

자료구조 및 알고리즘 레포트 4주차

임베디드시스템공학과

2015146003

김기덕

단순 연결 리스트 프로그램 만들기

1) 프로그램 완성 코드

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <Windows.h>

// 단순 연결 리스트의 노드 구조를 구조체로 정의
typedef struct ListNode {
    char data[4];
    struct ListNode* link;
} listNode;

// 리스트 시작을 나타내는 head 노드를 구조체로 정의
typedef struct {
    listNode* head;
} linkedList_h;

// 공백 연결 리스트를 생성하는 연산
linkedList_h* createLinkedList_h(void)
{
    linkedList_h* L;
    L = (linkedList_h*)malloc(sizeof(linkedList_h));
    L->head = NULL;
    return L;
}

// 연결 리스트의 전체 메모리를 해제하는 연산
void freeLinkedList_h(linkedList_h* L)
{

```

```

listNode* p;
while (L->head != NULL)
{
    p = L->head;
    L->head = L->head->link;
    free(p);
    p = NULL;
}
}

```

```

void printList(linkedList_h* L)
{
    listNode* p;
    printf("L=");
    p = L->head;
    while (p != NULL)
    {
        printf("%s", p->data);
        p = p->link;
        if (p != NULL)
            printf(", ");
    }
    printf("\n");
}

```

// 첫 번째 노드로 삽입하는 연산

```

void insertFirstNode(linkedList_h* L, const char *x)
{
    listNode* newNode;
    // 삽입할 새 노드 할당
    newNode = (listNode*)malloc(sizeof(listNode));
    strcpy(newNode->data, x); // 새 노드의 데이터 필드에 x 저장
    newNode->link = L->head;
    L->head = newNode;
}

```

// 마지막 노드로 삽입하는 연산

```

void insertLastNode(linkedList_h* L, const char* x)
{
    listNode* newNode;
    listNode* temp;

```

```

newNode = (listNode*)malloc(sizeof(listNode));
strcpy(newNode->data, x);
newNode->link = NULL;
if (L->head == NULL) // 현재 리스트가 공백인 경우
{
    L->head = newNode; // 새 노드를 리스트의 시작 노드로 연결
    return;
}
// 현재 리스트가 공백이 아닌 경우
temp = L->head;
while (temp->link != NULL)
    temp = temp->link; // 현재 리스트의 마지막 노드를 찾음
temp->link = newNode; // 새 노드를 마지막 노드(temp)의 다음 노드로 연결
}

int main()
{
    system("title Kimkideok");

    linkedList_h* L;
    L = createLinkedList_h();

    printf("(1) 공백 리스트 생성하기! \n");
    printList(L);
    getchar();

    printf("(2) 리스트에 [수] 노드 삽입하기! \n");
    insertFirstNode(L, "수");
    printList(L);
    getchar();

    printf("(3) 리스트 마지막에 [금] 노드 삽입하기! \n");
    insertLastNode(L, "금");
    printList(L);
    getchar();

    printf("(4) 리스트 첫 번째에 [월] 노드 삽입하기! \n");
    insertFirstNode(L, "월");
    printList(L);
    getchar();
}

```

