는 것 때문인지 result div안에 div가 생성 되지 않아서(생성은 되지만, 내부에서 작동을 하지 않아서) result 아래에 만들어준 다음에 margin-top : -195px를 해주어서 위치를 맞추어 주었다. <예시> 또한, 크기를 맞춰주기 위해서 내부 버튼들이 2개, 4개, 6개 이상이 되는 경우들을 나눠주어서 width와 height를 다르게 맞춰주었다.

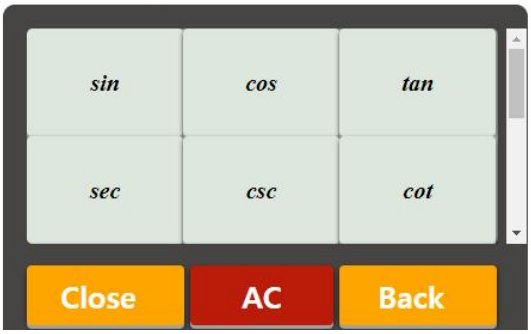
```
<div id="result">

</div>
<div id="select_btn">
    <div class="inner-key" id="fx">
        <span class="key function-key">f(x)</span>
        <span class="key function-key">g(x)</span>
        <span class="key function-key">h(x)</span>
```

```
    <span class="key function-key six">sech</span>
    <span class="key function-key six">csch</span>
    <span class="key function-key six">coth</span>
</div>
<div class="inner-key" id="exp_log">
    <span class="key function-key two">exp</span>
    <span class="key function-key two">log</span>
</div>
<div class="inner-key" id="vec">
    <span class="key function-key six">벡터</span>
    <span class="key function-key six">cross</span>
```

```css
.two{
    width: 198px
}
.four{
    width:198px;
    height:90px;
}
.six{
    height:90px;
}
```





2개 일 때는 위와 같이 화면에 단 2개만으로 꽉 채우기 위해 result width의 절반크기들로 생성하였다. 6개 이상은 6개가 화면에 전부 들어가게 크기를 맞췄으며, 그 이상은 스크롤바가 생기도록 overflow-y : auto를 설정하여 자동으로 스크롤바에 들어가도록 하였다.

```css
.inner-key {
    width: 419px;
    height: 180px;
    display: none;
    padding-left: 0px;
    border-radius: 5px;
    position: absolute;
    overflow-y: auto;
    margin-top: -195px;
    z-index: 2;
}
```

```
else if ($(this).text() == '변수') {
    if (IsShowed) {
        $(Previous).hide();
        $('#xyz').show();
        Previous = '#xyz';
        IsShowed = true;
    }
    else {
        $('#xyz').show();
        Previous = '#xyz';
        IsShowed = true;
    }
}
```

```
else if ($(this).text() == 'Close') {
    $(Previous).hide();
}
<script type="text/javascript">
    $(document).ready(function () {
        var parser = math.parser();
        var displayValue = '0';
        var Previous;
        var IsShowed = false;
```

그리고 내부 버튼들이 눌렸을 때에 띄우는 것과, 다른 버튼을 눌렀을 때 이전에 있던 것을 다시 내리기 위해서 previous라는 변수도 넣어주었고, IsShowed라는 변수를 넣어주어서 현재 result 화면에 함수 버튼이 이미 띄워져 있는지도 판단하게 하여 상황에 따라 켜기만 하던지, 전에 것을 끄고 띄우게 할 수 있도록 하였다. 또한 Close 역시, Previous 변수를 기준으로 끄게 하였다.

위를 위해서 아래와 같이 여러 key css를 더 추가하였다.

```css
.back-key {
    width: 120px;
    height: 40px;
    background-color: orange;
    color: white;
    padding-right: 12px;
    font-size: 25px;
}
```

