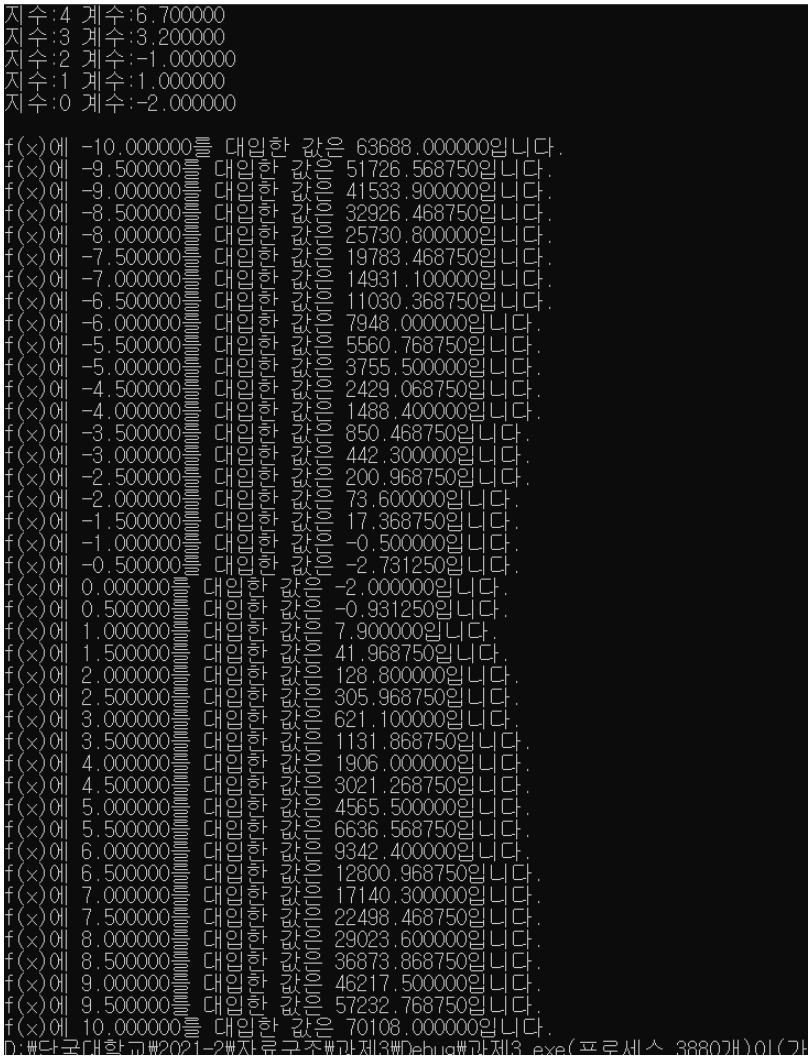
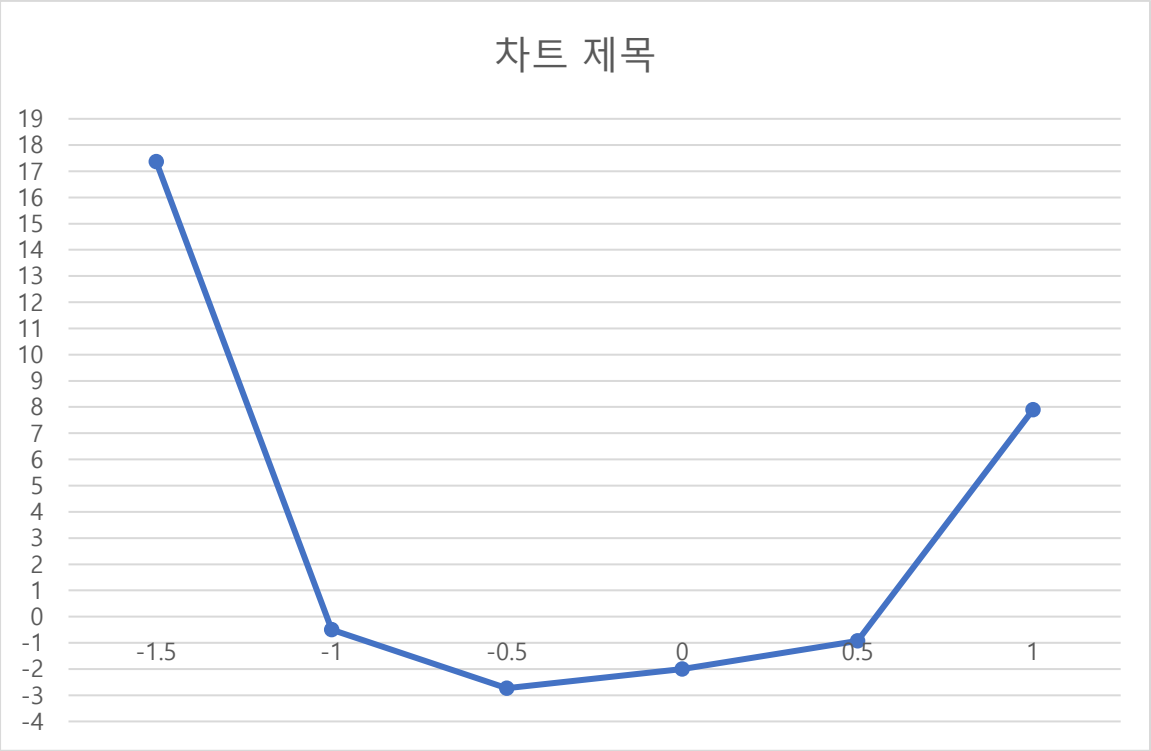
문제1