```
printf("(5) 리스트 공간을 해제하여 공백 리스트로 만들기! \n");
```

```
freeLinkedList_h(L);
```

```
printList(L);
```

```
getchar();
```

```
return 0;
```

```
}
```

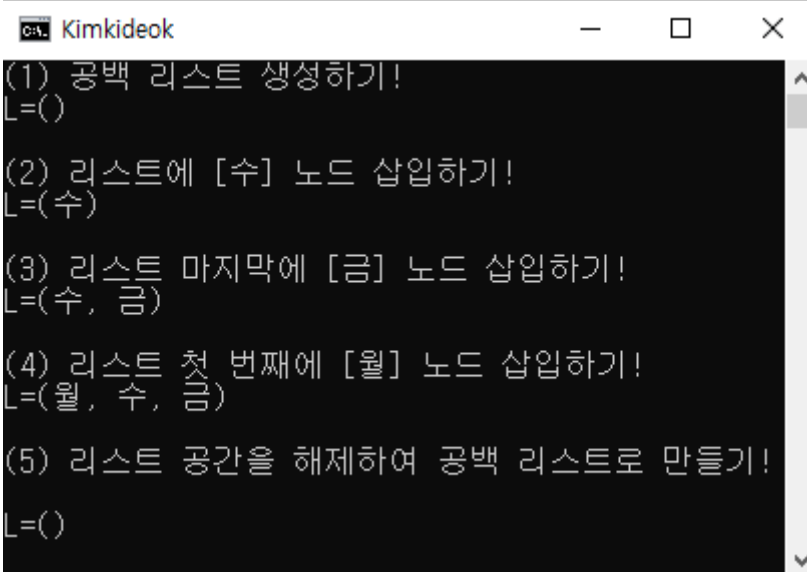
```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <string.h>
5  #include <Windows.h>
6
7  // 단순 연결 리스트의 노드 구조를 구조체로 정의
8  typedef struct ListNode {
9      char data[4];
10     struct ListNode* link;
11 } ListNode;
12
13 // 리스트 시작을 나타내는 head 노드를 구조체로 정의
14 typedef struct {
15     ListNode* head;
16 } linkedList_h;
17
18 // 공백 연결 리스트를 생성하는 연산
19 linkedList_h* createLinkedList_h(void)
20 {
21     linkedList_h* L;
22     L = (linkedList_h*)malloc(sizeof(linkedList_h));
23     L->head = NULL;
24     return L;
25 }
26
27 // 연결 리스트의 전체 메모리를 해제하는 연산
28 void freeLinkedList_h(linkedList_h* L)
29 {
30     ListNode* p;
31     while (L->head != NULL)
32     {
33         p = L->head;
34         L->head = L->head->link;
35         free(p);
36         p = NULL;
37     }
38 }
39
40 void printList(linkedList_h* L)
41 {
42     ListNode* p;
43     printf("L=(\n");
44     p = L->head;
45     while (p != NULL)
46     {
47         printf("%s", p->data);
48         p = p->link;
49         if (p != NULL)
50             printf(" ");
51     }
52     printf("\n");
53 }
```

```

64 // 첫 번째 노드 삽입하는 연산
65 void insertFirstNode(linkedList_h* L, const char *x)
66 {
67     listNode* newNode;
68     // 삽입할 새 노드 할당
69     newNode = (listNode*)malloc(sizeof(listNode));
70     strcpy(newNode->data, x); // 새 노드의 데이터 필드에 x 저장
71     newNode->link = L->head;
72     L->head = newNode;
73 }
74
75 // 마지막 노드 삽입하는 연산
76 void insertLastNode(linkedList_h* L, const char* x)
77 {
78     listNode* newNode;
79     listNode* temp;
80     newNode = (listNode*)malloc(sizeof(listNode));
81     strcpy(newNode->data, x);
82     newNode->link = NULL;
83     if (L->head == NULL) // 현재 리스트가 공백인 경우
84     {
85         L->head = newNode; // 새 노드를 리스트의 시작 노드로 연결
86         return;
87     }
88     // 현재 리스트가 공백이 아닌 경우
89     temp = L->head;
90     while (temp->link != NULL)
91         temp = temp->link; // 현재 리스트의 마지막 노드를 찾을
92     temp->link = newNode; // 새 노드를 마지막 노드(temp)의 다음 노드로 연결
93 }
94
95 int main()
96 {
97     system("title Kimkideok");
98
99     linkedList_h* L;
100     L = createLinkedList_h();
101
102     printf("(1) 공백 리스트 생성하기! \n");
103     printList(L);
104     getchar();
105
106     printf("(2) 리스트에 [수] 노드 삽입하기! \n");
107     insertFirstNode(L, "수");
108     printList(L);
109     getchar();
110
111     printf("(3) 리스트 마지막에 [금] 노드 삽입하기! \n");
112     insertLastNode(L, "금");
113     printList(L);
114     getchar();
115
116     printf("(4) 리스트 첫 번째에 [월] 노드 삽입하기! \n");
117     insertFirstNode(L, "월");
118     printList(L);
119     getchar();
120
121     printf("(5) 리스트 공간을 해제하여 공백 리스트로 만들기! \n");
122     freeLinkedList_h(L);
123     printList(L);
124     getchar();
125
126     return 0;
127 }

```

2) 실행화면



