과제1

Q. 각각 피연산자 99를 표현하기 위해 몇 개의 십진수 자리가 필요한가?

=> 2자리

두 N자리 십진수 정수의 곱을 표현하기 위해서는 최대 2N자리가 필요하다.

Q. 곱셈의 결과를 표현하기 위해서 몇 개의 십진수 자리가 필요한가?

=>4자리

곱셈 결과로 64비트까지 표현을 할 수 있기 때문이다.

과제2

Q. 두 개의 작은 크기의 정수를 곱셈하였다. 결과는 어디에 있겠는가?

=> MFHI, MFLO 명령어를 이용해서 hi 와 lo 레지스터에 저장한다.

과제3

Q. 두 개의 곱셈 명령을 하고싶다. 2번째 곱셈을 하기전에 꼭 hi와 lo에서 결과를 옮겨야 하는가?

=> 옳겨야한다. 왜냐하면 2번째 곱셈을 하기 전에 hi 와 lo 에 담겨져 있는 값들을 옮기지 않으면 2번째 곱셈을 할 때 결과값이 덮어씌여지기 때문에 이전의 값들을 잃어버리기 때문이다.

과제4

4-1)

lui : lui(load upper immediate) 명령어는 16비트 크기의 직접 피연산자를 지정된 레지스터 윗쪽 2 바이트로 복사한다.

ori : ori(Or immediate)명령어는 아래쪽 16비트를 채워 넣는 동작을 한다.

4-2)

hi : 32비트의 레지스터이며 곱셈되어질 때, 32번 비트부터 63번 비트까지 결과물을 담고있다.

lo : 32비트의 레지스터이며 곱셈되어질 때, 0부터 31번 비트까지 결과물을 담고있다.

여기서 주의할점은 hi와 lo레지스터는 다른 산술 명령이나 논리 명령과 함께 사용될 수 없다.

4-3)

mult : 2의보수법 표현방법으로 곱셈연산을 동작한다.

multu : 곱센연산을 하는데 계산시 피연산자들의 음수양수부호를 고려하지 않는다.

4-4)

div : 2의보수법 표현방법으로 나눗셈연산을 동작한다.

divu : 나눗셈연산을 하는데 계산시 피연산자들의 음수양수부호를 고려하지 않는다.

4-5)

mfhi : mfhi(Move From Hi)의 약자로 레지스터 hi로부터 범용 레지스터로 테이터를 가져오는 명령어

mflo : mflo(Move From Lo)의 약자로 레지스터 lo로부터 범용 레지스터로 테이터를 가져오는 명령어

과제5.

.text

.globl main

main:

addi $t0, $0, 20

sw $t0, 0x10010000

addi $t1, $0, 5

mult $t0, $t1

mfhi $t2

mflo $t3

addi $t4, $t3, -99

sw $t4, 0x10010001

li $v0, 10

syscall

