HANYANG UNIVERSITY
DIVISION OF COMPUTER SCIENCE

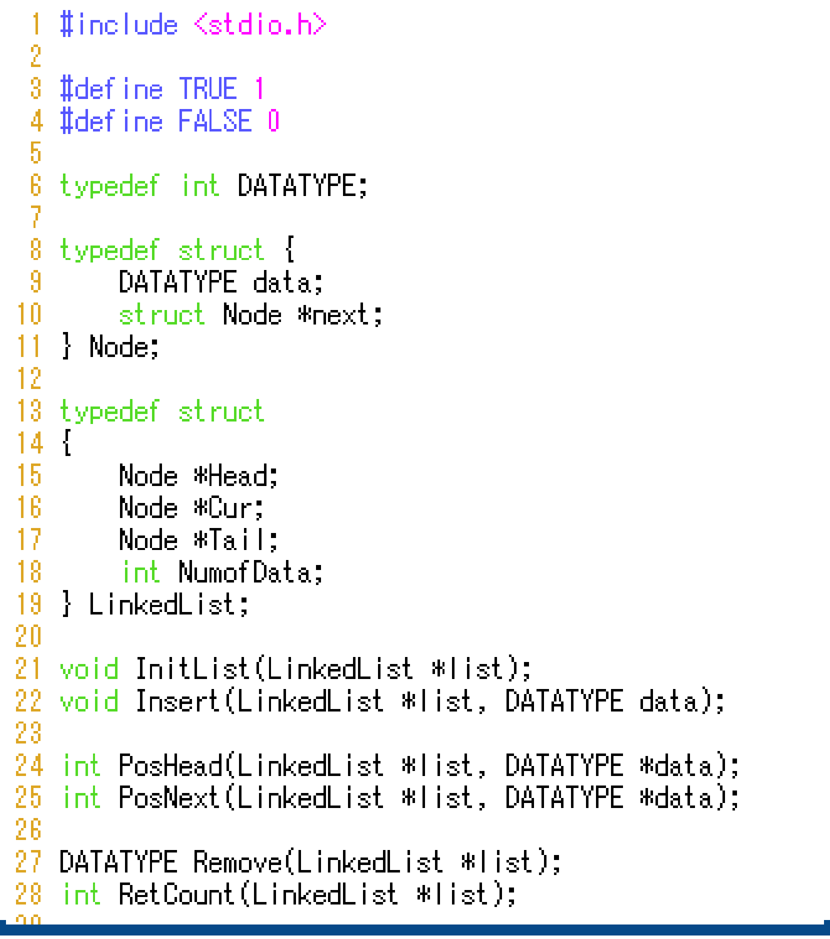
Data Structures

박영준 교수님

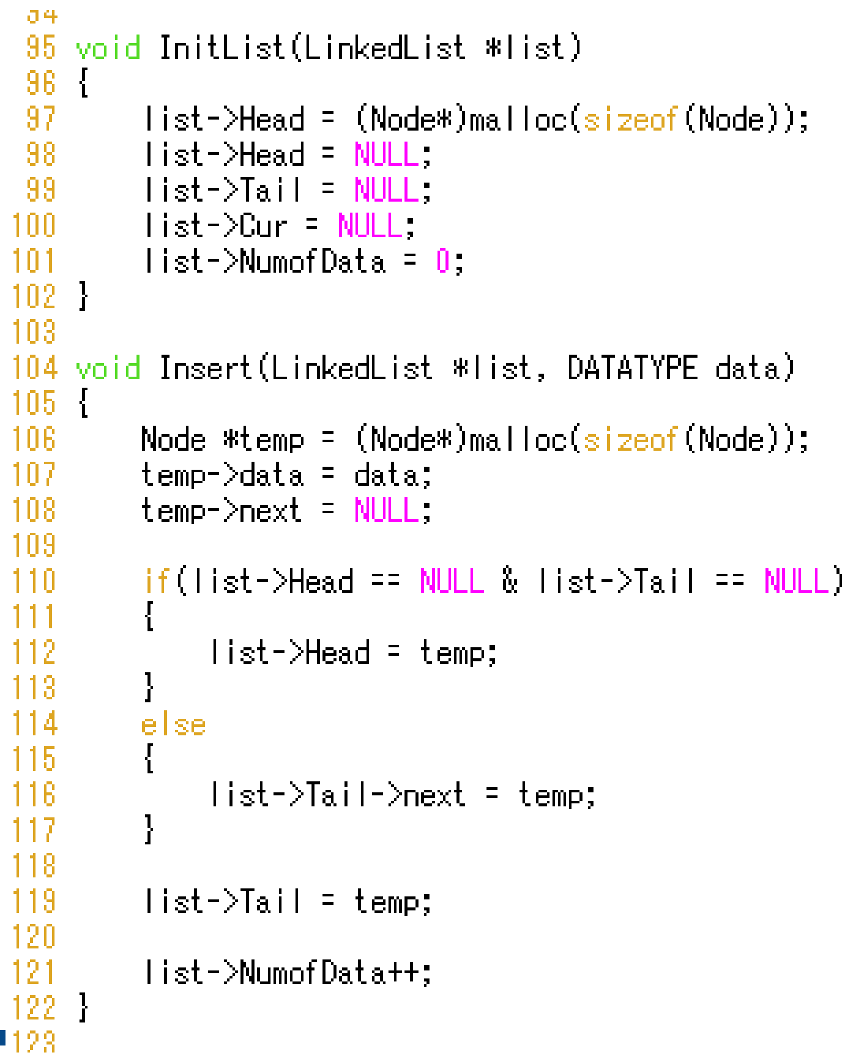
Lab2:LinkedList fixed

Linked List

Prev



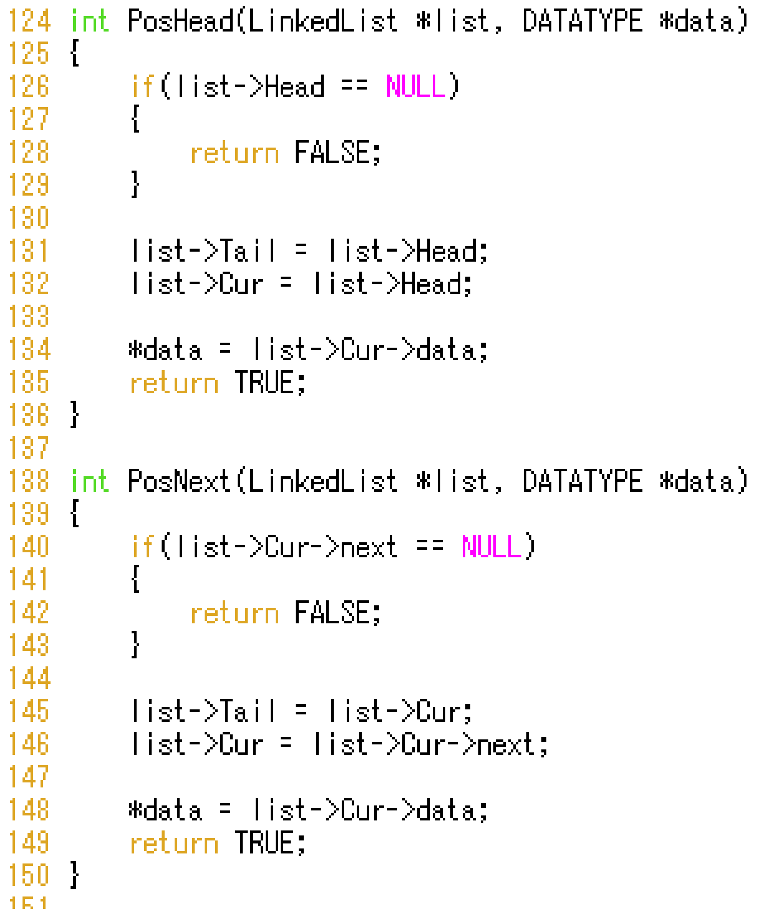
```c
#include <stdio.h>

#define TRUE 1
#define FALSE 0

typedef int DATATYPE;

typedef struct {
    DATATYPE data;
    struct Node *next;
} Node;

typedef struct
{
    Node *Head;
    Node *Cur;
    Node *Tail;
    int NumofData;
} LinkedList;

void InitList(LinkedList *list);
void Insert(LinkedList *list, DATATYPE data);

int PosHead(LinkedList *list, DATATYPE *data);
int PosNext(LinkedList *list, DATATYPE *data);

DATATYPE Remove(LinkedList *list);
int RetCount(LinkedList *list);
```



