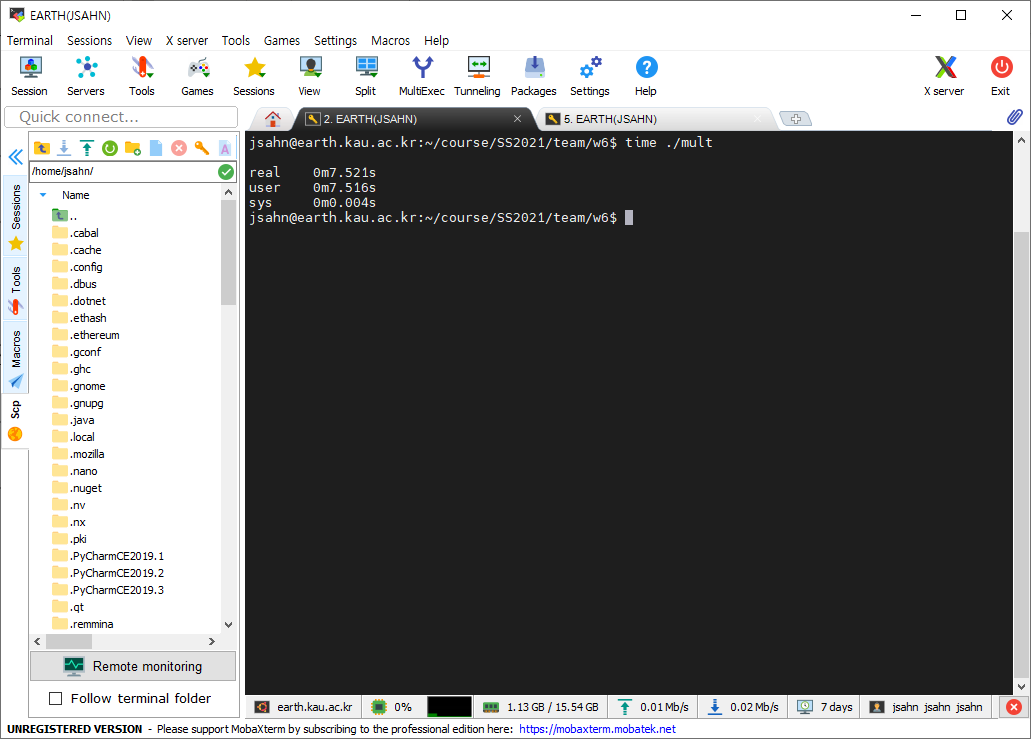
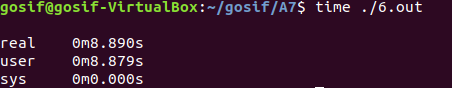
**고급시스템프로그래밍 6주차 실시간수업 과제**

|  |
| --- |
| 제출일: 2021.04.22  이름: 김현용 |

첨부 프로그램은 512x512 곱셈을 10번 수행하는 프로그램이다. 프로그램의 실행시간은 다음과 같이 time 명령어로 측정할 수 있다.

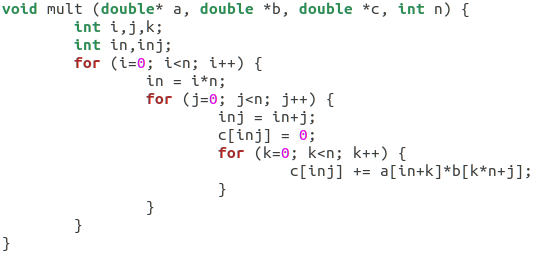


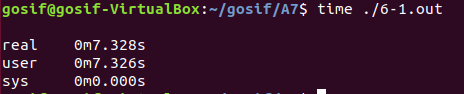
이에 대하여 다음 물음에 답하시오.

※ 그냥 코드를 실행한 경우 (8.89초 정도로 측정됨)  


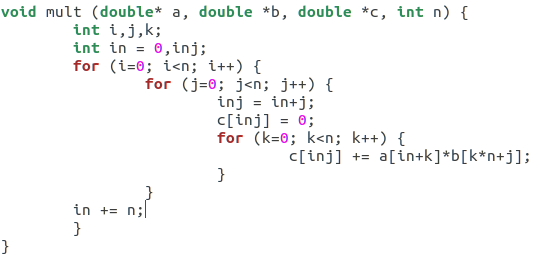
1. Code Motion과 Strength Reduction을 사용하여 프로그램을 최적화하고, 수행시간 단축을 측정하시오.

