

2018920065 루안리치

컴퓨터알고리즘 과제#1

HW 1-1(video 4-2) -> p.1~p.4

HW 1-2(video 4-5) -> p.5~p.6

HW 1-3(video 4-5) -> p.7~p.8

HW 1-2(video 4-5) -> p.9~p.11

// 2018920065 luan lichi

// algorithms hw 1-1

#include <stdio.h>

#include <stdlib.h>

```
typedef struct ListNode{
    int data;
    struct ListNode* link;
}ListNode;
```

```
typedef struct{
    ListNode* head;
}LinkedListType;
```

```
void init(LinkedListType* L){
    L->head = NULL;
}
```

```
void addFirst(LinkedListType* L, int item){
    ListNode* node = (ListNode*)malloc(sizeof(ListNode));
    node->data = item;
    node->link = L->head;
    L->head = node;
}
```

```

void add(LinkedListType* L, int pos, int item){
    ListNode* node = (ListNode*)malloc(sizeof(ListNode));
    ListNode* before = L->head;
    for(int i=0; i<pos-1; i++){
        before = before->link;
    }
    node->data = item;
    node->link = before->link;
    before->link = node;
}

```

//addLast function

```

void addLast(LinkedListType* L, int item){
    ListNode* node = (ListNode*)malloc(sizeof(ListNode));
    ListNode* last = L->head;
    for(ListNode* p = L->head; p != NULL; p = p->link){
        last = p;
    }
    node->data = item;
    node->link = NULL;
    last->link = node;
}

```

//removeNode fuction

```

void removeNode(LinkedListType* L, int pos){
    ListNode* before = L->head;
    ListNode* target = before->link;
    if(pos==1){ //remove head
        L->head = target;
        free(before);
    }
    else{
        for(int i=1; i<pos-1; i++){
            before = before->link;
        }
        target = before->link;
        if(target->link==NULL){ //remove tail
            before->link = NULL;
        }
    }
}

```

```

        free(target);
    }
    else{
        before->link = target->link;
        free(target);
    }
}
}

```

```

int get(LinkedListType* L, int pos){
    ListNode* p = L->head;
    for(int i=1; i<pos; i++){
        p = p->link;
    }
    return p->data;
}

```

```

void set(LinkedListType* L, int pos, int item){
    ListNode* p = L->head;
    for(int i=1; i<pos; i++){
        p = p->link;
    }
    p->data = item;
}

```

```

void printlist(LinkedListType* L){
    for(ListNode* p = L->head; p != NULL; p = p->link){
        printf("[%d] -> ", p->data);
    }
    printf("NULL\n");
}

```

```

void main(){

    LinkedListType list;
    init(&list);

    addFirst(&list, 10); printlist(&list);
}

```

```

addFirst(&list, 20); printlist(&list);
addFirst(&list, 30); printlist(&list);
add(&list, 2, 40); printlist(&list);
add(&list, 4, 50); printlist(&list);
addLast(&list, 60); printlist(&list);
addLast(&list, 70); printlist(&list);
/*int pos;
printf("\nEnter Number?\n");
scanf("%d",&pos);
printf("no. %d is %d\n", pos, get(&list, pos));*/

```

```

removeNode(&list, 3); printlist(&list);
removeNode(&list, 1); printlist(&list);
removeNode(&list, 5); printlist(&list);

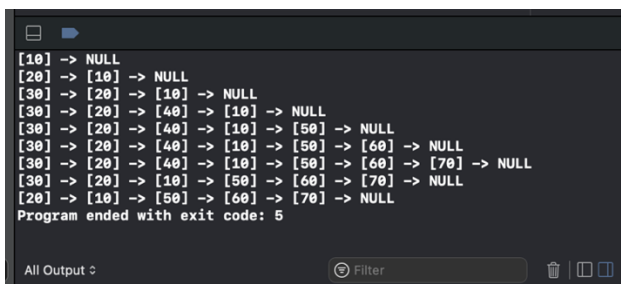
```

```

}

```

Simulation result :



```

[10] -> NULL
[20] -> [10] -> NULL
[30] -> [20] -> [10] -> NULL
[30] -> [20] -> [40] -> [10] -> NULL
[30] -> [20] -> [40] -> [10] -> [50] -> NULL
[30] -> [20] -> [40] -> [10] -> [50] -> [60] -> NULL
[30] -> [20] -> [40] -> [10] -> [50] -> [60] -> [70] -> NULL
[30] -> [20] -> [10] -> [50] -> [60] -> [70] -> NULL
[20] -> [10] -> [50] -> [60] -> [70] -> NULL
Program ended with exit code: 5

```

```
// 2018920065 luan lich
```

```
// algorithms hw 1-2
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
void buildList(int S[], int n){
```

```
    for(int i=0;i<n;i++){
```

```
        S[i] = i+1;
```

```
    }
```

```
}
```

```
int runSimulation1(int S[], int n, int k){
```

```
    int r=0;
```

```
    int size = n;
```

```
    int i;
```

```
    while(n>1){
```

```
        i=0;
```

```
        while(i<k){
```

```
            r = (r+1)%size;
```

```
            if(S[r] != 0){
```

```
                i = i+1;
```

```
            }
```

```
        }
```

```
        S[r]=0;
```

```
        n--;
```

```
        while(S[r]==0){
```

```
            r = (r+1)%size;
```

```
        }
```

```
    }
```

```
    return S[r];
```

```
}
```

```
void main(){
```

```
    int size, k;
```

```
    printf("insert size , key\n");
```

```
    scanf("%d %d",&size,&k);
```

```
int candle[size];

buildList(candle, size);

printf("candle no. %d\n", runSimulation1(candle, size, k));

}
```

Simulation results :