```
Kimkideok
(1) 공백 리스트 생성하기!
L=()

(2) 리스트에 [수] 노드 삽입하기!
L=(수)

(3) 리스트 마지막에 [금] 노드 삽입하기!
L=(수, 금)

(4) 리스트 첫 번째에 [월] 노드 삽입하기!
L=(월, 수, 금)

(5) 리스트 공간을 해제하여 공백 리스트로 만들기!
L=()
```

단순연결 리스트를 이용하여 영화 제목과 개봉 년도를 입력하고 출력하는 프로그램

1) 프로그램 코드

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <Windows.h>
#define SIZE 40

// 단순 연결 리스트의 노드 구조를 구조체로 정의
typedef struct ListNode {
    char data[SIZE];
    int year;
    struct ListNode* link;
} listNode;

// 리스트 시작을 나타내는 head 노드를 구조체로 정의
typedef struct {
    listNode* head;
} linkedList_h;

// 공백 연결 리스트를 생성하는 연산
linkedList_h* createLinkedList_h(void)
{
    linkedList_h* L;
    L = (linkedList_h*)malloc(sizeof(linkedList_h));
    L->head = NULL;
    return L;
}

// 연결 리스트의 전체 메모리를 해제하는 연산
void freeLinkedList_h(linkedList_h* L)
{
    listNode* p;
    while (L->head != NULL)
    {
```

```

        p = L->head;
        L->head = L->head->link;
        free(p);
        p = NULL;
    }
}

void printList(linkedList_h* L)
{
    int Num = 0;
    listNode* p; // = NULL;
    p = L->head;
    while (p != NULL)
    {
        Num++;
        printf(" %d 제목: %s\t", Num, p->data);
        printf(" 년도: %d\n", p->year);
        p = p->link;
    }
}

void insertNode(linkedList_h* L, char *s, int y)
{
    listNode* newNode;
    newNode = (listNode*)malloc(sizeof(listNode));
    strcpy(newNode->data, s);
    newNode->year = y;
    // 공백 리스트인 경우
    // 새 노드를 첫 번째이자 마지막 노드로 연결
    if (L->head == NULL)
    {
        newNode->link = NULL;
        L->head = newNode;
    }
    // 선행 노드가 있으면 그 다음 노드에 삽입되게 한다.
    else if (newNode == NULL)
    {
        newNode->link = L->head; // 새 노드를 첫 번째 노드로 삽입
        L->head = newNode;
    }
    else //리스트 중간에 삽입되게 한다.

```



```

        {
            newNode->link = L->head->link;
            L->head->link = newNode;
        }
    }

int menu()
{
    int men = 0;
    printf(" -----\\n");
    printf(" 1. 영화 정보 추가\\n");
    printf(" 2. 영화 정보 출력\\n");
    printf(" 3. 종료\\n");
    printf(" -----\\n");
    printf(" 번호를 선택하세요 : ");
    scanf("%d", &men);
    return men;
}

int main()
{
    int cho = 0, y;
    char s[40];
    linkedList_h* L;
    L = createLinkedList_h();
    while ((cho = menu()) != 3)
    {
        getchar();
        if (cho == 1)
        {
            printf(" 영화의 제목을 입력하세요 : ");
            scanf("%s", &s);
            printf(" 영화의 개봉 연도를 입력하세요 : ");
            scanf("%d", &y);
            insertNode(L, s, y);
        }
        else if (cho == 2)
        {
            printList(L);
            getchar();
        }
    }
}

```

```

        else if (cho == 3)
        {
            freeLinkedList_h(L);
            break;
        }
        system("cls");
    }
    return 0;
}