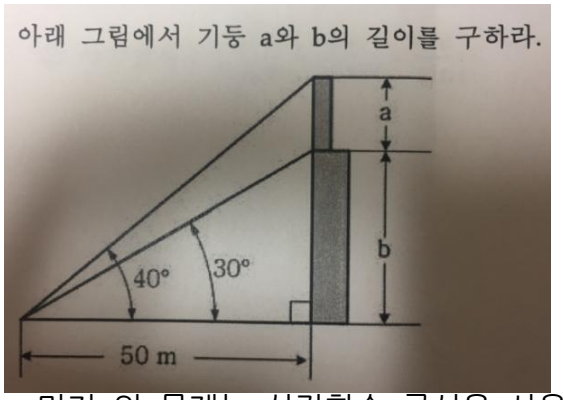
```css
.set-key {
    width: 95px;
    height: 40px;
    background-color: yellow;
    font: bold 18px "Times New Roman"
}
```

```css
.inner-key {
    width: 419px;
    height: 180px;
    display: none;
    padding-left: 0px;
    border-radius: 5px;
    position: absolute;
    overflow-y: auto;
    margin-top: -195px;
    z-index: 2;
}
```

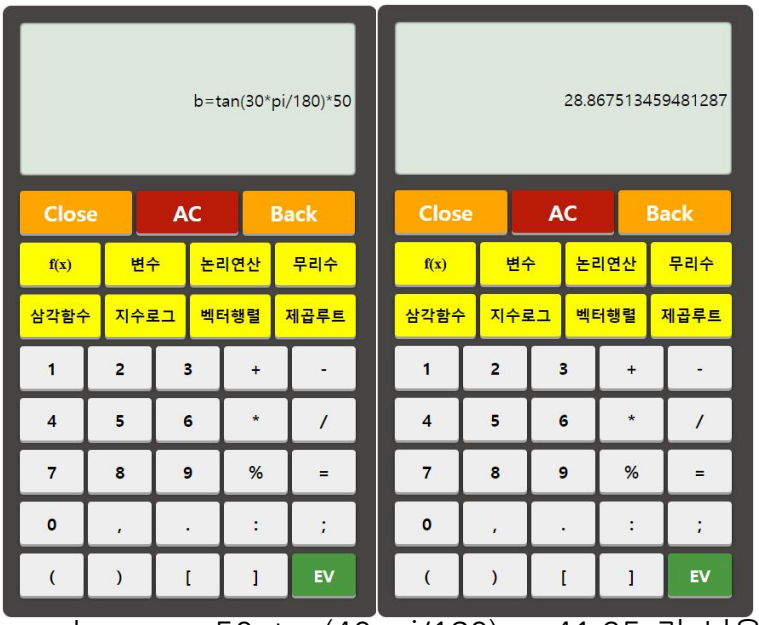
back-key는 ‘Back’ 버튼과 ‘Close’에 들어가는 css이고, set-key는 함수들의 집합에 대한 css, inner-key는 함수 버튼들을 눌렀을 때 나오는 기능들을 포함한 div에 대한 key이다.

