TI DSP, MCU, Xilinx Zynq FPGA 기반의 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 김형주
mihaelkel@naver.com

1번

문제	단 한 번의 연산으로 대소문자 전환을 할 수 있는 연산에 대해 기술하시오.
내용	C 언어로 프로그램을 구현하시오.

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
int main(void){
    char input_char;

    scanf("%c",&input_char);

    printf("%c\n",input_char^32);
}
```

2번

문제	Stack 및 Queue 외에 Tree 라는 자료구조가 있다.
내용	이 중에서 Tree 는 Stack 이나 Queue 와는 다르게 어떠한 이점이 있는가 ?

Ans)
Tree는 Stack이나 Queue구조에 비해 검색속도가 빠르다
128개의 데이터중 특정 데이터를 검색한다고 할 때,
Stack이나 Queue구조는 평균 64번의 비교를 해야 하지만,
Tree구조의 경우 평균 3.5번의 비교를 하면 되므로 64번에 비해 매우 빠르다고 할 수 있다.

3번

문제	임의의 값 x가 있는데, 이를 4096 단위로 정렬하고 싶다면 어떻게 해야할까 ?
내용	C 언어로 프로그래밍 하시오.

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
#include <stdlib.h>
int main(void){
    int num = 1111231232, size = 1, i = 1;
    int* a;
    a = (int*)malloc(sizeof(int)*size);
    printf("%d = ",num);
    while(1){
        *(a+size-1) = num&(0xfff);
        num>>=12;
        if(!num)
            break;
        size++;
        a = (int*)realloc(a,sizeof(int)*size);
    }
    printf("0x ");
    while(size-i+1){
        printf("%x ",*(a+size-i));
        i++;
    }
    printf("\n");
    return 0;
}
```

4번

문제	int p[7] 와 int (*p)[7] 가 있다.
내용	이 둘의 차이점에 대해 기술하시오.

Int p[7]이 아닌, int *p[7]이라 생각하고 답하겠습니다.

Ans)
포인터 배열과 배열 포인터
Int *p[7] : 포인터 배열로, int형 포인터(주소값)을 저장하는 배열 7개.
Int (*p)[7] : 배열 포인터로, int형 7개짜리 배열의 주소를 저장하는 포인터

5번

문제	다음을 C 언어로 구현하시오.
내용	이번 문제의 힌트를 제공하자면 함수 포인터와 관련된 문제다. 아래와 같은 행렬을 생각해보자! 1 2 3 1 2 3 sapply(arr, func) 이라는 함수를 만들어서 위의 행렬을 아래와 같이 바꿔보자! sapply 에 func 함수가 연산을 수행하는 함수로 만들어야 한다. 1 2 3 1 4 9

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>

void func(int (*p)[3]){
    int j;
    for(int j=0;j<3;j++)
        p[1][j] = p[0][j] + p[1][j];
}

void sapply(int (*arr)[3],void (*p)(int(*)[3])){
    p(arr);
}

int main(void){
    int arr[2][3]={{1,2,3},{1,2,3}};
    int i, j;
    sapply(arr,func);

    for(i=0;i<2;i++){
        for(j=0;j<3;j++)
            printf("%d ",arr[i][j]);
        printf("\n");
    }
    return 0;
}
```

8번

사전평가	
문제	아래 내용을 C 로 구현해보도록 하자.
내용	3x^2 + 7x 를 1 ~ 2 까지 정적분하는 프로그램을 구현해보라. 3x^2 에서 ^2 는 제곱을 의미한다. 예로 x 에 1 이 들어가면 3x^2 = 9 가 된다.

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
double integral(double x1,double x2);
double func(double x);
int main(void){
    double x1 = 1,x2 = 2;
    double res;

    res = integral(x1,x2);

    printf("res = %.2lf\n",res);

    return 0;
}
double integral(double x1,double x2){
    double res = 0, x, area;
    int i, n = 10000;
    double dx = (x2 - x1)/n;
    x = x1;
    for(i=x1;x<x2;i++){
        area = func(x)*dx;
        res += area;
        x+=dx;
    }
    return res;
}
double func(double x){
    double f_x;
    f_x = 3*x*x+7*x;
    return f_x;
}
```

12번

문제	아래 질문에 대해 답하시오.
내용	void (* signal(int signum, void (* handler)(int)))(int)라는 signal 함수의 프로토타입을 기술하시오. 프로토타입을 기술하라는 의미는 반환형(리턴 타입)과 함수의 이름, 그리고 인자(파라메터)가 무엇인지 기술하라는 뜻임.