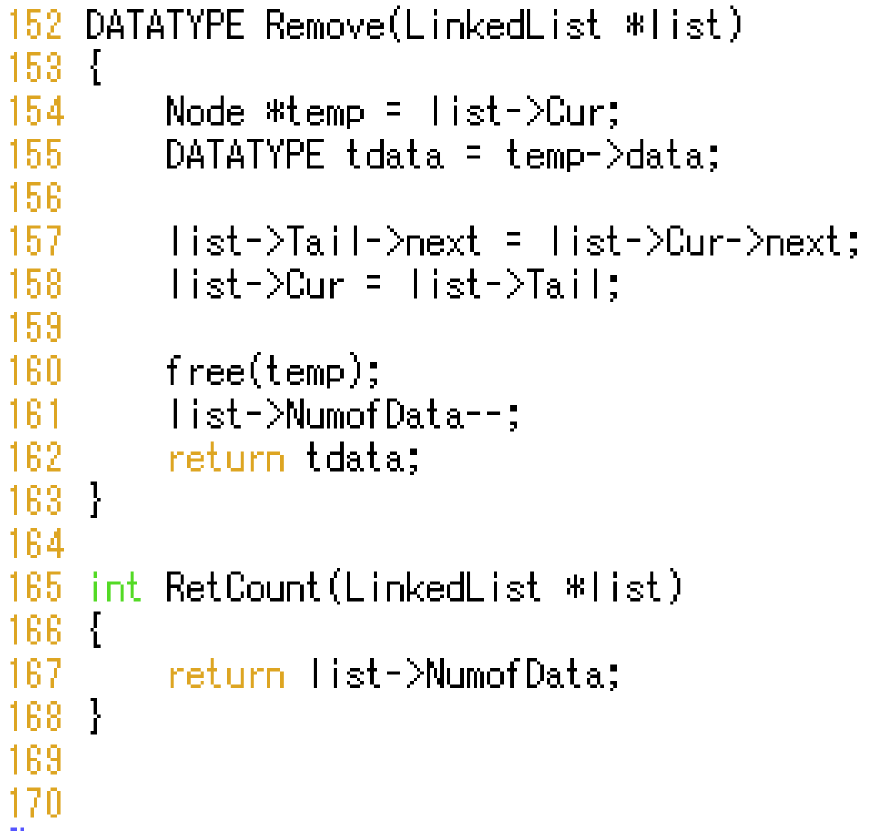
```c
void InitList(LinkedList *list)
{
    list->Head = (Node*)malloc(sizeof(Node));
    list->Head = NULL;
    list->Tail = NULL;
    list->Cur = NULL;
    list->NumofData = 0;
}

void Insert(LinkedList *list, DATATYPE data)
{
    Node *temp = (Node*)malloc(sizeof(Node));
    temp->data = data;
    temp->next = NULL;

    if(list->Head == NULL & list->Tail == NULL)
    {
        list->Head = temp;
    }
    else
    {
        list->Tail->next = temp;
    }

    list->Tail = temp;

    list->NumofData++;
}
```

Linked List

Prev



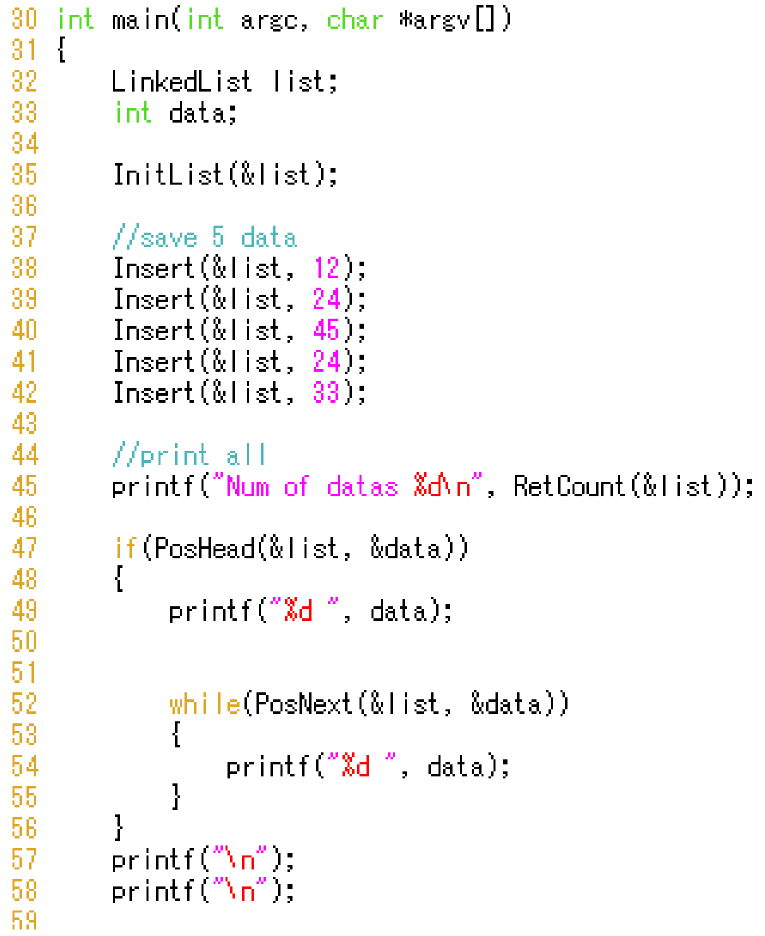
```c
int PosHead(LinkedList *list, DATATYPE *data)
{
    if(list->Head == NULL)
    {
        return FALSE;
    }

    list->Tail = list->Head;
    list->Cur = list->Head;

    *data = list->Cur->data;
    return TRUE;
}

int PosNext(LinkedList *list, DATATYPE *data)
{
    if(list->Cur->next == NULL)
    {
        return FALSE;
    }

    list->Tail = list->Cur;
    list->Cur = list->Cur->next;

    *data = list->Cur->data;
    return TRUE;
}
```