(4) 실제 문제에 대한 사용 예시

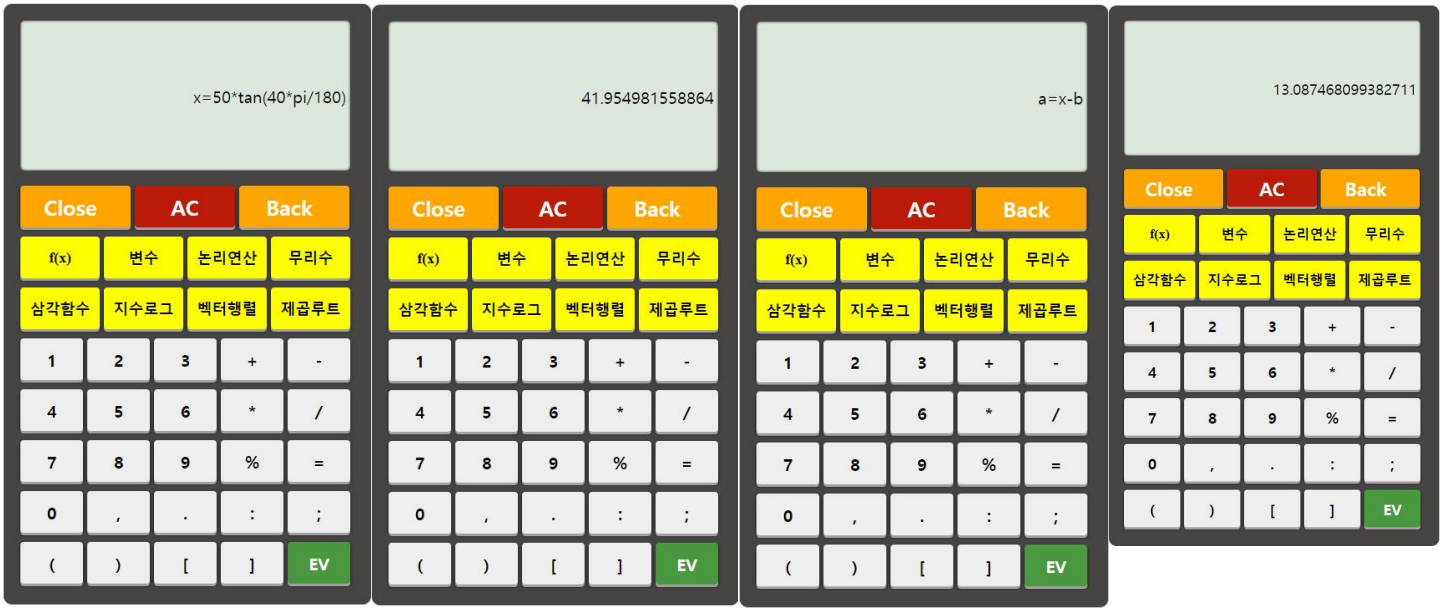
(1)번 문제



먼저 위 문제는 삼각함수 공식을 사용해야 하는 문제이다. 밑변의 길이를 구하기 위해서 tan를 사용한다. 먼저 a를 구하자면, tan(30*pi/180) = b/50을 이용한다. b = 50*tan(30*pi/180) = 28.87이 나오고,

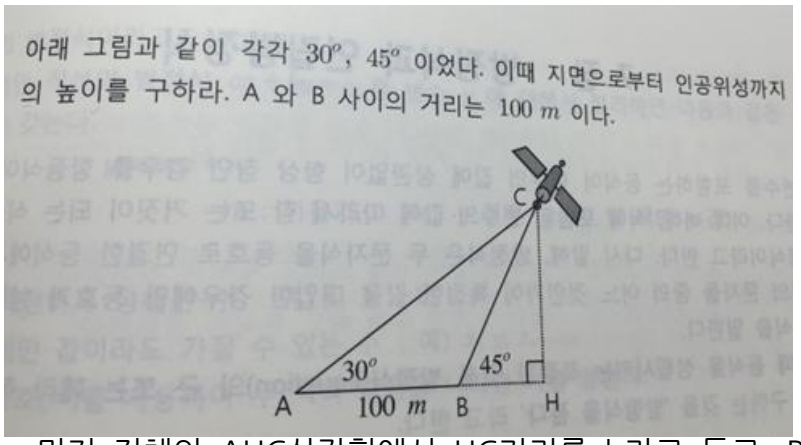


a+b = x = 50*tan(40*pi/180) = 41.95 가 나온다.



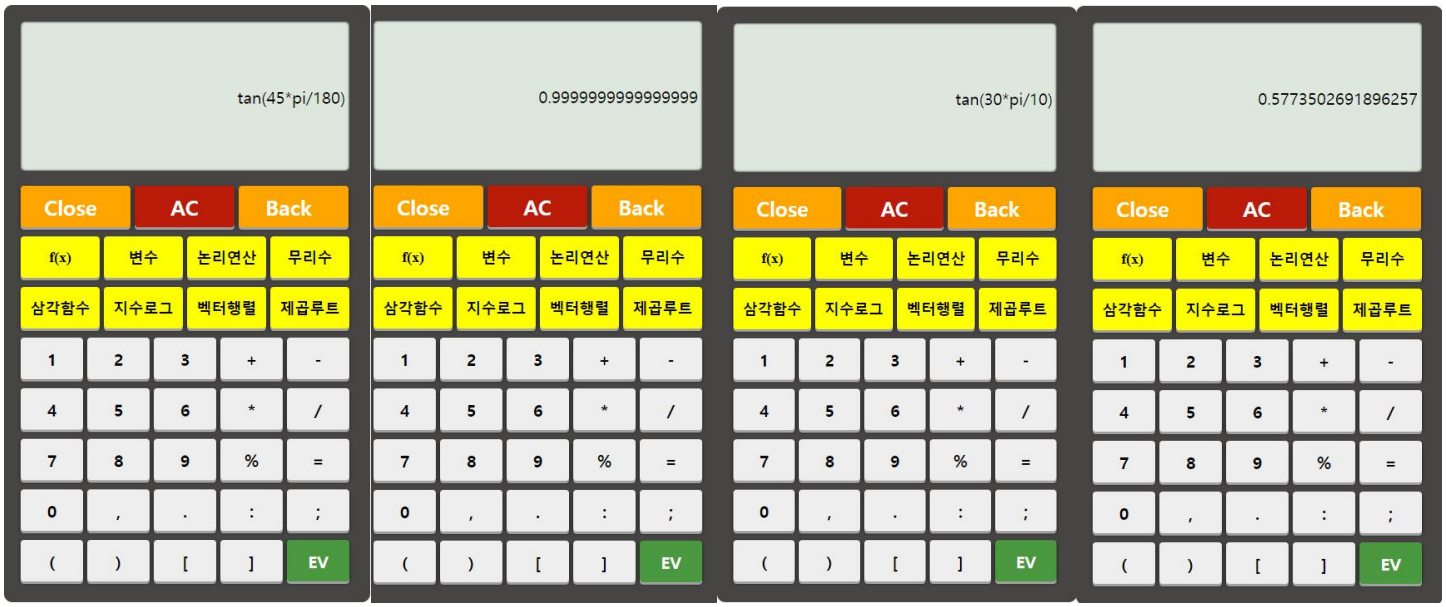
a = 13.1, b = 28.87이다.

