텀프로젝트 프로그램 보고서 1613509 최효린

1번

1. 함수에 대한 설명

-printpro()

Arrayin를 출력하는 함수입니다. for문을 사용하여 배열전체를 출력하게 하였고, 배열의 시작과 끝을 구분짓기 위해 [ ] 기호를 사용하였습니다.

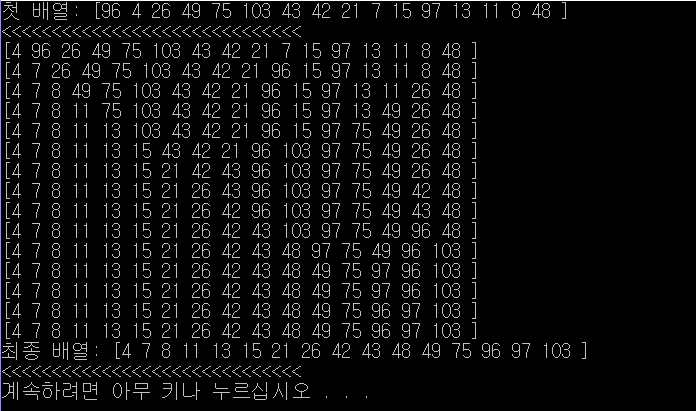
-SortArr()

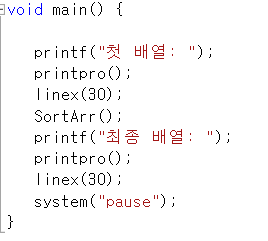
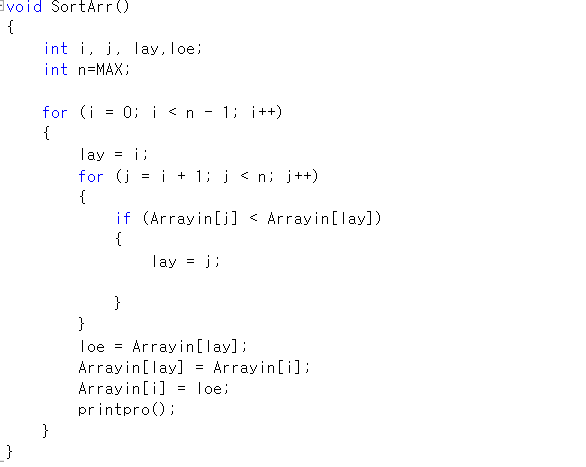
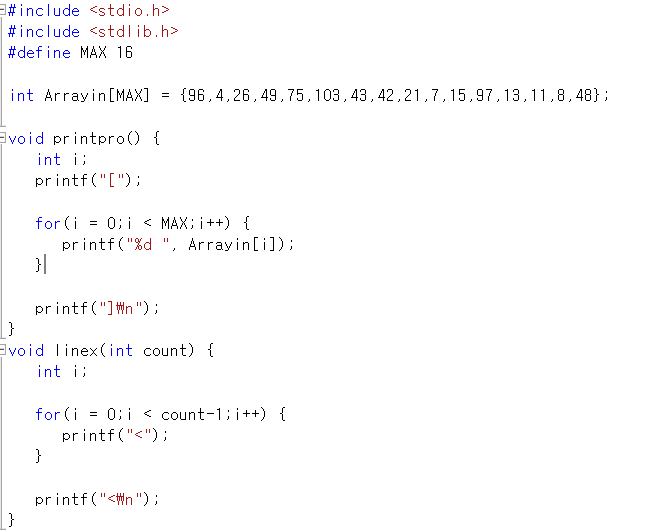
for문을 사용하여 왼쪽부터 차례대로 수를 비교하여 전의 수보다 작으면 둘의 위치를 바꾸고 크면 그대로 두고 다음 수로 넘어갈 수 있게 코드를 짰습니다. 그리고 배열이 끝나면 배열전체가 출력될 수 있게 printpro 함수를 사용하였습니다.

-linex(int count)

Selection sort 초기 배열, 결과 배열, 배열의 변화 절차를 구분하기 위해 >>>>>>> 기호를 출력하는 함수를 구현하였습니다.

