2016 시스템 프로그래밍
- 6 주차 -

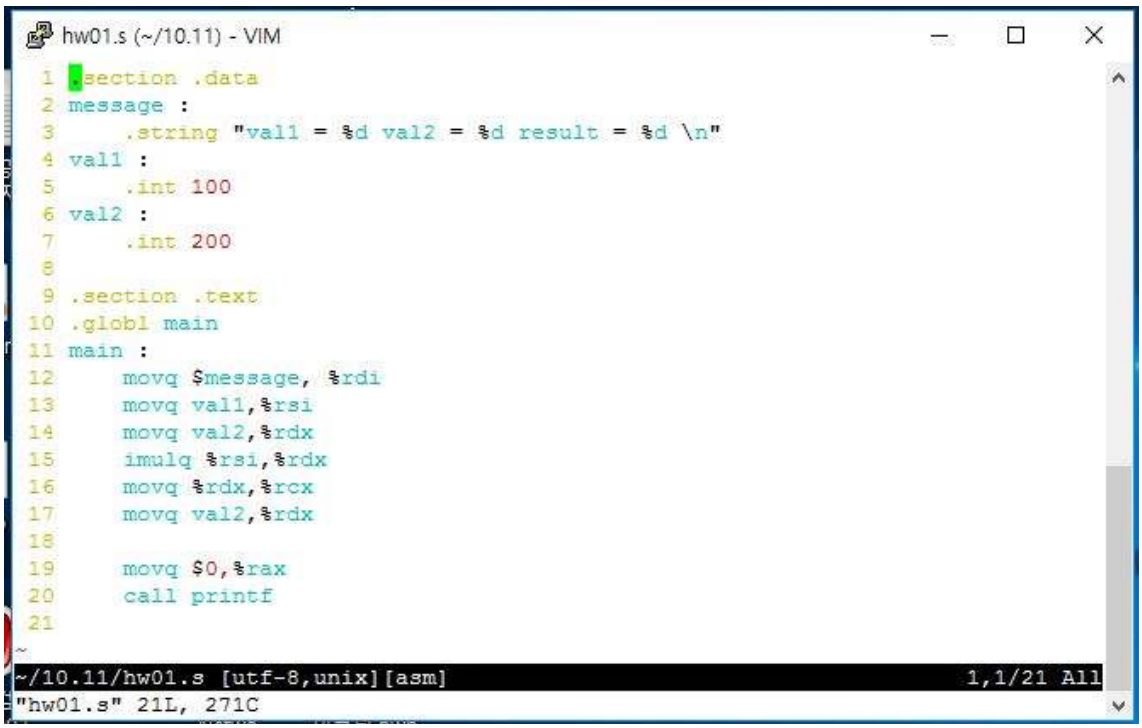
제출일자	2016.10.17
분　　반	00
이　　름	정 윤 수
학　　번	201302482

과제 1

소스코드 스크린샷

1) subq :

2) mulq :



3) incq :


hw01.s + (~/10.11) - VIM
1 .section .data
2 message :
3 .string "val1 = %d val2 = %d result1 = %d result2 = %d \n"
4 val1 :
5 .int 100
6 val2 :
7 .int 200
8 .section .text
9 .globl main
10 main :
11 movq $message, %rdi
12 movq val1,%rsi
13 movq val2,%rdx
14 incq %rsi
15 incq %rdx
16 movq %rsi,%rcx
17 movq %rdx,%r8
18 movq val1,%rsi
19 movq val2,%rdx
20 movq $0,%rax
21 call printf
22
~/10.11/hw01.s [utf-8,unix][+][asm] 13,1/22 All
-- INSERT --

4) decq :


hw01.s (~/10.11) - VIM
1 .section .data
2 message :
3 .string "val1 = %d val2 = %d result1 = %d result2 = %d \n"
4 val1 :
5 .int 100
6 val2 :
7 .int 200
8 .section .text
9 .globl main
10 main :
11 movq $message, %rdi
12 movq val1,%rsi
13 movq val2,%rdx
14 decq %rsi
15 decq %rdx
16 movq %rsi,%rcx
17 movq %rdx,%r8
18 movq val1,%rsi
19 movq val2,%rdx
20 movq $0,%rax
21 call printf
22
~/10.11/hw01.s [utf-8,unix][asm] 1,1/22 All
"hw01.s" 22L, 319C