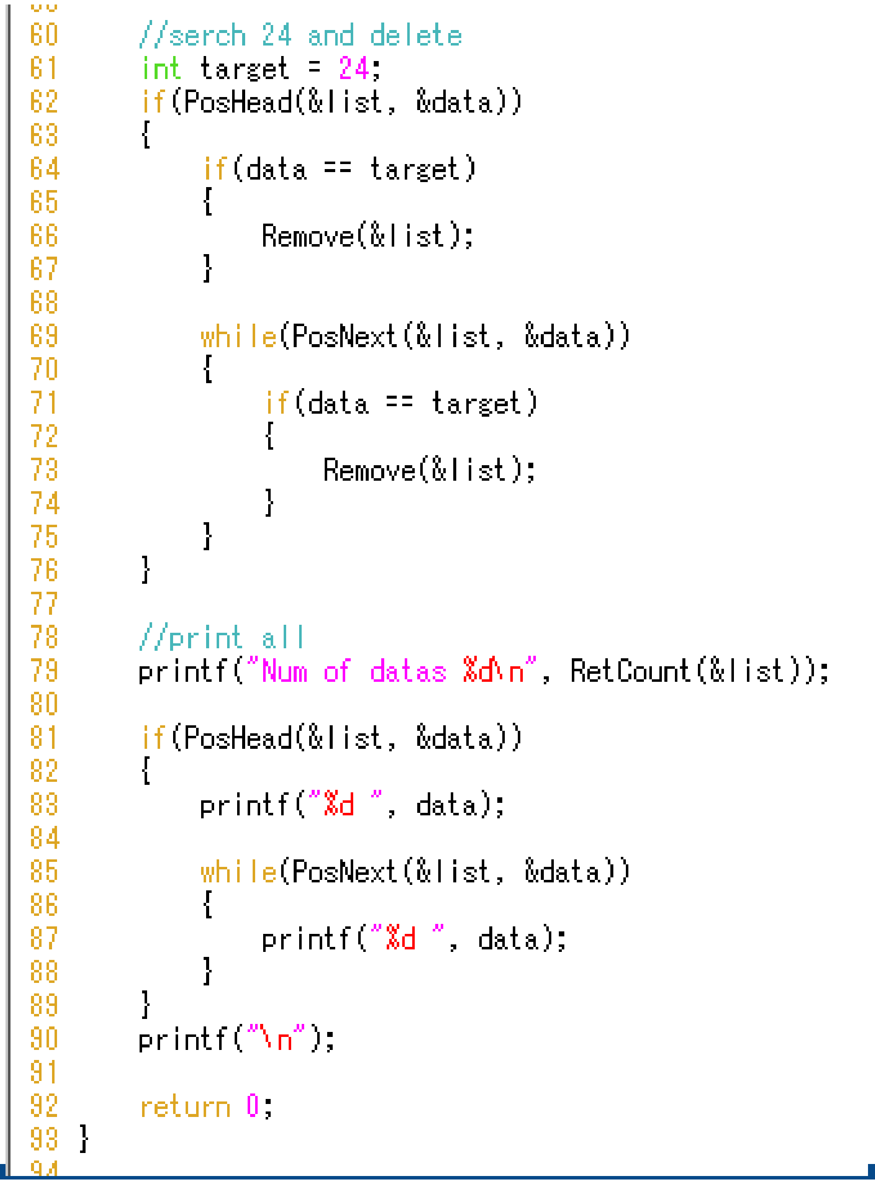
```c
DATATYPE Remove(LinkedList *list)
{
    Node *temp = list->Cur;
    DATATYPE tdata = temp->data;

    list->Tail->next = list->Cur->next;
    list->Cur = list->Tail;

    free(temp);
    list->NumofData--;
    return tdata;
}

int RetCount(LinkedList *list)
{
    return list->NumofData;
}
```