1. 실행화면





2번

1. 함수에 대한 설명

-addli(struct nunode \*\*gu, int num)

새로운 링크 노드를 추가하는 함수입니다.링크 노드가 없으면 새로 할당을 해주고, 이미 존재한다면 그 뒤에 연결하여 생성해주는 코드입니다.

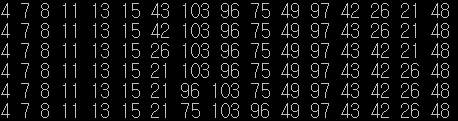
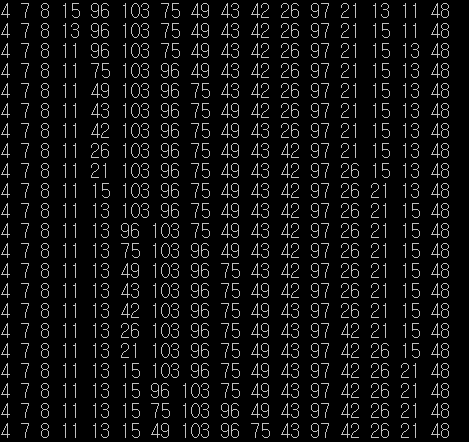
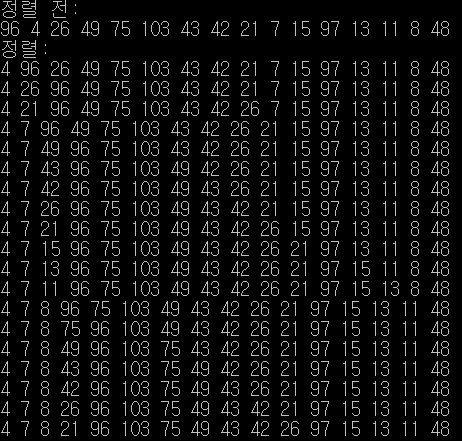
-selection\_sortli(void)

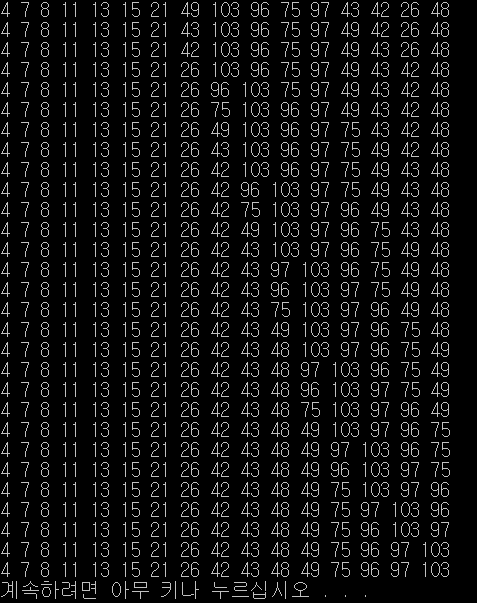
Selection sort 를 구현한 함수입니다. 교재에 있는 linkedlist를 활용한 selection sort 코드를 활용하여 코드를 짰습니다. n1이 head 일 때와 아닐 때를 나누어 케이스별로 코드를 작성하였습니다. 앞의 노드 data와 비교하여 작으면 노드의 링크를 바꾸어 연결하여 둘의 자리를 바꾸는 알고리즘을 이용하였습니다. 값을 출력할 때는 아래의 printli 함수를 이용하였습니다.

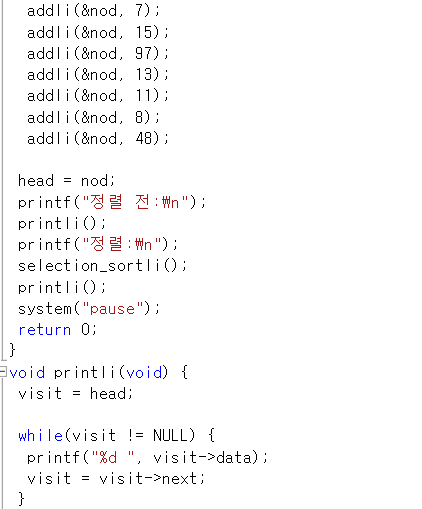
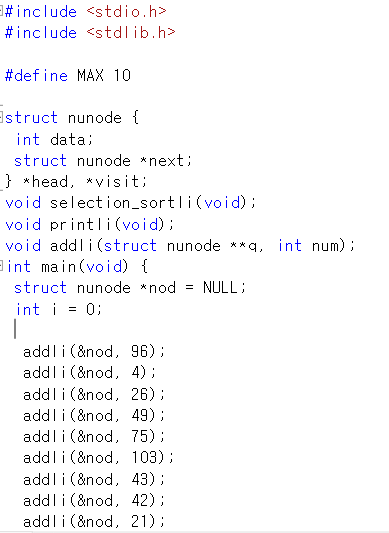
-printli(void)