```

```

1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <string.h>
5  #include <Windows.h>
6  #define SIZE 40
7
8
9  // 단순 연결 리스트의 노드 구조를 구조체로 정의
10 typedef struct ListNode {
11     char data[SIZE];
12     int year;
13     struct ListNode* link;
14 } ListNode;
15
16 // 리스트 시작을 나타내는 head 노드를 구조체로 정의
17 typedef struct {
18     ListNode* head;
19 } linkedList_h;
20
21 // 공백 연결 리스트를 생성하는 연산
22 linkedList_h* createLinkedList_h(void)
23 {
24     linkedList_h* L;
25     L = (linkedList_h*)malloc(sizeof(linkedList_h));
26     L->head = NULL;
27     return L;
28 }
29
30 // 연결 리스트의 전체 메모리를 해제하는 연산
31 void freeLinkedList_h(linkedList_h* L)
32 {
33     ListNode* p;
34     while (L->head != NULL)
35     {
36         p = L->head;
37         L->head = L->head->link;
38         free(p);
39         p = NULL;
40     }
41 }
42

```

```

43 void printList(linkedList_h* L)
44 {
45     int Num = 0;
46     listNode* p; // = NULL;
47     p = L->head;
48     while (p != NULL)
49     {
50         Num++;
51         printf(" %d 제목: %s\n", Num, p->data);
52         printf("   년도: %d\n", p->year);
53         p = p->link;
54     }
55 }
56
57 void insertNode(linkedList_h* L, char *s, int y)
58 {
59     listNode* newNode;
60     newNode = (listNode*)malloc(sizeof(listNode));
61     strcpy(newNode->data, s);
62     newNode->year = y;
63     // 공백 리스트인 경우
64     // 새 노드를 첫 번째이자 마지막 노드로 연결
65     if (L->head == NULL)
66     {
67         newNode->link = NULL;
68         L->head = newNode;
69     }
70     // 선행 노드가 있으면 그 다음 노드에 삽입되게 한다.
71     else if (newNode == NULL)
72     {
73         newNode->link = L->head; // 새 노드를 첫 번째 노드로 삽입
74         L->head = newNode;
75     }
76     else //리스트 중간에 삽입되게 한다.
77     {
78         newNode->link = L->head->link;
79         L->head->link = newNode;
80     }
81 }
82
83 int menu()
84 {
85     int men = 0;
86     printf("-----\n");
87     printf(" 1. 영화 정보 추가\n");
88     printf(" 2. 영화 정보 출력\n");
89     printf(" 3. 종료\n");
90     printf("-----\n");
91     printf(" 번호를 선택하세요 : ");
92     scanf("%d", &men);
93     return men;
94 }
95
96 int main()
97 {
98     int cho = 0, y;
99     char s[40];
100     linkedList_h* L;
101     L = createLinkedList_h();
102     while ((cho = menu()) != 3)
103     {
104         getchar();
105         if (cho == 1)
106         {
107             printf(" 영화의 제목을 입력하세요 : ");
108             scanf("%s", &s);
109             printf(" 영화의 개봉 연도를 입력하세요 : ");
110             scanf("%d", &y);
111             insertNode(L, s, y);
112         }
113         else if (cho == 2)
114         {
115             printList(L);
116             getchar();
117         }
118         else if (cho == 3)
119         {
120             freeLinkedList_h(L);
121             break;
122         }
123         system("cls");
124     }
125     return 0;
126 }
127

```

2) 실행화면

```
C:\WINDOWS\system32\cmd.exe
-----
1. 영화 정보 추가
2. 영화 정보 출력
3. 종료
-----
번호를 선택하세요 : 1
영화의 제목을 입력하세요 : 극한직업
영화의 개봉 연도를 입력하세요 : 2019
```

1번을 눌러 영화를 추가

```
C:\WINDOWS\system32\cmd.exe
-----
1. 영화 정보 추가
2. 영화 정보 출력
3. 종료
-----
번호를 선택하세요 : 2
1 제목: 명량          년도: 2014
2 제목: 어벤저스      년도: 2019
3 제목: 국제시장      년도: 2014
4 제목: 신과함께      년도: 2017
5 제목: 극한직업      년도: 2019
```

2번을 눌러 영화를 출력

```
C:\WINDOWS\system32\cmd.exe
-----
1. 영화 정보 추가
2. 영화 정보 출력
3. 종료
-----
번호를 선택하세요 : 3
계속하려면 아무 키나 누르십시오 . . .
```