Linked List

Prev



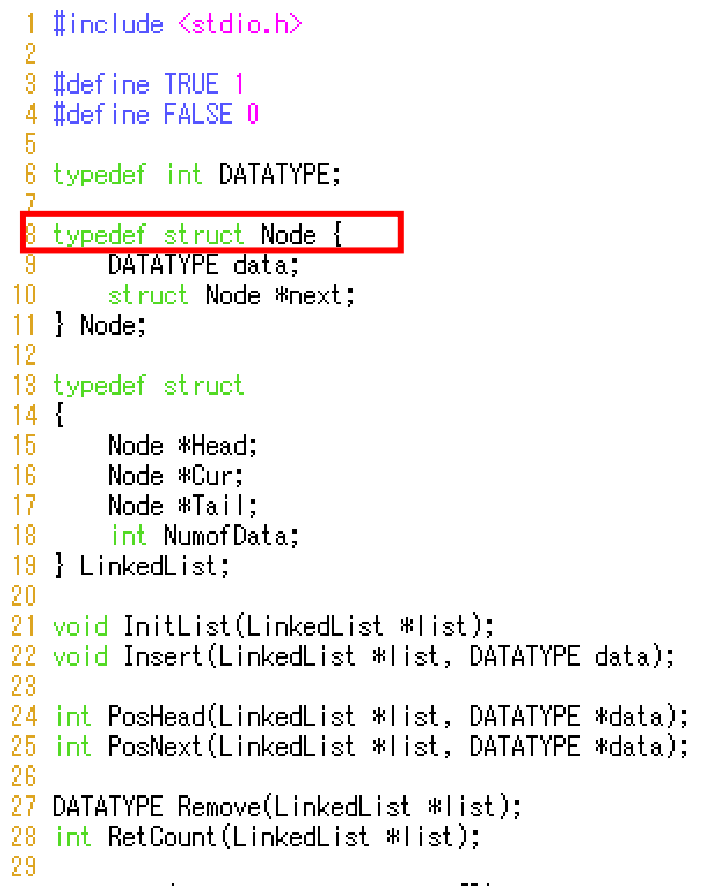
```c
int main(int argc, char *argv[])
{
    LinkedList list;
    int data;

    InitList(&list);

    //save 5 data
    Insert(&list, 12);
    Insert(&list, 24);
    Insert(&list, 45);
    Insert(&list, 24);
    Insert(&list, 33);

    //print all
    printf("Num of datas %d\n", RetCount(&list));

    if(PosHead(&list, &data))
    {
        printf("%d ", data);


        while(PosNext(&list, &data))
        {
            printf("%d ", data);
        }
    }
    printf("\n");
    printf("\n");
```



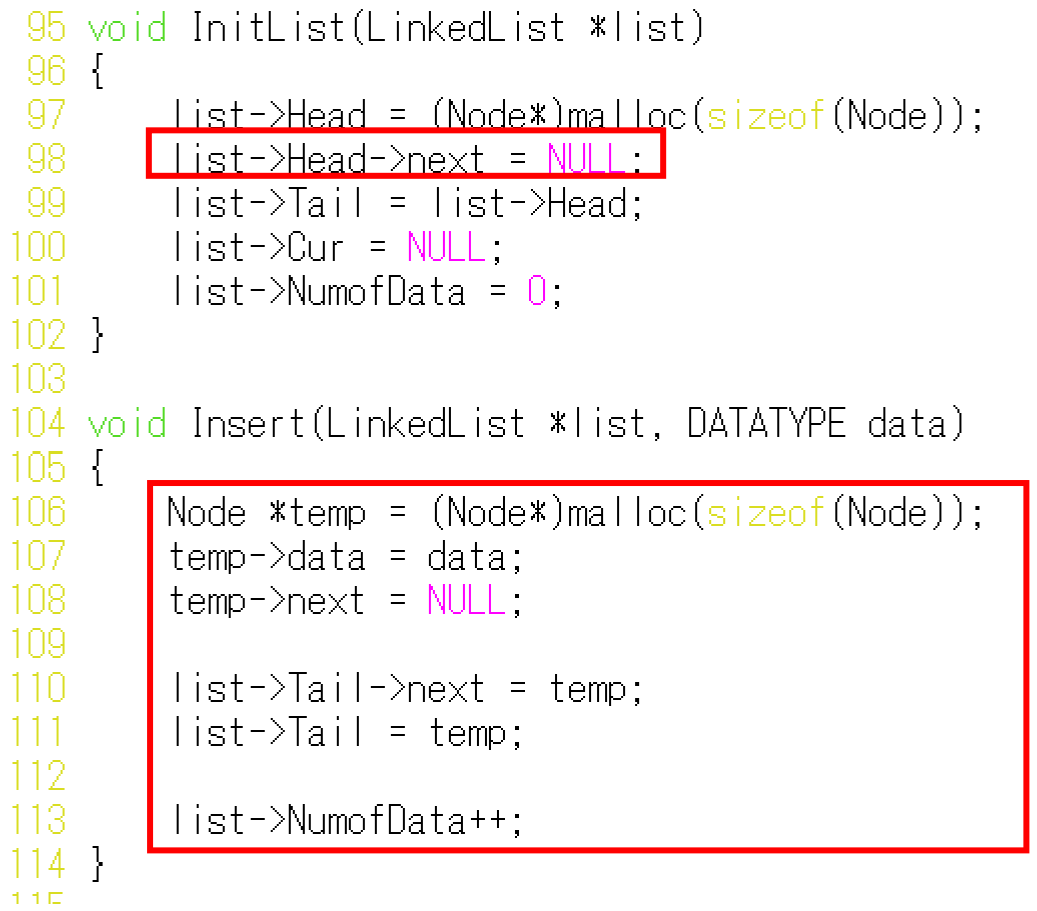
```c
    //serch 24 and delete
    int target = 24;
    if(PosHead(&list, &data))
    {
        if(data == target)
        {
            Remove(&list);
        }

        while(PosNext(&list, &data))
        {
            if(data == target)
            {
                Remove(&list);
            }
        }
    }

    //print all
    printf("Num of datas %d\n", RetCount(&list));

    if(PosHead(&list, &data))
    {
        printf("%d ", data);

        while(PosNext(&list, &data))
        {
            printf("%d ", data);
        }
    }
    printf("\n");

    return 0;
}
```