고차방정식의 해를 구하기 위해 주어진 x의 범위를 단위별로 대입해 결과를 얻어내고 그렇게 나온 결과 중 내가 원하는 결과에 가장 가까운 x를 찾아낸 후 그 사이에 더 범위를 세분화하면서 대입을 반복하다보면 내가 원하는 결과에 근사하는 값을 가지는 x를 알아낼 수 있을 것이라고 생각했다. 그러기 위하여 선형 리스트 구조체를 만들어 f(x)를 구성하고, x값을 대입해 결과를 관찰하는 함수를 따로 설계하였다.

먼저 f(x)=0을 만족하는 x와 f(x)가 최속=가 되는 x값을 만족하기 위해 아래 결과와 같이 선형리스트로 함수 f(x)를 만들고 x범위 [-10,10] 에서 0.5씩 증가시켜가며 값을 결과를 관찰하였다. 그 결과 -10 부터 -0.5까지 꾸준하게 값이 감소 했으며 그 이후에 다시 증가하는 양상의 그래프임을 알 수 있었다. X=-1.5 일때 약 17 이고 x=-1 일때 -0.5 였고, X=0.5 일때 약 -0.93 이고 x=1 일때 7.9 였으므로 이 사이의 범위를 더 세분화하여 대입하면 f(x)가 0이 되는 지점을 찾을 수 있을 것이라고 생각했다. 또한 x=-1.0~0.0에서 각각 약 -0.5, -2.73125, -2.0 이였다. 즉 감소하다가 증가하는 형태인데 이 근방의 x값들을 더 세분화하여 더 작은 값이 있는지 추가로 확인하기로 했다.





여러 번 반복한 결과 출력 화면을 첨부하진 못했지만 다음과 같은 결과를 얻어낼 수 있었다.

f(x)에 -1.040000를 대입한 값은 0.116888입니다.

f(x)에 -1.030000를 대입한 값은 -0.046717입니다.

f(x)에 -1.033000를 대입한 값은 0.001690입니다. (0에 가장 근접)

f(x)에 -1.032000를 대입한 값은 -0.014509입니다.