(2)번 문제



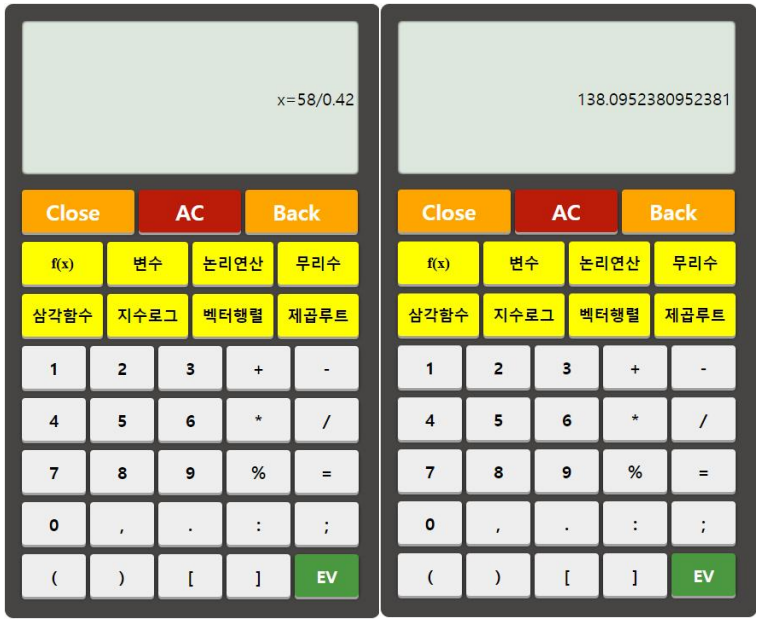
먼저 전체의 AHC삼각형에서 HC거리를 h라고 두고, BH거리를 x라고 둔다. 1번 문제와 비슷하게, tan(30*pi/180) = h/(100+x) 이다. h = (100+x)*tan(30*pi/180)이다.

그리고 작은 BHC 삼각형에서 tan(45*pi/180) = h/x이다. h = x*tan(45*pi/180)을 연립하여 풀면 된다.



tan(45*pi/180) = 1이므로, h=x 이다. 그리고 tan(30*pi/180) = 0.58 이다.

따라서, h = (100+h)*0.58 => 0.42*h = 58 => h = 58 / 0.42 = 138.1m이다.