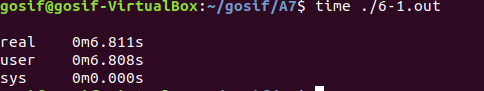
**A1-1) Code Motion만 사용한 경우**



  
Code Motion만 사용한 경우 대략 1.5초정도 실행시간이 줄어든 것을 알 수 있음

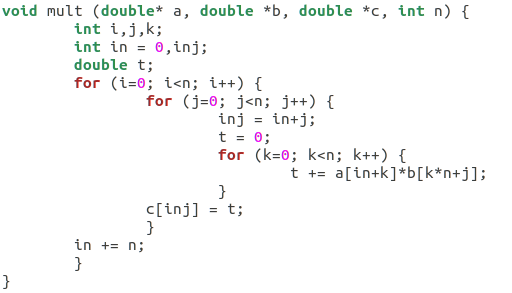
**A1-2) Strength Reduction도 사용하여 프로그램 최적화**

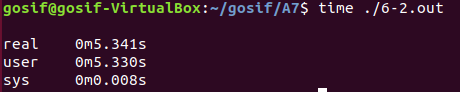


  
in = i\*n의 곱셈 연산 대신에 덧셈 연산으로 바꿔주었을 경우 대략 0.5초정도 실행시간이 더 줄어든 것을 확인할 수 있었다.  
※ k\*n의 경우 바로 아래에 kn += n;을 하고 k for문 전에 k = 0을 넣어서도 시도해봤지만, 코드가 오히려 더 늘어나서 그런지 오히려 실행시간이 늘어났다. 따라서 무작정 곱셈 연산을 덧셈 연산으로 바꾼다고 실행시간이 더 좋아지진 않는다.

1. 1번의 결과에 지역변수에 계산 쌓기(**Accumulate in temporal local memory)** 최적화를 적용하여 수행시간 단축을 측정하시오. 이러한 최적화는 컴파일러가 자동으로 수행할 수 없다. 그 이유를 설명하시오.

**A2-1) 지역변수에 계산 쌓기 적용**

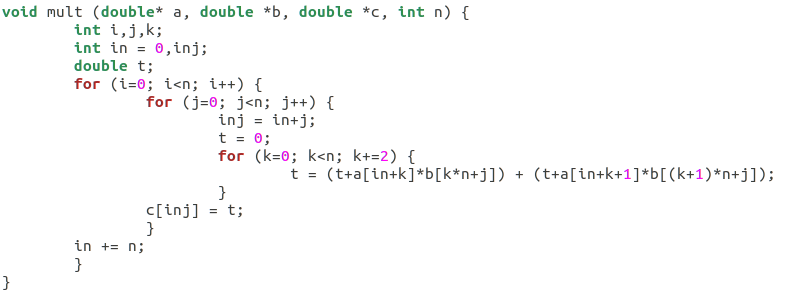


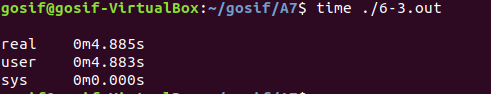


지역변수에 계산 쌓기를 적용했을 경우 이전보다 대략 1.5초정도 더 실행시간이 단축된 것을 확인할 수 있었다.  
**A2-2) 컴파일러가 자동으로 최적화 못하는 이유**  
c배열이 혹시나 a나 b 배열의 주소를 가리키고 있는 경우에는 c배열이 업데이트 되면서 a,b의 배열 값이 변경될 수 있어 그에 따라 계산 결과가 다르게 나올 수 있으므로, 컴파일러가 마음대로 지역변수에 계산 쌓기를 적용할 수 없게 되는 것이다.

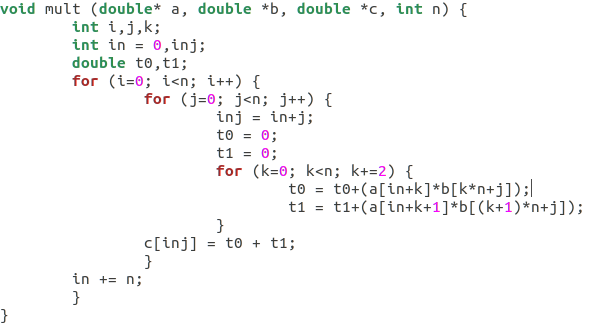
1. 2번의 결과에, Loop Unrolling을 사용하여 최적화하고, 수행시간 단축을 측정하시오.

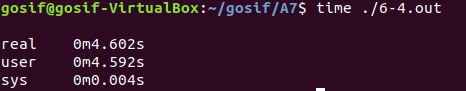
**A3-1) Loop Unrolling 사용**



  
t의 의존성 문제 (다음 t 계산을 위해 이전 t 계산이 끝날 때까지 기다려야함)의 해결을 위해 두 개의 t의 계산을 하는 것으로 변경하는 Loop Unrolling을 적용했더니, 이전보다 대략 0.5초정도 줄어드는 것을 알 수 있다.

1. 3번의 결과에 Separate Accumulation을 적용하여 최적화를 수행하고 수행시간 단축을 측정하시오.

A4-1) Separate Accumulation 적용  


  
Separate Accumulation 까지 적용했을 경우 효과는 미미하지만 0.2초정도 줄어든 것을 알 수 있다.