TI DSP,Xilinx zynq FPGA,MCU 및 Xilinx zynq FPGA 프로그래밍 전문가 과정

강사-INNOVA LEE(이상훈)

Gccompil3r@gmail.com

학생-윤지완

Yoonjw7894@naver.com

오늘은 저번에 했던 2 진트리를 재귀함수를 쓰지않고 돌리는 프로그래밍이다.

```
Void non_recur_tree_ins(tree **root, int data)
{
 tree **tmp = root;

while(*tmp)
 {
if((*tmp)->data > data)
 tmp = &(*tmp)->left;
elseif((*tmp)->data < data)
 tmp = &(*tmp)->right;
 }

 *tmp = get_tree_node();
 (*tmp)->data = data;
}
```

처음 이부분은 값을 넣는 프로그래밍이다.처음에는 똑깥지만 다른점은 재귀함수를 풀었기때문에 다시 자기자신을 호출을 할 수가없어서 while 무한 루프를 돌려서 각 값을 넣고 *tmp = get_tree_node(); 동적메모리 할당, 이 부분은 상당히 비슷하다. delete 도 그 전 2 진트리와 프로그래밍 형태는 거진 비슷하다 .하지만 push 나 pop 을 이용해 값을 넣고 빼는것이 많이 어려웠다.

처음에

```
Bool stack_is_not_empty(stack *top)
{
if(top != NULL)
returntrue;
else
Return false;
```

}

이 명령어를 이용해 STACK TOP에 공간이 있는지 없는지 확인 하는것이다. 이것이 while문을 돌아가게 하는 명령어다 .이제 공간이 있는것을 확인하고

```
Void print_tree(tree **root)
{
 tree **tmp = root;
 stack *top = NULL;


push(&top, *tmp);
```

함수를 선언하고 이중 포인터 tmp를 선언하고 push를 이용해 메모리에 값을 넣는다.

```
voidpush(stack **top, void *data)
{
if(data == NULL)
return;


 stack *tmp = *top;
 *top = get_stack_node();
 (*top)->data = malloc(sizeof(void *));
 (*top)->data = data;
 (*top)->link = tmp;
}
```

data가 “0”이면 리턴되는 값은 없으며 *top에 동적 메모리할당하고 그 다음이 모양이 지금까지 쓰던거랑은 모양이 다른데 이것은 data값에 모든 값을 적용하겠다는 것이다.그 아래는 처음에 stack에 값넣을때랑 같다. push가 되면 다시 돌아와서

```
while(stack_is_not_empty(top))
 {
 tree *t = (tree *)pop(&top);
 tmp = &t;


printf("data = %d, ", (*tmp)->data);
```

```
if((*tmp)->left)
printf("left = %d, ", (*tmp)->left->data);
else
printf("left = NULL, ");


if((*tmp)->right)
printf("right = %d₩n", (*tmp)->right->data);
else
printf("right = NULL₩n");


push(&top, (*tmp)->right);
push(&top, (*tmp)->left);


//tmp = &(*tmp)->left;


//*tmp = (tree *)pop(&top);
```

Printf_tree 함수의 while문이 실행이 된다. 포인터 t를 만들고 만듬과 동시에 pop의 주소를 t에 넣어주고 t의 주소값을 tmp로 넣어준다.pop의 프로그램은

```
void *pop(stack **top)
{
 stack *tmp = *top;
void *data = NULL;


if(*top == NULL)
 {
printf("stack is empty!₩n");
```

```
returnNULL;
 }


 data = (*top)->data;
 *top = (*top)->link;
free(tmp);




return data;
}
```

pop은 if 문을 통해서 안에 다 비었으면 stack is empty라고 문구가 나온다 그 전까지는 top에 값을 tmp에 넣고 t에 처음 tmp에 주소를 넣고 push 함수를 호출한다.이때push(&top, (*tmp)->right);을 통해서 3000주소에 100을 넣어준고 그 다음 다시 pop에 들어가서 3000주소에 100값을 반환하고 그 주소는 해제한다.다시 포인터 t에 주소를 tmp에 넣어준다 이 순서를 계속 반복하다가 더이상 뺄께없으면 empty 문구가 나온다.