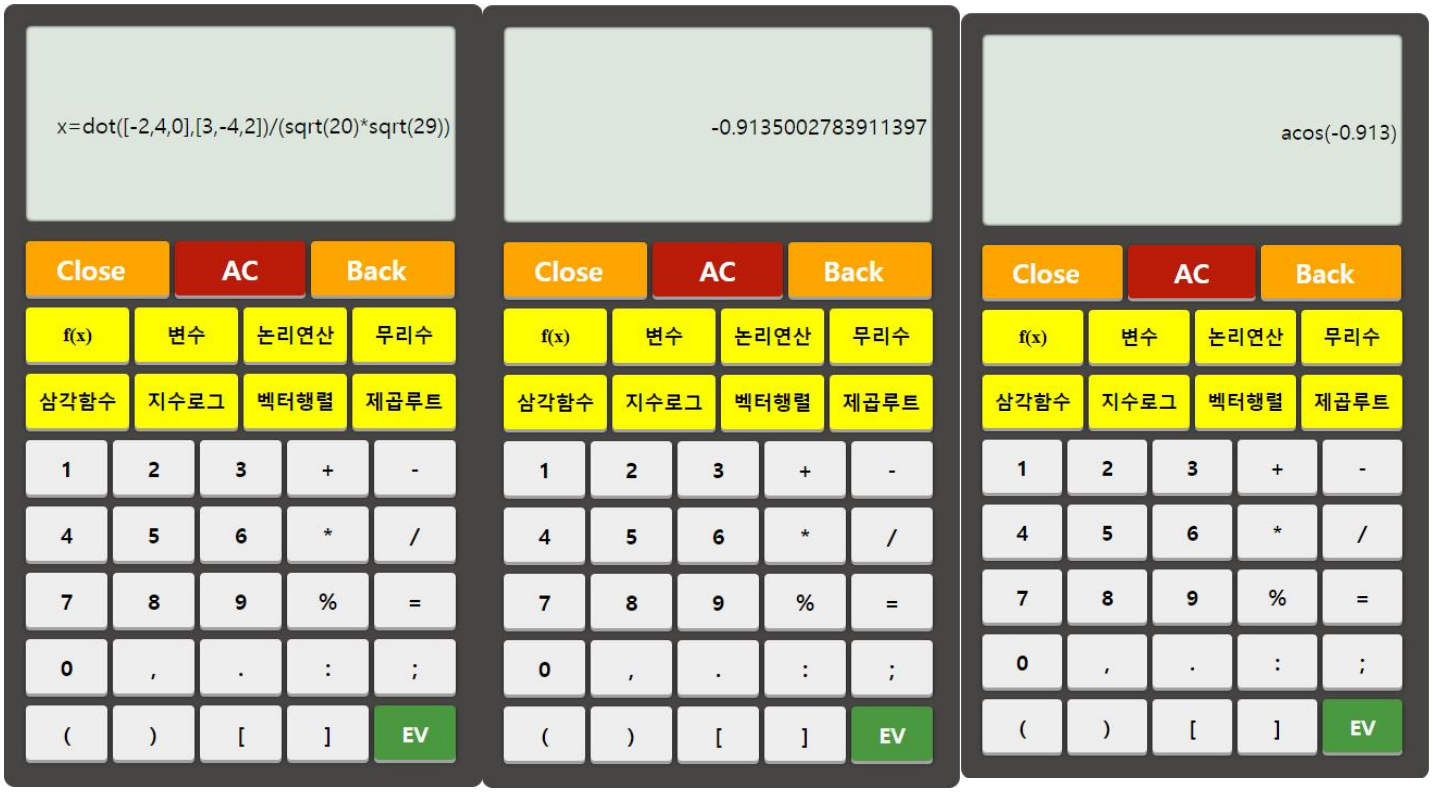


(3)번 문제 : 두 벡터사이의 각도를 구하여라

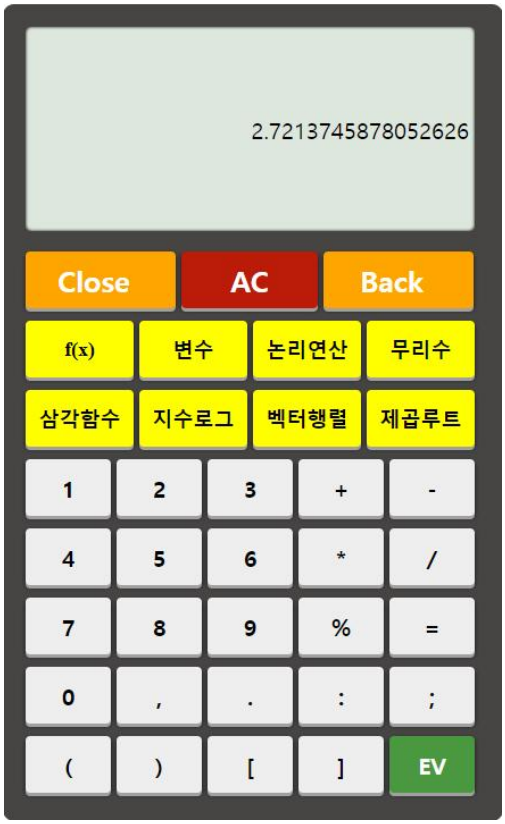
A: -2i + 4j B : 3i - 4j + 2k

A = [-2,4,0] , B = [3,-4,2]. $\cos\theta$ = AB내적/(A길이*B길이)