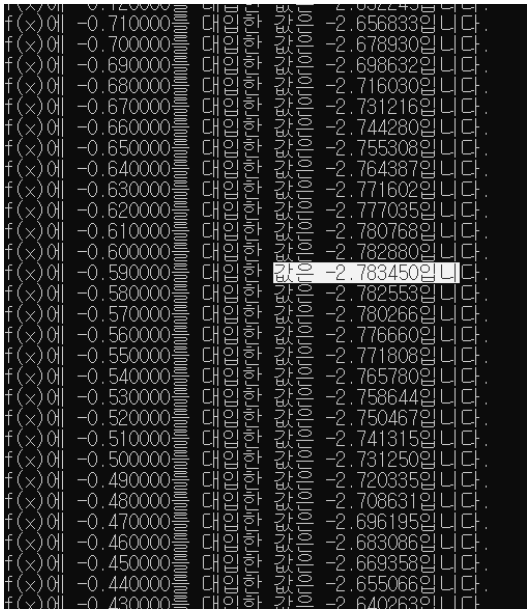
f(x)에 0.620000를 대입한 값은 -0.011736입니다.

f(x)에 0.630000를 대입한 값은 0.088699입니다.

f(x)에 0.621000를 대입한 값은 -0.001878입니다.

f(x)에 0.622000를 대입한 값은 0.008021입니다.

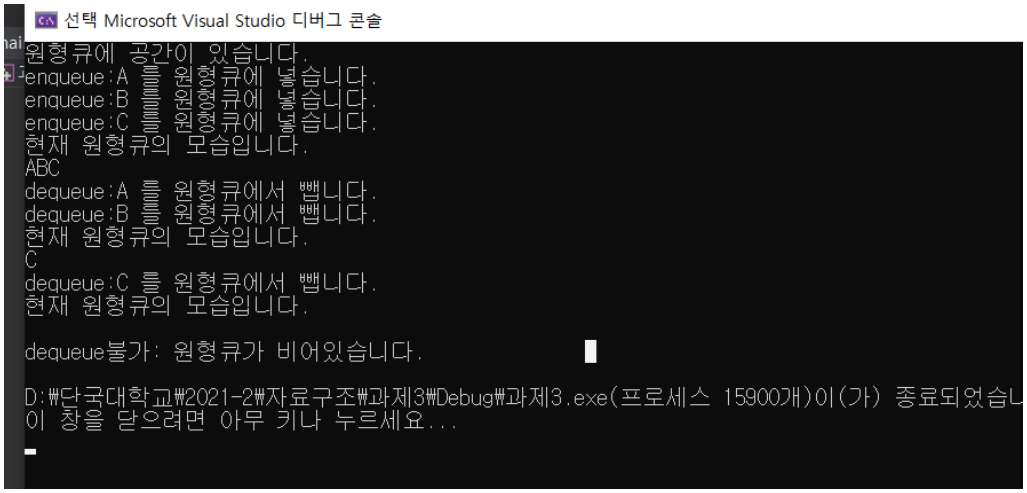
또한 x = -0.59에서 -2.783450으로 최소값에 가까운 값을 얻어낼 수 있었다.



시간이 없어 x를 대입하는 것으로 2,3번을 해결했지만 f(x)를 비문하여 기울기가 0인 x의 근사값을 찾고 그 x를 미분전 함수에 대입하며 실제 최솟값인지 확인하는 방법(기울기가 0이라고 해서 항상 그래프가 아래쪽에 있는 것은 아니기 때문)으로 더 개선된 결과를 얻어낼 수도 있을 것 같다고 생각했다.