Linked List

Fixed



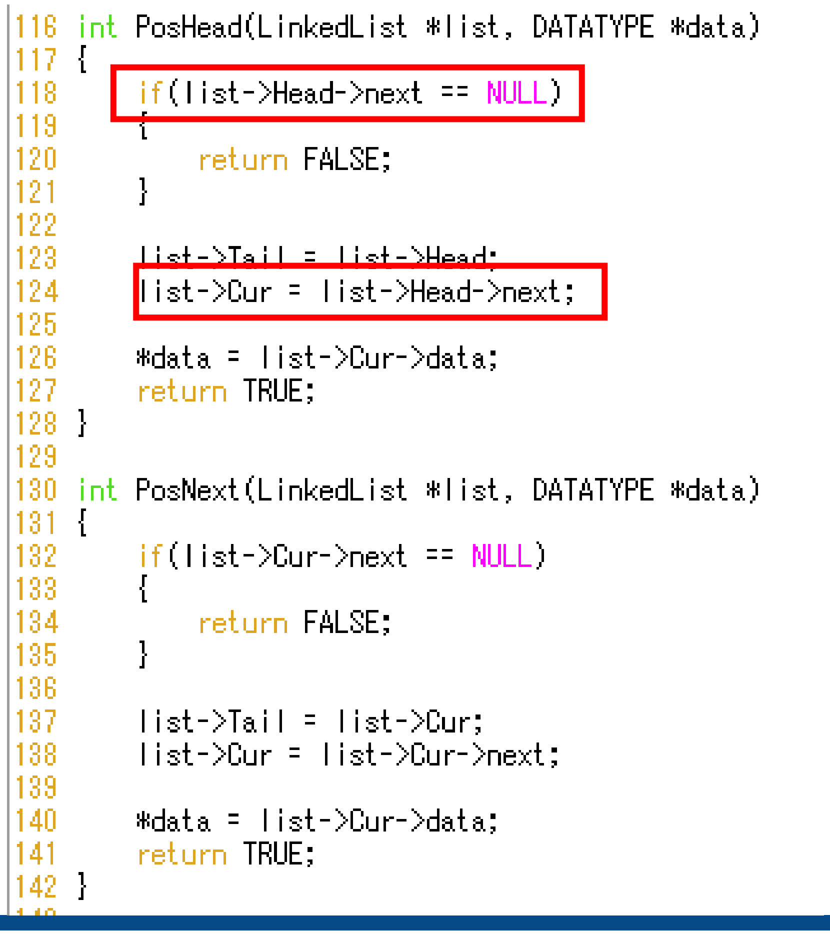
```c
#include <stdio.h>

#define TRUE 1
#define FALSE 0

typedef int DATATYPE;

typedef struct Node {
	DATATYPE data;
	struct Node *next;
} Node;

typedef struct
{
	Node *Head;
	Node *Cur;
	Node *Tail;
	int NumofData;
} LinkedList;

void InitList(LinkedList *list);
void Insert(LinkedList *list, DATATYPE data);

int PosHead(LinkedList *list, DATATYPE *data);
int PosNext(LinkedList *list, DATATYPE *data);

DATATYPE Remove(LinkedList *list);
int RetCount(LinkedList *list);
```



