KIOTT 사물인터넷 임베디드 개발자 과정

(3회차 과제 미완성)

2018-02-25

정유경

과제 1. 스키장에서 스키 장비를 임대하는데 37500원이 든다.
3일 이상 이용할 경우 20%를 할인 해준다.
일주일간 이용할 경우 임대 요금은 얼마일까?

```
#include <stdio.h>
int TOTAL_F(int);
int main(void)
{
	int result=0, day=0;
		printf("이용일수를 입력하세요: ");
		scanf("%d", &day);
		result = TOTAL_F(day);
		printf("%d 일간 이용할 경우 임대요금은 %d원 입니다.₩n", day, result);
	return 0;
}
int TOTAL_F(int day)
{
	double total = 37500 * day;
	if (day >= 3)
		total *= 0.8;
	return total;
}
```

과제3. 1 ~ 1000사이에 3의 배수의 합을 구하시오

```
#include <stdio.h>

int SUM_F(int, int);
int main(void)
{
        int i = 0, sum = 0;
        for (i= 1; i < 1001; i++)
                sum = SUM_F(i, sum);
        printf("□n1~1000사이의 3의 배수의 합은 %d 이다.□n□n", sum);
        return 0;
}
int SUM_F(int i, int result)
{
        if (i % 3 == 0)
                result += i;
        return result;
}
```

과제4. 1 ~ 1000사이에 4나 6으로 나눠도 나머지가 1인 수의 합을 출력하라.

```c
#include <stdio.h>
int FUNC(int, int);
int main(void)
{
	int i = 0, sum = 0;

	for (i = 1; i < 1001; i++)
		sum = FUNC(i, sum);
	printf("₩n1~1000사이에서 4나 6으로 나누어도 나머지가 1인 수의 합은 %d이다.₩n", sum);
	return 0;
}

int FUNC(int i, int result)
{
	if (i % 4 == 0 && i % 6 == 0)
	{
		i++;
		result += i;
	}
	return result;
}
```

과제7. 스택구조의 동작을 어셈블리어를 해석하며 기술할 것

C로 함수 생성시 Stack 구조가 생성된다.
스택구조의 동작을 어셈블리어를 해석하며 기술할 것.
(esp, ebp, eip등의 Register에 어떤 값이 어떻게 들어가는지, 메모리에 어떤 값들이 들어가는지 등을 자세히 쓰시오)

Main() 분석

pushl %ebp : 스택top을 4바이트 감소시켜 확장된 공간에 스택 base pointer를 저장한다.

movl %esp, %ebp : 현재 스택의 top을 ebp 레지스터에 저장한다.

Mult2() 분석

```
#include <stdio.h>

int MULT2(int);
int main(void)
{
        int i, sum = 0, result;
        for (i = 0; i < 5; i++)
                sum += i;
        result = MULT2(sum);
        return 0;
}
 int MULT2(int num)
{
        return num * 2;
}
```

과제10. 구구단 만들기

```
#include <stdio.h>
void GUGU();
int main(void)
{
        printf("구구단을 출력합니다.₩n₩n");
        GUGU();
        return 0;
}
void GUGU()
{
        int i = 0, j = 0;
        for (i = 1; i < 10; i++)
        {
                printf("₩n₩n구구단 %d단입니다.₩n₩n", i);
                for (j = 1; j < 10; j++)
                        printf("%d x %d = %d₩n", i, j, j*i);
        }
}
```

[과제 4번] 1 ~ 1000사이에 4나 6으로 나눠도 나머지가 1인 수의 합을 출력하라.

과제 12-1. 리눅스에서 디버깅 방법 서술하기

디버깅 옵션(-g)을 주어 컴파일 한다. (단, 최적화 옵션인 -O 은 주지 않는다)

gcc -g -o [프로그램명] [소스파일명]

디버거를 실행시킨다

(종료방법 : q 또는 Ctrl + d)

gdb [프로그램명]

프로그램을 실행한다(r) 이때, 중단점을 사용하여 프로그램을 분석할 수 있다

b func : func 함수에 브레이크 포인트 설정

b *0x0000406 : 0x0000406 주소에 브레이크 포인트 설정

디버깅 한다

c (다음 브레이크 포인트를 만날 때 까지 계속 수행한다)

레지스터 값을 출력한다: p $eax (eax 레지스터의 값을 확인한다)

레지스터: eax, ebx, ecx, edx, eip

이때, 출력 형식을 지정한다

p/t var : var 변수를 2진수로 출력

p/x var : var 변수를 16진수로 출력

디스어셈블링하여 전체 스택 동작을 확인한다

disas

과제 12-2. GDB의 명령어 bt와 c에 대해 조사 및 활용하기

1. **bt (backtrace) : 오류가 발생한 함수를 역으로 찾아간다.**
2. **c : 다음 브레이크 포인트를 만날때 까지 디버깅을 계속 수행한다.**

➔ bt 와 c는 스택에서의 동작을 분석할 때 사용한다.

b main 또는 원하는 곳에 브레이크 포인트를 잡고 오류가 발생할 때 까지 c를 통해 진행한다.

오류가 발생하고 디버그가 멈추면

bt를 이용하여 어떤 함수에서 호출 시에 문제가 발생하였는지 확인할 수 있다.