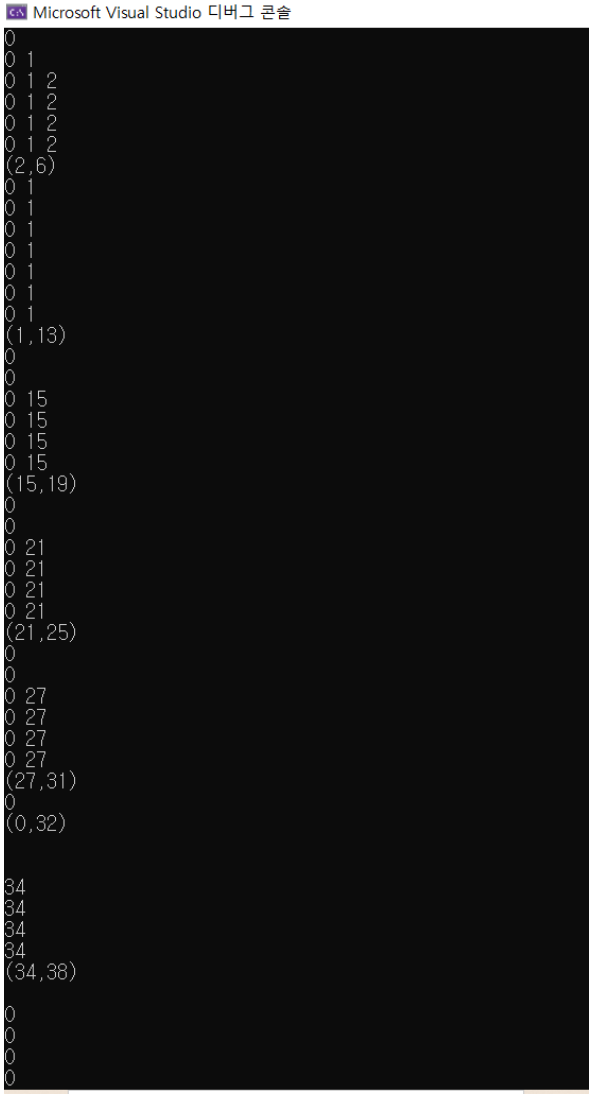
문제2

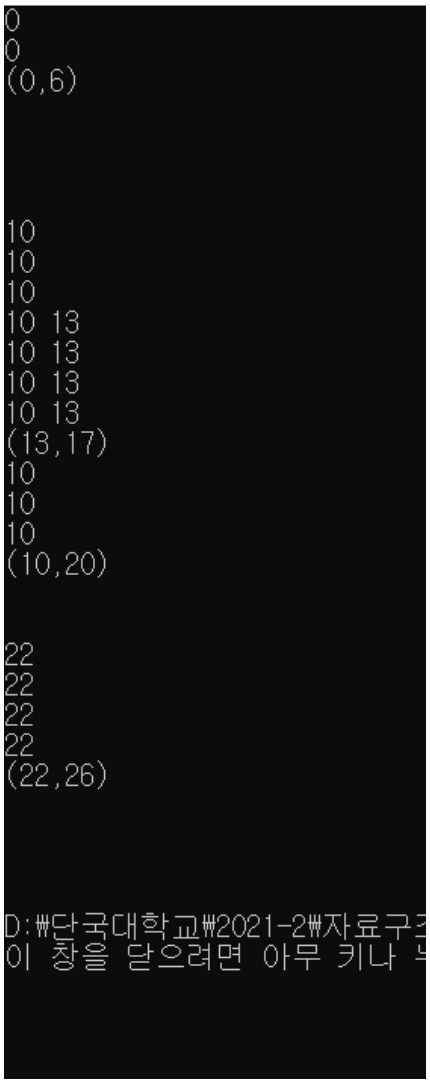
동적으로 원형큐를 생성하고 문자를 넣을 수 있도록 설계했다. 큐 생성 및 할당, 삽입, 삭제, 출력, 공간 여부 확인 등의 함수를 모두 독립적으로 작성했다. isEmpty()를 통해 원형큐에 공간이 있는지 알 수 있지만 삽입, 삭제 중 큐에 공간이 없거나 혹은 삭제할 원소가 없을 때도 메시지를 출력한다.



문제3

'(' 를 만나면 스택에 해당 인덱스를 넣고 ')' 를 만나면 가장 '(' 와 ')' 인덱스를 출력한다. 가장 늦게 들어간 '('가 가장 먼저 만난 ')'와 쌍을 이루기 때문에 스택의 top을 출력해주면 된다. 매 순간마다 스택의 상태(결과)를 출력해줘서 중첩 '(' ')' 가 어떻게 스택에 쌓여 있는지 확인할 수 있었다.





줄단위로 스택을 출력(가장 오른쪽이 top)하고 각 pair 순서쌍을 출력했다. (공백은 스택에 아무것도 없는 경우이다.)

문제4

연산자의 우선순위를 고려하여 infix 식을 postfix 식으로 변환하기 위해 스택을 사용하였다.

해당 식 5개를 가지고 테스트 해본 결과이다.

```
char ex1[] = "(A+B)*C-D";
char ex2[] = "A*B*C/D+E+F*G";
char ex3[] = "A/B+(C*D)-E";
char ex4[] = "A*(B+C)-E/F";
char ex5[] = "A+B+C+D+E";
```