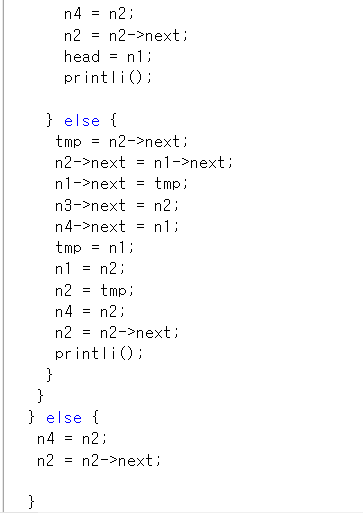
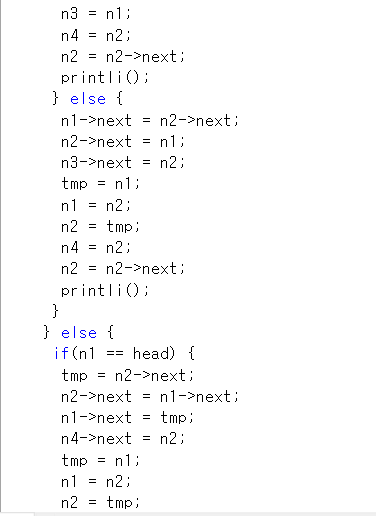
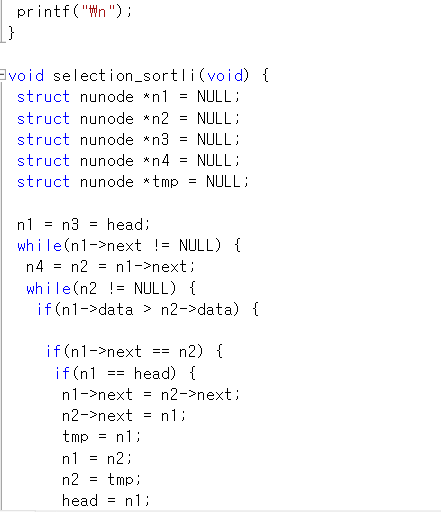
노드의 데이터를 출력해 주는 함수입니다.

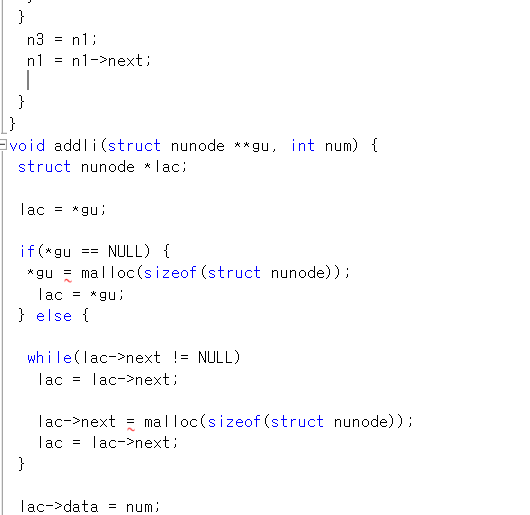
1. 실행화면













3번

1. 함수에 대한 설명

-void sortAY(int arr[], int n)

배열을 정렬해주는 함수입니다. arr와 n(배열의 수)을 입력 받으면 arr 배열 중에서 n개의 수만큼을 정렬하게 해줍니다.