3번을 눌러 종료

3) 분석

```

9 // 단순 연결 리스트의 노드 구조를 구조체로 정의
10 typedef struct ListNode {
11     char data[SIZE];
12     int year;
13     struct ListNode* link;
14 } ListNode;
15
16 // 리스트 시작을 나타내는 head 노드를 구조체로 정의
17 typedef struct {
18     ListNode* head;
19 } linkedList_h;
20
21 // 공백 연결 리스트를 생성하는 연산
22 linkedList_h* createLinkedList_h(void)
23 {
24     linkedList_h* L;
25     L = (linkedList_h*)malloc(sizeof(linkedList_h));
26     L->head = NULL;
27     return L;
28 }

```

단순히 구조체를 선언하고 정의

연결 리스트를 생성한다.

```

96 int main()
97 {
98     int cho = 0, y;
99     char s[40];
100     linkedList_h* L;
101     L = createLinkedList_h();
102     while ((cho = menu()) != 3)
103     {
104         getchar();
105         if (cho == 1)
106         {
107             printf(" 영화의 제목을 입력하세요 : ");
108             scanf("%s", &s);
109             printf(" 영화의 개봉 연도를 입력하세요 : ");
110             scanf("%d", &y);
111             insertNode(L, s, y);
112         }
113         else if (cho == 2)
114         {
115             printList(L);
116             getchar();
117         }
118         else if (cho == 3)
119         {
120             freeLinkedList_h(L);
121             break;
122         }
123         system("cls");
124     }
125     return 0;
126 }

```

main문 먼저 살펴보면

menu 출력

```

83 int menu()
84 {
85     int men = 0;
86     printf(" -----<
87     printf(" 1. 영화 정보 추가\n");
88     printf(" 2. 영화 정보 출력\n");
89     printf(" 3. 종료\n");
90     printf(" -----<
91     printf(" 번호를 선택하세요 : ");
92     scanf("%d", &men);
93     return men;
94 }

```

cho가 1이면 영화를 입력받는다.

cho가 2이면 입력받은 영화를 모두 출력한다.

cho가 3이면 list를 초기화하고 루프를 탈출한다.

```

57 void insertNode(linkedList_h* L, char *s, int y)
58 {
59     listNode* newNode;
60     newNode = (listNode*)malloc(sizeof(listNode));
61     strcpy(newNode->data, s);
62     newNode->year = y;
63     // 공백 리스트인 경우
64     // 새 노드를 첫 번째이자 마지막 노드로 연결
65     if (L->head == NULL)
66     {
67         newNode->link = NULL;
68         L->head = newNode;
69     }
70     // 선행 노드가 있으면 그 다음 노드에 삽입되게 한다.
71     else if (newNode == NULL)
72     {
73         newNode->link = L->head; // 새 노드를
74         L->head = newNode; // 첫 번째 노드로 삽입
75     }
76     else // 리스트 중간에 삽입되게 한다.
77     {
78         newNode->link = L->head->link;
79         L->head->link = newNode;
80     }
81 }

```

insert노드 함수는 새로운 노드를 기존 노드에 삽입시키는 함수이다.

공백리스트인 경우에 새노드를 첫 번째이자 마지막노드로 연결

선행노드가 있으면 그 다음 노드에 삽입되게 한다.

리스트 중간에 삽입되게 한다.

cho가 1이면 실행된다.

```

30 // 연결 리스트의 전체 메모리를 해제하는 연산
31 void freeLinkedList_h(linkedList_h* L)
32 {
33     listNode* p;
34     while (L->head != NULL)
35     {
36         p = L->head;
37         L->head = L->head->link;
38         free(p);
39         p = NULL;
40     }
41 }
42
43 void printList(linkedList_h* L)
44 {
45     int Num = 0;
46     listNode* p; // = NULL;
47     p = L->head;
48     while (p != NULL)
49     {
50         Num++;
51         printf(" %d 제목: %s\t", Num, p->data);
52         printf(" 년도: %d\n", p->year);
53         p = p->link;
54     }
55 }

```

연결리스트의 전체 메모리를 해제한다.

cho가 3이면 실행된다.

리스트를 출력한다.

cho가 2면 실행된다.