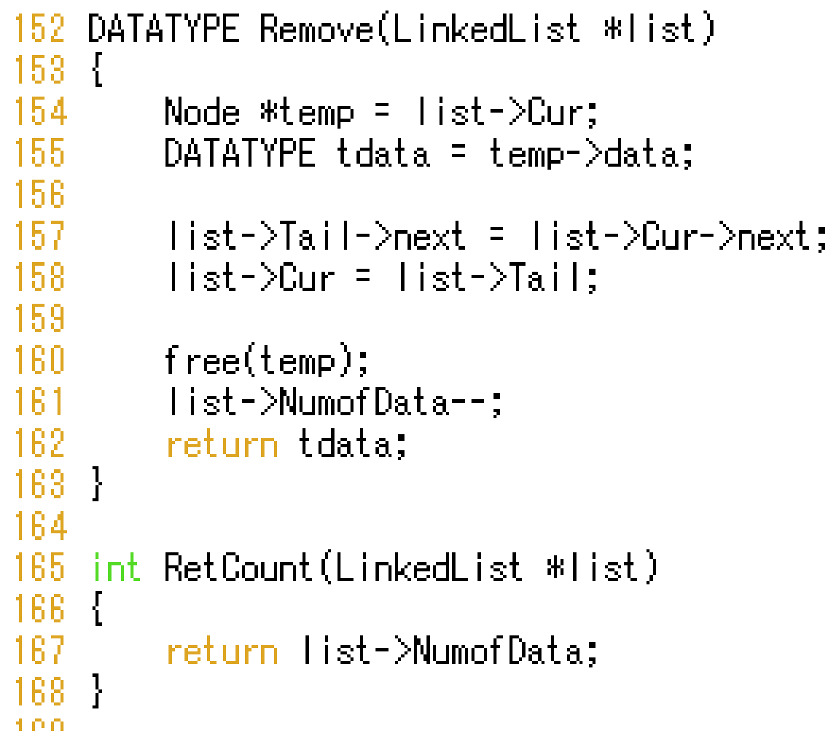
```c
void InitList(LinkedList *list)
{
	list->Head = (Node*)malloc(sizeof(Node));
	list->Head->next = NULL;
	list->Tail = list->Head;
	list->Cur = NULL;
	list->NumofData = 0;
}

void Insert(LinkedList *list, DATATYPE data)
{
	Node *temp = (Node*)malloc(sizeof(Node));
	temp->data = data;
	temp->next = NULL;

	list->Tail->next = temp;
	list->Tail = temp;

	list->NumofData++;
}
```

Linked List

Fixed



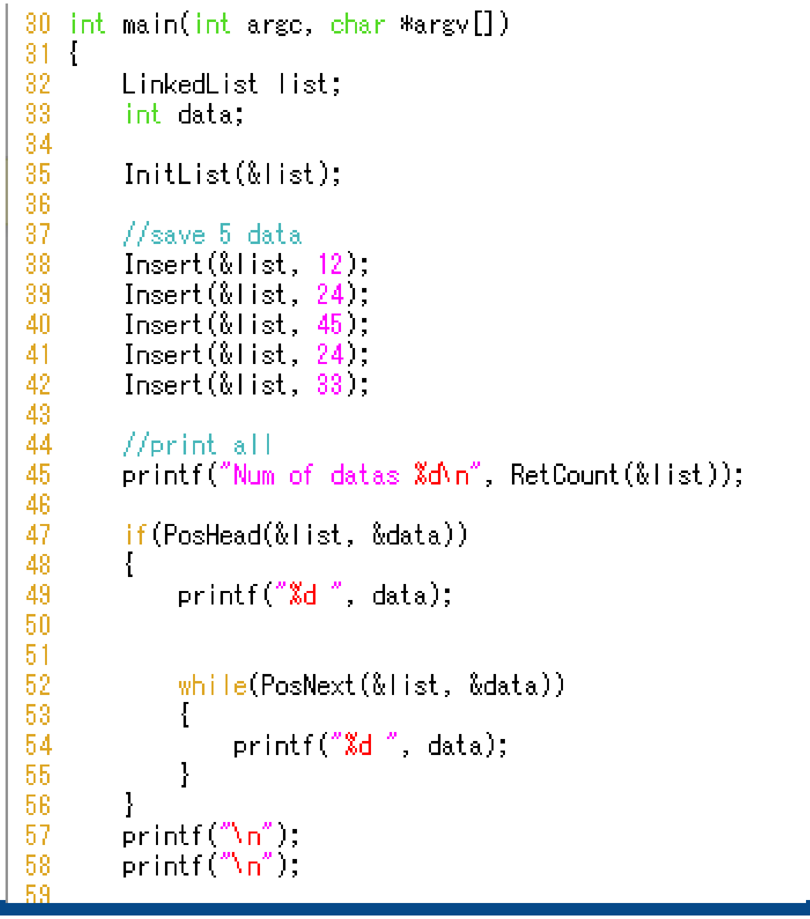
```c
int PosHead(LinkedList *list, DATATYPE *data)
{
    if(list->Head->next == NULL)
    {
        return FALSE;
    }

    list->Tail = list->Head;
    list->Cur = list->Head->next;

    *data = list->Cur->data;
    return TRUE;
}

int PosNext(LinkedList *list, DATATYPE *data)
{
    if(list->Cur->next == NULL)
    {
        return FALSE;
    }

    list->Tail = list->Cur;
    list->Cur = list->Cur->next;

    *data = list->Cur->data;
    return TRUE;
}
```