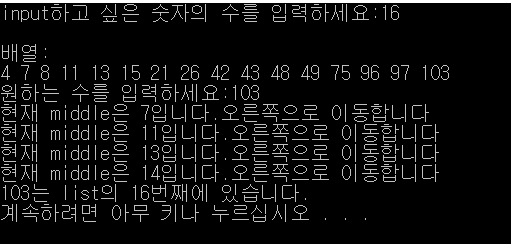
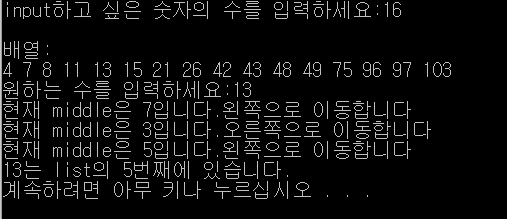
-int comp(int m, int n)

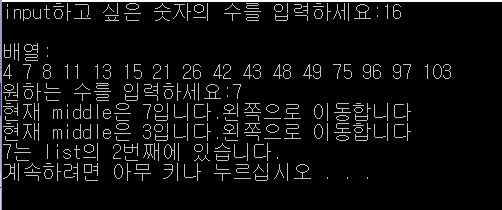
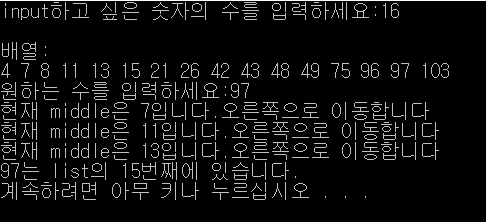
입력 받은 m 과 n의 크기를 비교하며 m==n일 경우 0을 반환하고 m < n 의 경우에는 1을 반환하고 그 외의 경우에는 -1을 반환합니다.

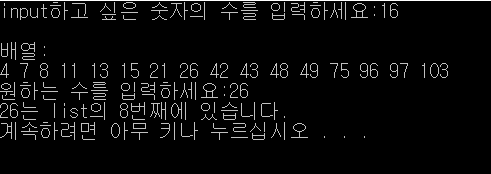
-int bin(int arr[], int find, int left, int right)

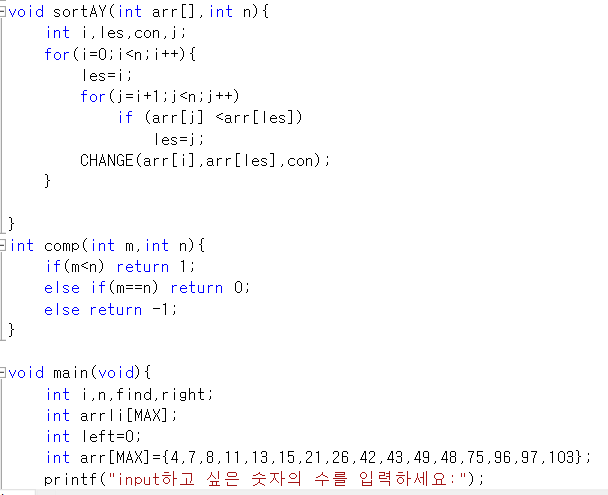
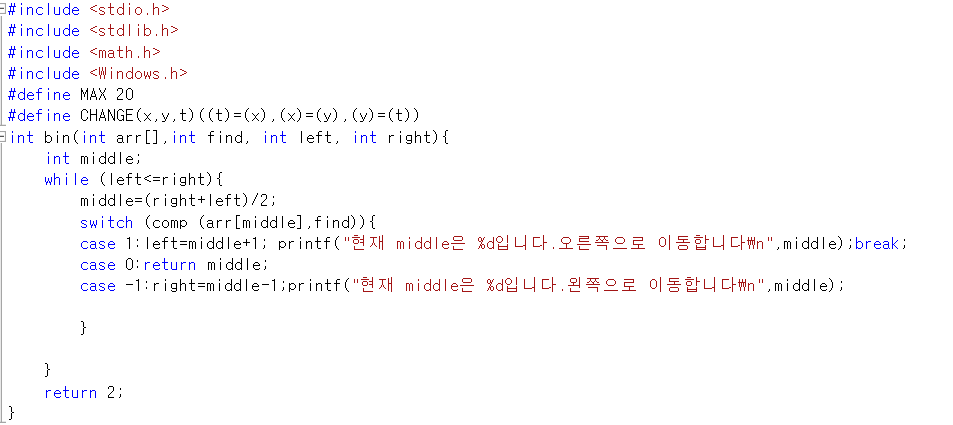
binary search를 구현한 함수입니다. arr[]는 binary search하려는 배열을 의미하고 find는 찾고자 하는 수를 의미합니다. 그리고 그 배열의 첫 시작인 left와 끝인 right를 index값으로 주고 함수 안에서 left와 right의 중간을 middle로 정하였습니다. 그리고 while문을 사용하여 left와 right가 같아질 때까지 실행하도록 하였고 그 안에서 comp함수를 써서 arr[middle]과 find을 비교 한 후 middle보다 찾고자 하는 수의 index가 크면 왼쪽으로 이동한다는 메시지가 뜨고 작으면 오른쪽으로 이동한다는 메시지가 출력되도록 하였습니다.