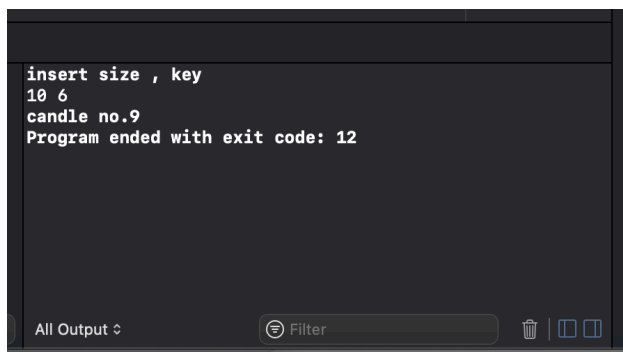


```
insert size , key
7 3
candle no.2
Program ended with exit code: 12
```

This screenshot shows a terminal window with a dark background. The output text is as follows:

```
insert size , key
7 3
candle no.2
Program ended with exit code: 12
```

At the bottom of the terminal, there is a status bar with the text "All Output" and a "Filter" button.



```
insert size , key
10 6
candle no.9
Program ended with exit code: 12
```

This screenshot shows a terminal window with a dark background. The output text is as follows:

```
insert size , key
10 6
candle no.9
Program ended with exit code: 12
```

At the bottom of the terminal, there is a status bar with the text "All Output" and a "Filter" button.

```
// 2018920065 luan lich
```

```
// algorithms hw 1-3
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
void buildList(int S[], int n){  
    for(int i=0;i<n;i++){  
        S[i] = i+1;  
    }  
}
```

```
void removeCandle(int S[], int r, int size){  
    for(int i=r;i<size;i++){  
        S[i]=S[i+1];  
    }  
}
```

```
int runSimulation1(int S[], int n, int k){  
    int r=0;  
    while(n>1){  
        r=(r+k)%n;  
        removeCandle(S,r,n);  
        n--;  
    }  
    /*for(int j=0;j<n;j++){  
        printf("%d -> ",S[j]);}*/  
    return S[0];  
}
```

```
void main(){  
    int size, k;  
    printf("insert size , key\n");  
    scanf("%d %d",&size,&k);  
    int candle[size];  
  
    buildList(candle, size);
```

```
printf("candle no. %d\n", runSimulation1(candle, size, k));  
  
}
```

Simulation results :

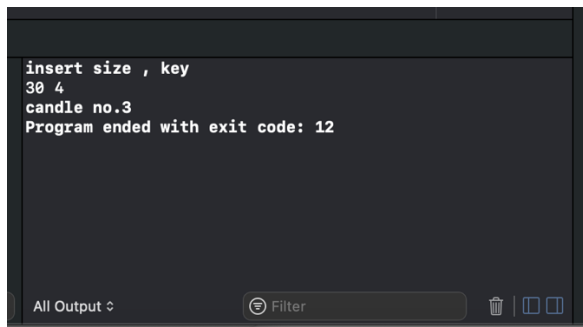


```
insert size , key  
7 3  
candle no.2  
Program ended with exit code: 12
```

This screenshot shows a terminal window with a dark background. The output text is as follows:

```
insert size , key  
7 3  
candle no.2  
Program ended with exit code: 12
```

At the bottom of the terminal, there is a status bar containing the text "All Output", a "Filter" input field, and icons for deleting, copying, and pasting.



```
insert size , key  
30 4  
candle no.3  
Program ended with exit code: 12
```

This screenshot shows a terminal window with a dark background. The output text is as follows:

```
insert size , key  
30 4  
candle no.3  
Program ended with exit code: 12
```

At the bottom of the terminal, there is a status bar containing the text "All Output", a "Filter" input field, and icons for deleting, copying, and pasting.


```
// 2018920065 luan lichi
```

```
// algorithms hw 1-4
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
typedef struct ListNode{  
    int data;  
    struct ListNode* link;  
}ListNode;
```

```
typedef struct{  
    ListNode* head;  
}LinkedListType;
```

```
void init(LinkedListType* L){  
    L->head = NULL;  
}
```

```
void buildList(LinkedListType* L, int size){  
    //head  
    ListNode* head = (ListNode*)malloc(sizeof(ListNode));  
    head->data = 1;  
    L->head = head;  
    ListNode* before = head;  
  
    //body  
    for(int i=2;i<=size;i++){  
        ListNode* node = (ListNode*)malloc(sizeof(ListNode));  
        node->data = i;  
        before->link = node;  
        before = node;  
    }  
  
    //tail points to head  
    before->link = L->head;  
}
```

```

/*void printList(LinkedListType* L, int size){
    ListNode* p = L->head;
    for(int i=0; i<size; i++){
        printf("[%d] -> ", p->data);
        p = p->link;
    }
    printf("\n");
}*/

```

//for checking the function buildlist

```

int runSimulation1(LinkedListType* L, int n, int k){
    ListNode* p = L->head;
    ListNode* pnext;
    while(p != p->link){
        for(int i=1; i<k; i++){
            p = p->link;
        }
        pnext = p->link;
        p->link = p->link->link;
        free(pnext);
        p = p->link;
    }
    return p->data;
}

```

```

void main(){

```

```

    int size, k;
    printf("insert size , key\n");
    scanf("%d %d",&size,&k);

```

```

    LinkedListType list;
    init(&list);
    buildList(&list, size);
    /*printList(&list, size);*/

```

```

    printf("candle no.%d\n",runSimulation1(&list, size, k));

```

}

Simulation results :

```
insert size , key
7 3
candle no.2
Program ended with exit code: 12
```

All Output ⌵ Filter

```
insert size , key
50 9
candle no.36
Program ended with exit code: 13
```

All Output ⌵ Filter