Ans)void (*signal(int signum,void (*handler)(int)))(int)

Return type : void (*)(int)
Name : signal
Parameter : int, void(*)(int)

14번

문제	아래 질문에 적절한 C 코드를 구현하시오.
내용	TI Cortex-R5F Safety MCU is very good to Real-Time System. 위의 문장에서 Safety MCU 가 몇 번째 부터 시작하는지 찾아내보자! (배열의 인덱스로 표기해도 상관없고, 실제 위치로 표기해도 상관없으며 둘 중 무엇인지 표기하시오)

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
#include <string.h>
int StrCmp(char* str1,char* str2);
int main(void){
    char* str1 = "TI Cortex-R5F Safety MCU is very good to Real-Time System.
";
    char* str2 = "Safety MCU";
    int index;
    index = StrCmp(str1,str2);

    printf("index = %d\n",index);


}
int StrCmp(char* str1,char* str2){
    int index, length, chk;
    length = strlen(str1) - strlen(str2);
    for(index = 0;index < length ;index++){
        chk = strncmp(str1+index,str2,strlen(str2));
        if(!chk)
            break;
    }
    return index;
}
```

16번

문제	아래 질문에 적절한 C 코드를 구현해보자.
내용	16 비트 ADC 를 가지고 있는 장치가 있다. 보드는 12 V 로 동작하고 있고 ADC 는 -5 ~ 5 V 로 동작한다. ADC 에서 읽은 값이 12677 일 때 이 신호를 전압으로 변환해보자! (실제 보드가 없으므로 단순히 C 로 구현하면 된다)

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
double dac(int minV,int maxV,int bit,int digit);
int main(void){
    int minV = -5, maxV = 5;
    int digit = 12677, bit = 16 ;
    double v;

    v = dac(minV,maxV,bit,digit);
    printf("v = %.4lf\n",v);


    return 0;
}
double dac(int minV,int maxV,int bit,int digit){
    double v=1;
    int maxDigit = (1<<(bit-2)) - 1;
    int minDigit = -1*(1<<(bit-2)) ;
    int baseDigit = (digit>=0)?maxDigit:minDigit;
    int baseV = (digit>=0)?maxV:minV;

    v = baseV *((double)digit / baseDigit);
    return v;
}
```

17번

문제	설명을 읽고 C 코드를 구현해보자.
내용	24 비트 DAC 장치가 있다. 이 장치는 -12 V ~ 12 V 로 동작하며 보드는 5 V 로 동작한다. DAC 에서 나온 전압이 9.7 V 일 때 어떤 디지털 신호를 입력 받은것인지 파악해보자!

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
int adc(int minV,int maxV,int bit,double v);
int main(void){
    int maxV = 12, minV = -12;
    int bit = 24;
    double v = 9.7;
    int digit;

    digit = adc(minV,maxV,bit,v);

    printf("res = %d\n",digit);

    return 0;
}
int adc(int minV,int maxV,int bit,double v){
    int maxDigit = (1<<(bit-1)) -1;
    int minDigit = -1*(1<<(bit-1));
    int baseV =v>0 ? maxV : minV;
    int baseDigit = v>=0 ? maxDigit:minDigit;
    int digit;

    digit = (v / baseV)*baseDigit;

    return digit;
}
```

19번

문제	아래 자료 구조를 C 언어로 구현해보시오.

내용	Stack 자료구조를 아래와 같은 포맷에 맞춰 구현해보시오.
	(힌트: 이중 포인터)
	
	ex)
	
	int main(void)
	{
	stack *top = NULL;
	push(&top, 1);
	push(&top, 2);
	printf("data = %d₩n", pop(&top));
	}