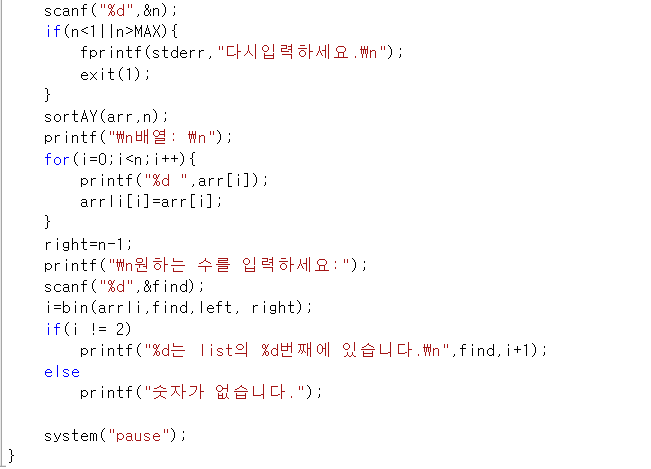
1. 실행화면











4번

1. 함수에 대한 설명

-int longwid(listnode \*list)

리스트의 길이를 출력 하는 함수입니다.

-void error(char \*warn)

에러 메세지를 출력하여 주는 함수입니다.

-int comp(int m, int n)

m과 n의 크기를 비교하며 m< n 의 경우에는 1을 반환하고 m==n일 경우 0을 반환하고 그 외의 경우에는 -1을 반환하는 함수입니다.

-linkedlist \*getnode (listnode \*list, int place)

매개변수로 입력 받은 place위치의 노드를 반환합니다. place가 0보다 작을 경우에는 null을 반환하고 그 외의 경우에는 for문을 이용하여 place위치까지 그 노드를 반환하게 하는 함수입니다.

-void insnode(linkedlist \*\*go, linkedlist \*top, linkedlist \*new\_node)

새로운 노드를 삽입하기 위한 함수입니다. ppt에 있던 노드 삽입 코드를 변형하여 사용하였습니다.

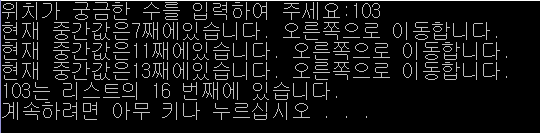
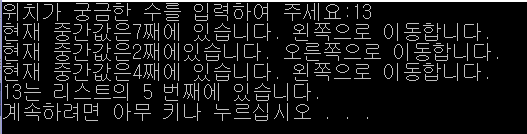
-void add(listnode \*ln, int boom, int con) 과 void addlt(listnode \*ln, int con)

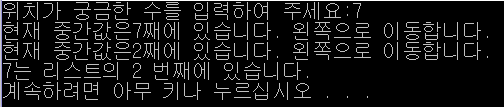
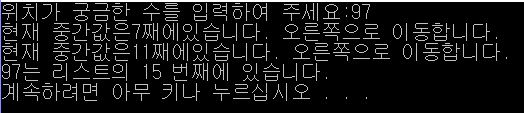
리스트에 원하는 노드를 추가하여 줍니다. 헤드 노드가 없을 경우에는 첫번째 노드로 삽입하고 아닐 경우에는 기존의 노드 뒤에 이어줍니다.