5) andl :


hw01.s + (~/10.11) - VIM
1 .section .data
2 message :
3 .string "val1 = %d val2 = %d result = %d \n"
4 val1 :
5 .int 100
6 val2 :
7 .int 200
8
9 .section .text
10 .globl main
11 main :
12 movq $message, %rdi
13 movq val1,%rsi
14 movq val2,%rdx
15 addq %rsi,%rdx
16 movq %rdx,%rcx
17 movq val2,%rdx
18
19 movq $0,%rax
20 call printf
21
~/10.11/hw01.s [utf-8,unix][+][asm] 16,15/21 All
-- INSERT --

6) xorl :


hw01.s (~/10.11) - VIM
1 .section .data
2 message :
3 .string "val1 = %d val2 = %d result = %d \n"
4 val1 :
5 .int 100
6 val2 :
7 .int 200
8
9 .section .text
10 .globl main
11 main :
12 movq $message, %rdi
13 movq val1,%rsi
14 movq val2,%rdx
15 xorl %esi,%edx
16 movq %rdx,%rcx
17 movq val2,%rdx
18
19 movq $0,%rax
20 call printf
21
~/10.11/hw01.s [utf-8,unix][asm] 1,1/21 All
"hw01.s" 21L, 270C

결과화면 캡쳐

1) subq :

```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result = 100
a201302482@localhost:~/10.11$
```

2) imulq :

```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result = 20000
a201302482@localhost:~/10.11$
```

3) incq :

```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result1 = 101 result2 = 201
a201302482@localhost:~/10.11$
```

4) decq :

```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result1 = 99 result2 = 199
a201302482@localhost:~/10.11$
```

5) andl :

```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result = 64
a201302482@localhost:~/10.11$
```

6) xorl :

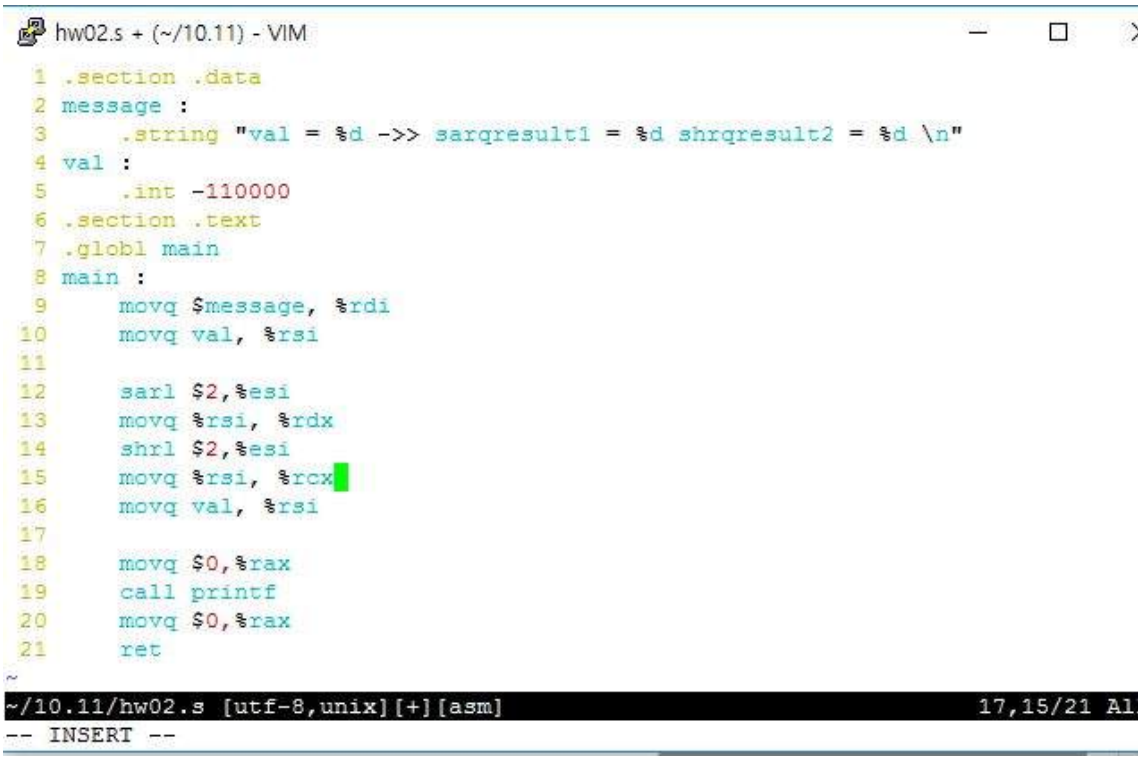
```
a201302482@localhost:~/10.11$ ./hw01.out
val1 = 100 val2 = 200 result = 172
a201302482@localhost:~/10.11$
```