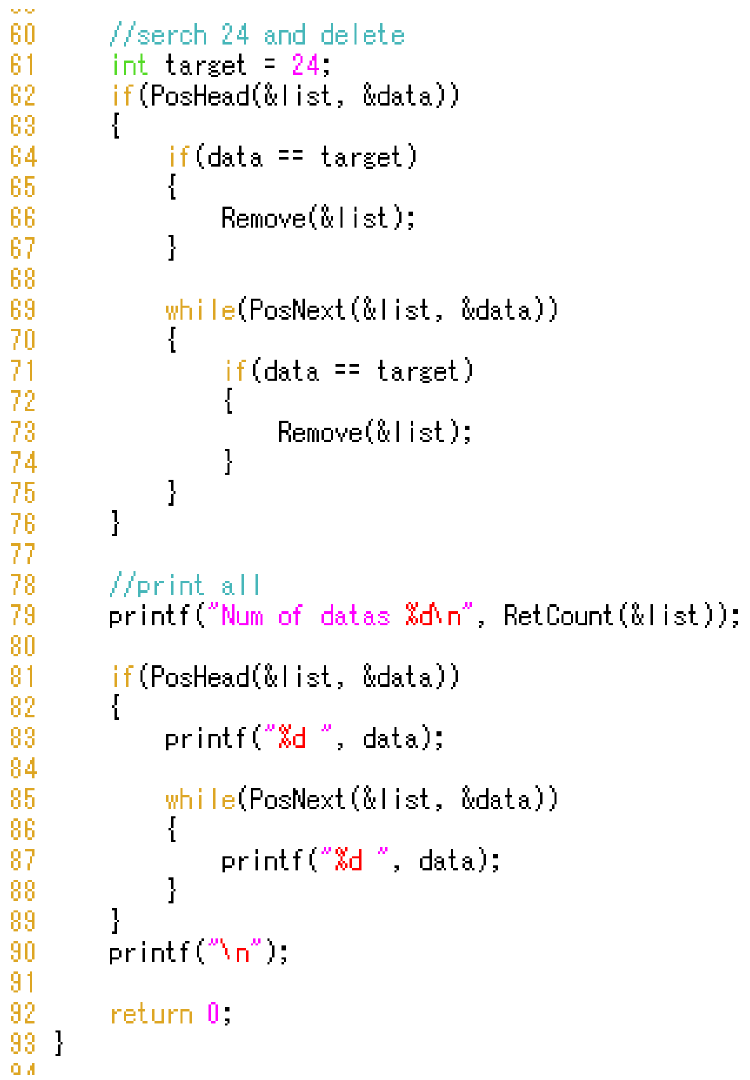
```c
DATATYPE Remove(LinkedList *list)
{
    Node *temp = list->Cur;
    DATATYPE tdata = temp->data;

    list->Tail->next = list->Cur->next;
    list->Cur = list->Tail;

    free(temp);
    list->NumofData--;
    return tdata;
}

int RetCount(LinkedList *list)
{
    return list->NumofData;
}
```

Linked List

Fixed



```c
int main(int argc, char *argv[])
{
    LinkedList list;
    int data;

    InitList(&list);

    //save 5 data
    Insert(&list, 12);
    Insert(&list, 24);
    Insert(&list, 45);
    Insert(&list, 24);
    Insert(&list, 33);

    //print all
    printf("Num of datas %d\n", RetCount(&list));

    if(PosHead(&list, &data))
    {
        printf("%d ", data);


        while(PosNext(&list, &data))
        {
            printf("%d ", data);
        }
    }
    printf("\n");
    printf("\n");

```



```c
    //serch 24 and delete
    int target = 24;
    if(PosHead(&list, &data))
    {
        if(data == target)
        {
            Remove(&list);
        }

        while(PosNext(&list, &data))
        {
            if(data == target)
            {
                Remove(&list);
            }
        }
    }

    //print all
    printf("Num of datas %d\n", RetCount(&list));

    if(PosHead(&list, &data))
    {
        printf("%d ", data);

        while(PosNext(&list, &data))
        {
            printf("%d ", data);
        }
    }
    printf("\n");

    return 0;
}
```