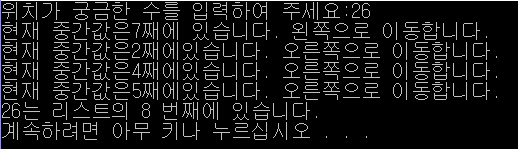
-int bin(listnode \*list, int find, int left, int right)

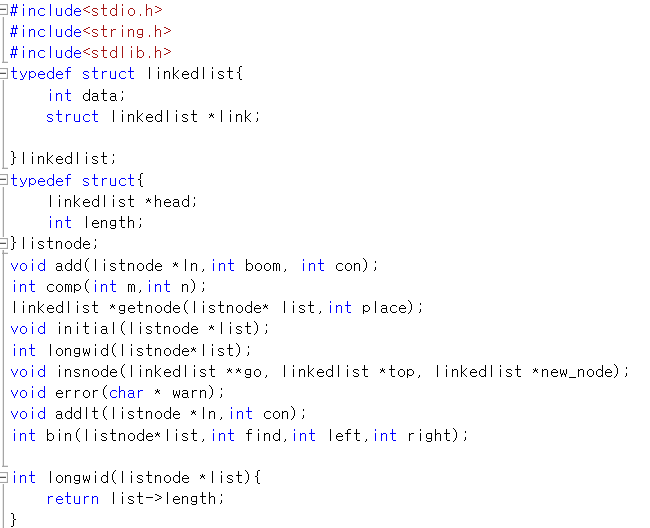
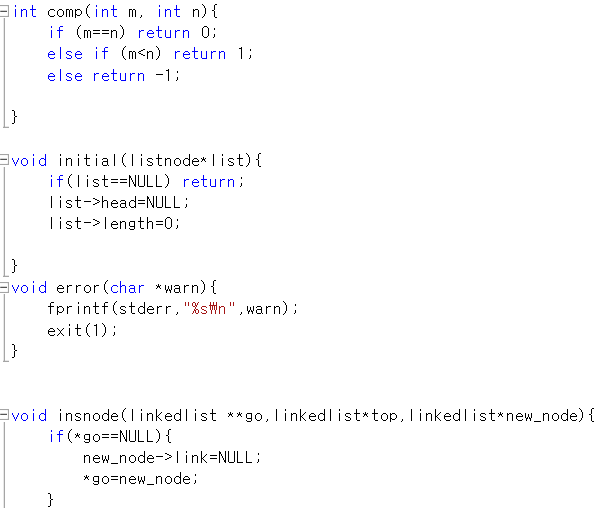
binary search를 구현한 함수입니다. find는 위치를 알고자 하는 수이고 list는 binary search하려는 것입니다. 그리고 시작인 left와 끝인 right를 입력 받습니다. 또한 함수 내에서 중간 값을 나타내는 변수 mid를 설정하였습니다. 그 후 getnode 함수를 이용하여 mid위치의 노드의 데이터를 받아온 후 comp 함수을 이용하여 mid위치의 데이터 값과 find를 비교하여 left와 right가 같아질 때 mid를 반환하게 합니다. 이때 left와 right 값은 index값이므로 comp함수로 비교할 때 기준이 되는 index 변수인 stand변수로 비교를 하게 하였습니다.