설명

이 어셈블리어 코드는 string변수로 message, int 변수로 val1과 val2를 선언을 하고 값들을 초기화 시켜준다. message변수의 값을 레지스터 %rdi로 옮겨주고 val1,val2의 값을 %rsi,%rdx로 옮긴후 subq,imulq,andl,xorl연산을 이용하여 %rdx에 값을 저장을 한다. %rdx는 출력문에서 2번째 인자에 해당함으로 결과값을 %rcx로 옮겨준후 val2의 값을 다시 %rdx레지스터로 옮겨준다. incq,decq는 문장의 출력문의 변화를 주어 2개의 값이 바뀌는 것을 확인을 할수있게 한다. incq,decq로 인한 연산의 결과가 %rsi,%rdx에 저장이 되면 그 값을 3,4번째 인자에 해당하는 %rcx,%r8에 옮겨주고 val1,val2,의 값을 %rsi,%rdx에 저장을 해준다.

과제 2

소스코드 스크린샷



결과화면

```
a201302482@localhost:~/10.11$ ./hw02.out
val = -110000 ->> sarqresult1 = -27500 shrqresult2 = 1073734949
a201302482@localhost:~/10.11$
```

설명

이 어셈블리어 코드는 sarl명령어와 shrl명령어의 차이를 확인을 하는 프로그램이다. sarl과shrl은 둘다 rightShift연산이다. 이 둘의 차이는 산술적이냐 논리적인가에 차이를 가진다. shrl은 부호에 상관없이 rightShift연산을 하면 빈자리에 0을 채워 넣지만 sarl연산은 rightShift연산시 빈자리에 자신의 부호 비트를 채워 넣는다. 그러므로 양수인값을 연산을 할때에는 서로 값이 같지만 음수인값을 연산을 하게 되면 서로 값이 다르게 나오게 된다.

과제 4

소스코드 스크린샷

```
1  .section .data
2  scanf_str :
3      .string "%d %d"
4  printf_str :
5      .string "%d is less \n"
6  val1 :
7      .int 0
8  val2 :
9      .int 0
10 .section .text
11 .globl main
12 main :
13     movl $val1,%esi
14     movl $val2,%edx
15     movq $scanf_str, %rdi
16
17     movq $0,%rax
18     call scanf
19
20     movl val1,%esi
21     movl val2,%edx
22
23     cmpl %edx,%esi
24     jl L2
25     movl %edx,%esi
26 L2 :
27     movq $printf_str, %rdi
28     movq $0, %rax
29     call printf
30     ret
```

결과화면

```
a201302482@localhost:~/10.11$ gcc -o hw04.out hw04.s
a201302482@localhost:~/10.11$ ./hw04.out
3 5
3 is less
a201302482@localhost:~/10.11$
```

설명

이 어셈블리어 코드는 2개의 값을 입력받아서 2개의 값 중에 작은값을 출력을 해주는 프로그램이다. 출력을 위한 변수 printf_str과 입력을 위한 변수 val1,val2를 선언을 해주고 각각의 값들을 %rdi ,%rsi ,%rdx에 저장을 한다. 그 후 scanf함수를 호출을 하여 val1과 val2에 값을 입력을 한후 입력이 되어진 값을 %esi, %edx에 저장을 한 후 cmpl 명령어를 사용하여 2개의 값중에 작은 값을 찾아낸다 %esi의 값이 작으면 .L2로 바로 이동을 하여 출력을 해주고 %edx의 값이 더 작으면 %edx의 값을 %esi로 옮겨준 후 자동적으로 아래에 있는 명령어가 실행이 되게 하여 결국 마지막으로는 더 작은 값을 출력을 해 줄 것이다.

