2018920065 루안리치 컴퓨터알고리즘 과제#2

```
//
//    2018920065 luan li chi
//

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define MAX_ELEMENT 100

//heap
typedef struct{
        int heap[MAX_ELEMENT];
        int heap_size;
}HeapType;

void initHeap(HeapType* h){
        h->heap_size=0;
}

void upHeap(HeapType* h){
        int i = h->heap_size;
        int key = h->heap[i];

        while((i != 1) && (key < h->heap[i/2])){
                h->heap[i] = h->heap[i/2];
                i /= 2;
        }
        h->heap[i] = key;
}

void insertItem(HeapType* h, int key){
        h->heap_size += 1;
        h->heap[h->heap_size] = key;
        upHeap(h);
}
```

```
void printHeap(HeapType* h){
	for(int i=1; i<=h->heap_size; i++){
		printf("[%d]", h->heap[i]);
	}
	printf("\n");
}

//stack
typedef struct{
	int stack[MAX_ELEMENT];
	int st_top;
}StackType;

void initStack(StackType* s){
	s->st_top=-1;
}

int isEmpty(StackType* s){
	if(s->st_top == -1){
		return 1;
	}
	else return 0;
}

void push(StackType* s, int key){
	s->st_top += 1;
	s->stack[s->st_top] = key;
}

int pop(StackType* s){
	if(isEmpty(s) == 1){
		printf("empty.\n");
		return -1;
	}
	else return s->stack[s->st_top--];
}
```

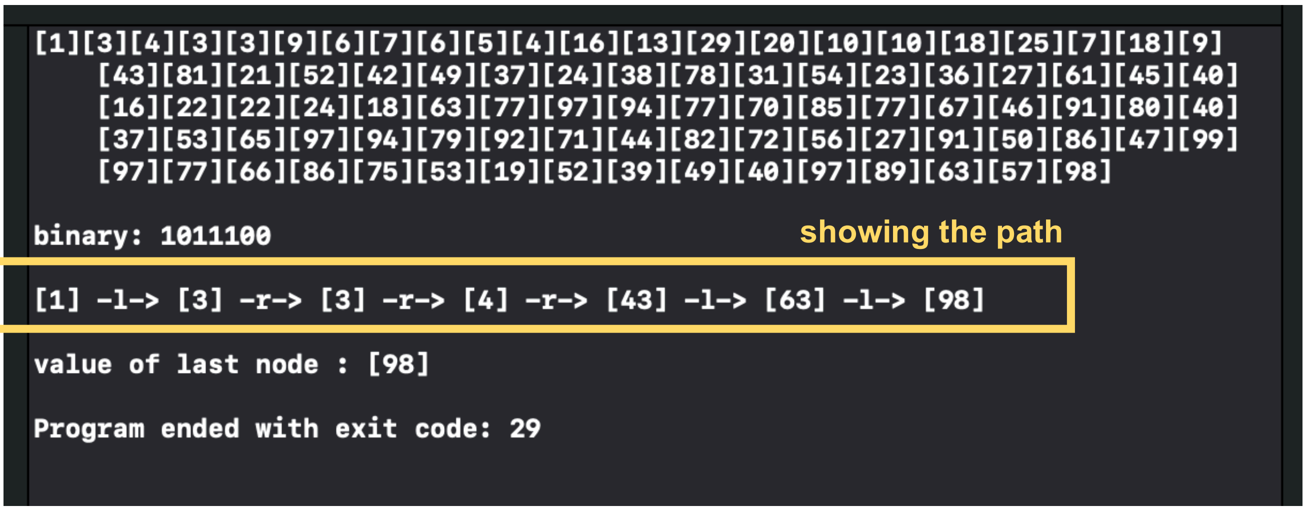
```
void printStack(StackType* s){
    printf("\nbinary: ");
    for(int i=s->st_top; i>=0; i--){
        printf("%d",s->stack[i]);
    }
    printf("\n\n");
}

void binaryExpansion(int i, StackType* s){
    while(i >= 2){
        push(s, i%2);
        i=i/2;
    }
    push(s, i);
    printStack(s);
}

int findLastNode(HeapType* h){
    int bit, value=0;
    int n = h->heap_size;     //n
    int i=1;      //root of the heap, heap[1]
    StackType stack;      //stack 생성
    initStack(&stack);  //stack 초기화
    binaryExpansion(n, &stack);      //2진수로 변환하기
    pop(&stack);     //첫 숫자를 제거하기
    printf("[%d] ",h->heap[i]);
    while(isEmpty(&stack) == 0){
        bit=pop(&stack);
        if(bit==0){
            i=i*2;        //leftChild
            value = h->heap[i];
            printf("-l-> [%d] ",value);      //show the path
        }
        else{
            i=i*2+1;      //rightChild
            value = h->heap[i];
            printf("-r-> [%d] ",value);
        }
    }
    return value;
}


void main(){
    HeapType heap;
    int num, size;
    initHeap(&heap);     //힙 초기화
    srand(time(NULL));
    size=(rand()%100)+2;
    for(int i=1;i<size; i++){    //랜덤으로 힙 생성
        num=(rand()%100)+1;
        insertItem(&heap,num);
    }
    printHeap(&heap);
    printf("\n\nvalue of last node : [%d]\n\n",findLastNode(&heap));
}
```

Simulation results:



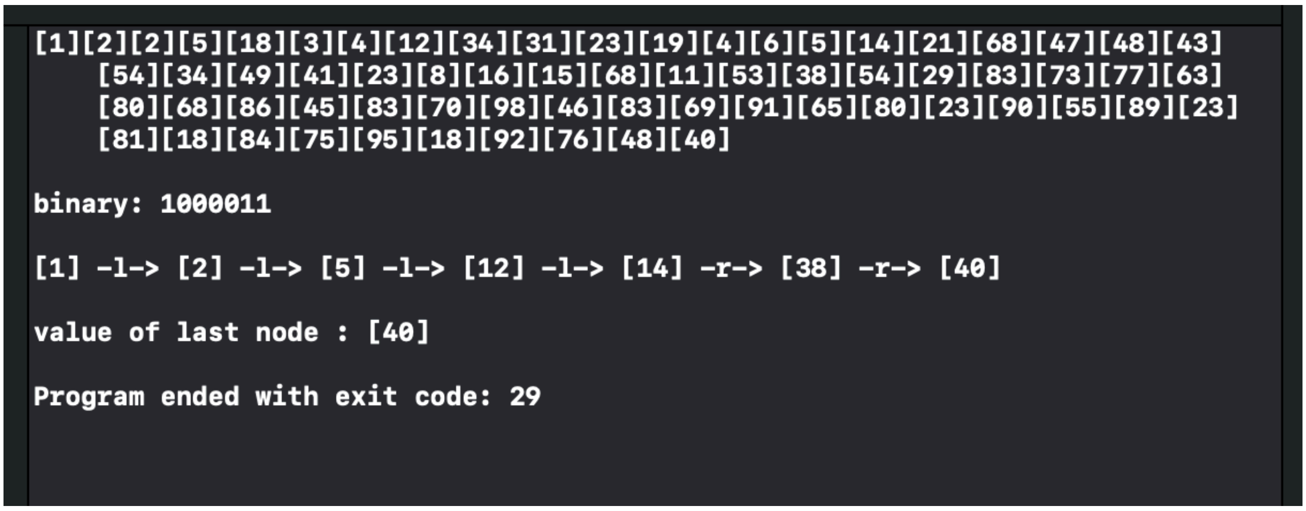
```
[1][3][4][3][3][9][6][7][6][5][4][16][13][29][20][10][10][18][25][7][18][9]
    [43][81][21][52][42][49][37][24][38][78][31][54][23][36][27][61][45][40]
    [16][22][22][24][18][63][77][97][94][77][70][85][77][67][46][91][80][40]
    [37][53][65][97][94][79][92][71][44][82][72][56][27][91][50][86][47][99]
    [97][77][66][86][75][53][19][52][39][49][40][97][89][63][57][98]

binary: 1011100

[1] -l-> [3] -r-> [3] -r-> [4] -r-> [43] -l-> [63] -l-> [98]

value of last node : [98]

Program ended with exit code: 29
```



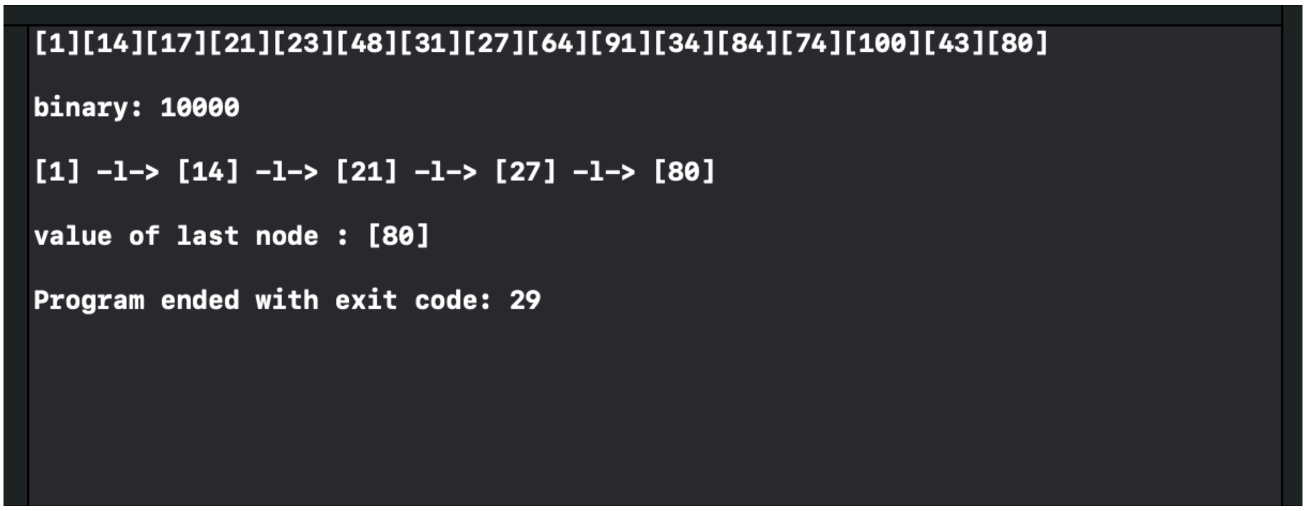
```
[1][2][2][5][18][3][4][12][34][31][23][19][4][6][5][14][21][68][47][48][43]
    [54][34][49][41][23][8][16][15][68][11][53][38][54][29][83][73][77][63]
    [80][68][86][45][83][70][98][46][83][69][91][65][80][23][90][55][89][23]
    [81][18][84][75][95][18][92][76][48][40]

binary: 1000011

[1] -l-> [2] -l-> [5] -l-> [12] -l-> [14] -r-> [38] -r-> [40]

value of last node : [40]

Program ended with exit code: 29
```



```
[1][14][17][21][23][48][31][27][64][91][34][84][74][100][43][80]

binary: 10000

[1] -l-> [14] -l-> [21] -l-> [27] -l-> [80]

value of last node : [80]

Program ended with exit code: 29
```