```c
void push(Stack** top,int data){
    Stack* tmp;
    tmp = *top;
    *top = get_node();
    (*top)->data = data;
    (*top)->p_node = tmp;
}
int pop(Stack** top){
    Stack* tmp;
    int num;
    tmp = *top;
    if(tmp==NULL){
        printf("Stack is EMPTY!\n");
        return 0;
    }
    num = tmp->data;
    *top = tmp->p_node;
    free(tmp);
    return num;
}
```

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
#include <malloc.h>
typedef struct stack{
    int data;
    struct stack *p_node;
}Stack;

Stack* get_node();
void push(Stack** top,int data);
int pop(Stack** top);

int main(void){
    Stack *top = NULL;

    push(&top,1);
    push(&top,2);

    printf("data = %d\n",pop(&top));

    return 0;
}
Stack* get_node(){
    Stack* tmp;
    tmp = (Stack*)malloc(sizeof(Stack*));
    tmp->data = 0;
    tmp->p_node = NULL;
    return tmp;
}
```

85번

문제	기본 for 문 활용 문제다.
내용	1 ~ 100 까지의 숫자중 홀수만 더해서 출력해보시오.

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
int oddSum(int start,int end);
int main(void){
    int start = 0, end = 100;
    int sum;

    sum = oddSum(start,end);

    printf("sum = %d\n",sum);
    return 0;
}
int oddSum(int start,int end){
    int i, sum = 0;

    for(i=start;i<=end;i++){
        if(i%2)
            sum+=i;
    }
    return sum;
}
```

86번

문제	기본 배열 문제다.
내용	1 ~ 100 까지 숫자를 모두 더해서 첫 번째 배열에 저장하고 1 ~ 100 까지 숫자중 홀수만 더해서 두 번째 배열에 저장하고 1 ~ 100 까지 숫자중 짝수만 더해서 세 번째 배열에 저장한다. 다음으로 1 ~ 100 까지 숫자중 3 의 배수만 더해서 네 번째 배열에 저장한다. 각 배열의 원소를 모두 더해서 결과값을 출력하시오.

howard@ubuntu: ~/HomeworkBackup/assessment

```c
#include <stdio.h>
int sum(int start,int end,int chk);
int main(void){
    int start = 1, end = 100, i = 0;
    int res[4];
    int res2=0;
    for(i=0;i<sizeof(res)/sizeof(int);i++){
        res[i]=sum(start,end,i);
        res2+=res[i];
    }
    printf("res = %d\n",res2);



    return 0;
}
```

86번

문제	기본 배열 문제다.
내용	1 ~ 100 까지 숫자를 모두 더해서 첫 번째 배열에 저장하고 1 ~ 100 까지 숫자중 홀수만 더해서 두 번째 배열에 저장하고 1 ~ 100 까지 숫자중 짝수만 더해서 세 번째 배열에 저장한다. 다음으로 1 ~ 100 까지 숫자중 3 의 배수만 더해서 네 번째 배열에 저장한다. 각 배열의 원소를 모두 더해서 결과값을 출력하시오.

```
20 //chk , 0 : sum all, 1 : sum odd number, 2 : sum even number
21 //3 : sum multiples of 3
22 int sum(int start,int end,int chk){
23     int sum = 0;
24     int i;
25 
26     for(i = start; i<=end;i++){
27         switch(chk){
28             case 0:
29                 sum+=i;
30                 break;
31             case 1:
32                 if(i%2)
33                     sum+=i;
34                 break;
35             case 2:
36                 if(!(i%2))
37                     sum+=i;
38                 break;
39             case 3:
40                 if(!(i%3))
41                     sum+=i;
42             default :
43                 break;
44         }
45     }
46 
47     return sum;
48 }
```

Total Score : 13 / 91