A의 길이는 $\sqrt{20}$, B의 길이는 $\sqrt{29}$



acos으로 역수를 구한다. 그러면 156도가 나온다.



3. 논의

(1) 구현 측면에서 성공적인 부분과 실패한 부분

먼저, 초기 아이디어로는 매뉴얼이 필요 없게, 사용할 함수를 누르면 한 측면에 그 함수에 대한 사용 예시(ex. $\log_{10}100=\log(100,10)$)를 주려고 하였다. 하지만 저런 것 같은 경우는 코드에서가 아니라 이미지를 따오는 것이 나을 것 같다고 생각이 들었고, 아직 함수들을 다 정하지도 않았고, 구현도 제대로 되지 않았었는데 저 이미지를 만들고 그 이미지를 출력시키는 것 까지는 빠듯할 것 같아서 아직 하지 못하였다. 또한, result 내부 div에 div를 넣어서 함수들을 보이게 하려고 하였으나, 실패하였다. 아마도 결과값이 나오는 곳이라서 안에 div가 들어가지 않는 것 같았고, 대안으로 아래에 div를 넣어서 그것을 margin-top: -195px를 주어서 성공시켰다. 또한 삼각함수처럼 내부 기능들이 많아졌을 때, 스크롤 기능을 넣고 싶었는데 구글을 통해서 검색하여 찾을 수 있었다.

마지막으로 더 많은 기능들을 넣고 싶었지만 계산기의 크기가 한정되어서 그 노란 부분의 함수들 옆에 옆으로 기울여진 삼각형(▶)을 하나 삽입하여서 옆으로 넘기면 다른 함수들이 뜨게 하고 싶었지만, 아직 구현하지 못하였다. 추가적으로 구현할 계획이다.

(2) 사용성 측면에서 긍정적인 측면과 부정적인 측면

먼저 국내의 일반적인 수학 문제를 푸는 공대생들에게는 매우 긍정적이라고 생각한다. 딱 보았을 때 뭐가 무엇인지 정확히 알 수 있으며, 그들에게는 쓸 데 없는 자잘한 것들이 없기 때문이다. 하지만, 원래 생각했던 간단한 예시가 뜬다고 했더라면, 매우 긍정적이었을 테지만, 아직은 큰 메리트가 없다고 생각한다.

또한, 일반 공학용 계산기를 자주 사용하는 공대생들에게는 사용할 수 있는 함수들이 많이 없어서, 이 공학용계산기를 그들이 전공문제를 푸는데 사용하지 않을 것 같다.

(3) 과제 #2에 대한 전반적인 자체 평가 및 향후 개선 계획

이번 과제 #2를 통해서는 외형적인 측면은 완성되어 기능만 더 추가만 하면 된다고 생각되고, 앞으로 어떤 기능들을 추가하고, 어떤 식으로 배치할지 대략의 방향은 잡혔다고 생각한다. 원래 과제 #1에서 노트에 그려가며 구상한 대로 짜고 싶었으나, 실제로 구상한 대로 구현을 해보니 오히려 보기 싫고, 너무 텅텅 비어있다는 생각이 들어 외형적인 측면을 거의 뜯어 고쳤다. 하지만 math.js로 인해 내부 구현을 하지 않아도 된다는 점이 매우 다행이라고 생각된다.

본인의 코드가 매우 비효율적으로 생각되는 점이 모두 하나하나 일일이 지정을 해줘야 한다는 점인데, 뭔가 다른 계산기들과 다른 차별점이 있게 하려면 아무래도 예시가 뜨면서 누가 봐도 어떻게 사용해야 하는지 알 수 있도록 해야 할 것 같다.

시연영상 링크

https://youtu.be/VOISwC16hSI