과제 5

소스코드 스크린샷

```
1  .section .data
2  scanf_str :
3      .string "%d %d"
4  printf1_str :
5      .string "%d %d are equal \n"
6  printf2_str :
7      .string "%d %d are not equal \n"
8  val1 :
9      .int 0
10 val2 :
11     .int 0
12 .section .text
13 .globl main
14 main :
15     movl $val1,%esi
16     movl $val2,%edx
17     movq $scanf_str, %rdi
18
19     movq $0,%rax
20     call scanf
21
22     movl val1,%esi
23     movl val2,%edx
24
25     cmpl %edx,%esi
26     je L2
27     movq $printf2_str,%rdi
28     movq $0, %rax
29     call printf
30     ret
31 L2 :
32     movq $printf1_str, %rdi
33     movq $0, %rax
34     call printf
35     ret
```

결과화면

```
a201302482@localhost:~/10.11$ ./hw04.out
10 10
10 10 are equal
a201302482@localhost:~/10.11$ ./hw04.out
10 15
10 15 are not equal
a201302482@localhost:~/10.11$
```

설명

이 어셈블리어 코드는 두 개의 값을 입력을 받아 서로같은지 다른지 확인을 한 후 결과를 출력을 해주는 프로그램이다. 그러기 위해서 2개의 출력문을 준비해 준다. val1과 val2에 scanf 호출로 값을 입력받아서 저장을 하고 val1과 val2의 값을 %esi,%edx에 저장을 한다. 그 후 cmpl명령어를 이용하여 두 개의 입력받은 값의 관계를 확인하고 두 개의 값이 같다면 .L2로 내려가 두 개의 값이 서로 같다고 출력을 해주고 종료를 한다. 두 개의 값이 서로 다른 값이라면 .L2로 가지 않고 바로 아래에 있는 명령어를 수행을한다. 두 개의 명령어가 서로 같지 않다고하는 출력을 해주고 아래쪽 명령어로 내려가지 않게하기위해 ret으로 프로그램을 종료시킨다.

과제 6

소스코드 스크린샷

```
1  .section .data
2  scanf_str :
3      .string "%d %d"
4  printf_str :
5      .string "result : %d \n"
6  i :
7      .int 0
8  a :
9      .int 0
10 sum :
11     .int 1
12 n :
13     .int 0
14 .section .text
15 .globl main
16 main :
17     movl $a, %esi
18     movl $n, %edx
19     movq $scanf_str, %rdi
20     movq $0,%rax
21     call scanf
22     movl n, %edx
23     movl sum,%ecx
24     movl i, %r8d
25     testq %rdx,%rdx
26     je .L2
27 loop:
28     imul a,%ecx
29     incl %r8d
30     cmpl n, %r8d
31     jl loop
32 .L2 :
33     movl %ecx,%esi
34     movq $printf_str, %rdi
35     movq $0, %rax
36     call printf
37     ret
```

결과화면

```
a201302482@localhost:~/10.11$ ./hw06.out
5 3
result : 125
a201302482@localhost:~/10.11$ ./hw06.out
5 0
result : 1
a201302482@localhost:~/10.11$
```

설명

이 코드는 두 수를 입력을 받고 Loop문을 이용하여 제곱연산을 하여 결과 값을 출력을 하는 프로그램이다. string변수 scanf_str, printf_str을 만들어 주고 두 개의 입력한 값을 저장을 하기 위해서 a,n을 반복문의 조절을 위한 변수 I, 결과를 저장하기 위한 변수 sum이다. 먼저 scanf를 호출을 하여 a와 n에 입력한 값을 저장을 한다. 그 후 n의 값이 0이면 결과 값이 1이 출력이 되게 해야함으로 n과 n을 testq 명령어를 사용하여 and시킨후 n의 값이 0이면 바로 .L2 지점으로 가게 하여 1이 저장이 된 sum변수의 값을 출력을 시킨다. n의 값이 0이 아니면 n번 만큼 Loop문을 수행을 하여 a의 값을 n번 곱셈을 하게 한다. 연산의 결과를 sum에 저장을 한후 Loop문을 모두 수행을 하면 결과 값이 출력이 되도록 한다.

과제 7

소스코드 스크린샷

```c
#include<stdio.h>

int main(void){
    int a,b;
    scanf("%d %d",&a,&b);

    if(a > b){
        printf("a is bigger than b\n");
    }
    else if(a<b){
        printf("b is bigger than a\n");
    }
    else{
        printf("a and b are equal \n");
    }
return 0;
}
```

설명

pdf에 나와있는 과제 7번의 어셈블리어 코드를 기반으로 이 c언어를 생각해 낸 이유는 먼저 어셈블리어 코드의 변수들에서 생각을 받았었다. 입력 형식이 2개의 값을 받아들이는 것과 출력문의 형식이 a와 b의 대소를 비교를 하는 결과를 출력을 해주는 것이었음으로 먼저 생각의 방향을 잡을수 있게 되었다. 또한 cmpl 명령어를 사용하여 %edx와 %eax의 값을 비교를 하고 출력을 달리 하는 것을 보고 이러한 c언어 코드를 생각을 해낼수 있었다.