

객체지향 프로그래밍3주차 과제

2016112158 김희수



<1번>

static double Newton\_Method\_SQRT(double X, int ntimes)

{

    if(ntimes == 0)

        return 1.0;

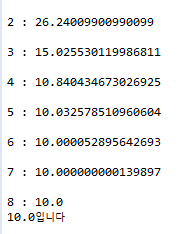
    double before = Newton\_Method\_SQRT(X, ntimes-1);

    return before - (before\*before - X) / (2\*before);

}

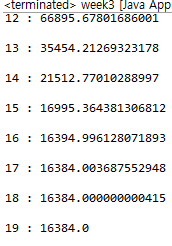
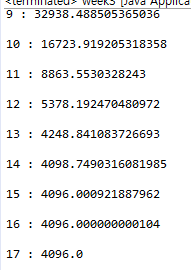
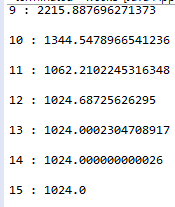
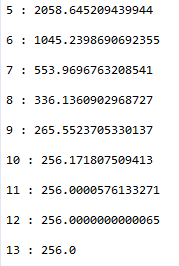
제곱근을 구하는 메소드는 다음과 같다. 점화식이 주어졌으므로 recursion을 이용한다면 손쉽게 메소드를 구현할 수 있다.

100의 제곱근이 10을 출력하는 최소의 반복회수는 다음과같다.



10.0인지 아닌지 판단할 때 if(Newton\_Method\_SQRT(100,i) == 10.0)이면 10.0입니다를 출력하도록 프로그래밍하였다. 그 이유는 혹여 결과로 나온 10.0에 부동소수점 오차가 일어나진 않았는가 궁금했다.

다음은 2^16, 2^20, 2^24, 2^28의 제곱근을 구하려고 할때의 반복횟수이다



재밌는 점은 2의 거듭제곱에서 지수가 2씩 증가할수록 제곱근을 구할때의 반복회수가 1씩 증가한다는 것이다. 사진에선 지수가 4씩 증가해 반복회수는 2씩 증가하였다.

<2번>

static double Newton\_Method\_CubeRoot(double X, int ntimes)

{

    if(ntimes == 0)

        return 1.0;

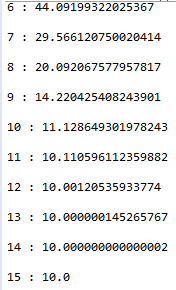
    double before = Newton\_Method\_CubeRoot(X, ntimes-1);

    return before - (before\*before\*before - X)/(3\*before\*before);

}

세제곱근을 구하는 메소드는 위와 같다. 역시 recursion을 활용해 구현하였다.

1000의 세제곱근인 10을 출력하는 최소의 반복횟수는 다음과 같다.



2^15, 2^18, 2^21, 2^24, 2^27의 세제곱근을 구하기 위한 최소의 반복횟수