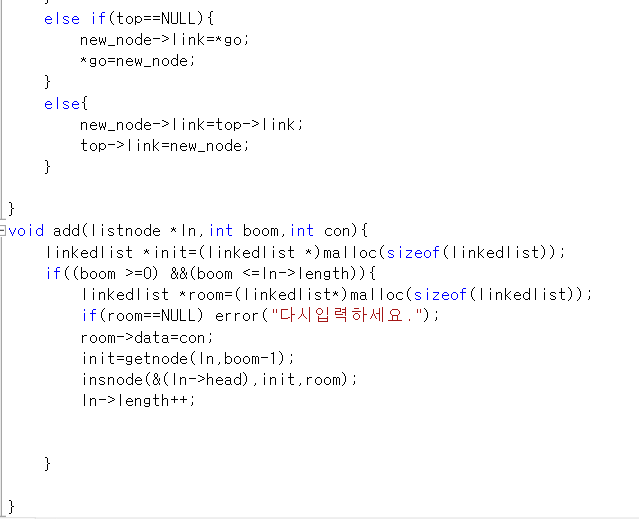
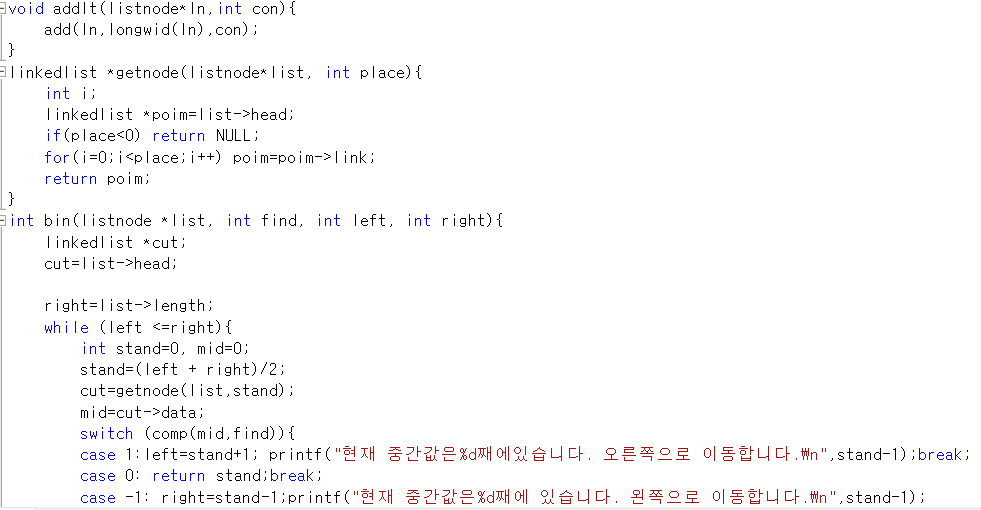
1. 실행화면

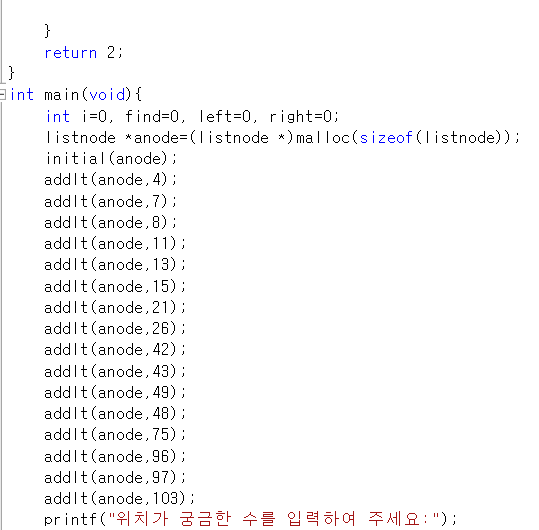
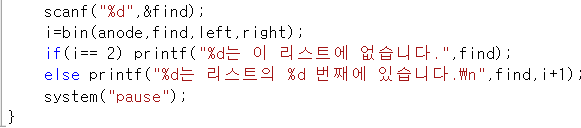






5번-1,2

1. 함수에 대한 설명

-void back(char \*warn)

에러를 출력해주는 함수입니다.

-void addlt(linkedlist \*\*phed, linkedlist \*node)

리스트에 원하는 노드를 추가해주는 함수입니다. 헤드 노드가 없으면 첫번째 노드로 삽입하고 아닐 경우에는 기존에 있던 노드 뒤에 이어줍니다.

-void rem(linkedlist \*\*phead, linkedlist \*top, linkedlist \*erase)

erase는 제거하고자 하는 노드를 의미합니다. 제거하고 싶은 노드가 가지고 있는 링크필드의 정보를 선행노드 top이 가리키게 만들어준 후 erase노드의 메모리를 free라는 함수를 이용하여 제거합니다.

-linkedlist \*creat(char \*mem, linkedlist \* nub)

새로운 노드를 만드는 함수입니다. 새로 만들어진 노드에 입력 받은 mem을 데이터 필드에 저장하고 매개변수로 입력 받은 nub를 링크 필드에 저장하여 리스트를 만들어줍니다. 이 때 함수를 호출할 때마다 리스트의 길이를 1씩 증가하게 하였습니다.

-linkedlist \*gtnode(linkedlist \*list, int nav)

nav위치의 노드를 반환하는 함수입니다. for문을 이용하여 nav위치까지 노드를 반환하게 하였고, nav가 음수일 경우에는 null을 반환하도록 하였습니다.

-void printli(linkedlist \*head)

do-while문을 이용하여 입력 받은 리스트의 데이터 필드를 출력하는 함수입니다.

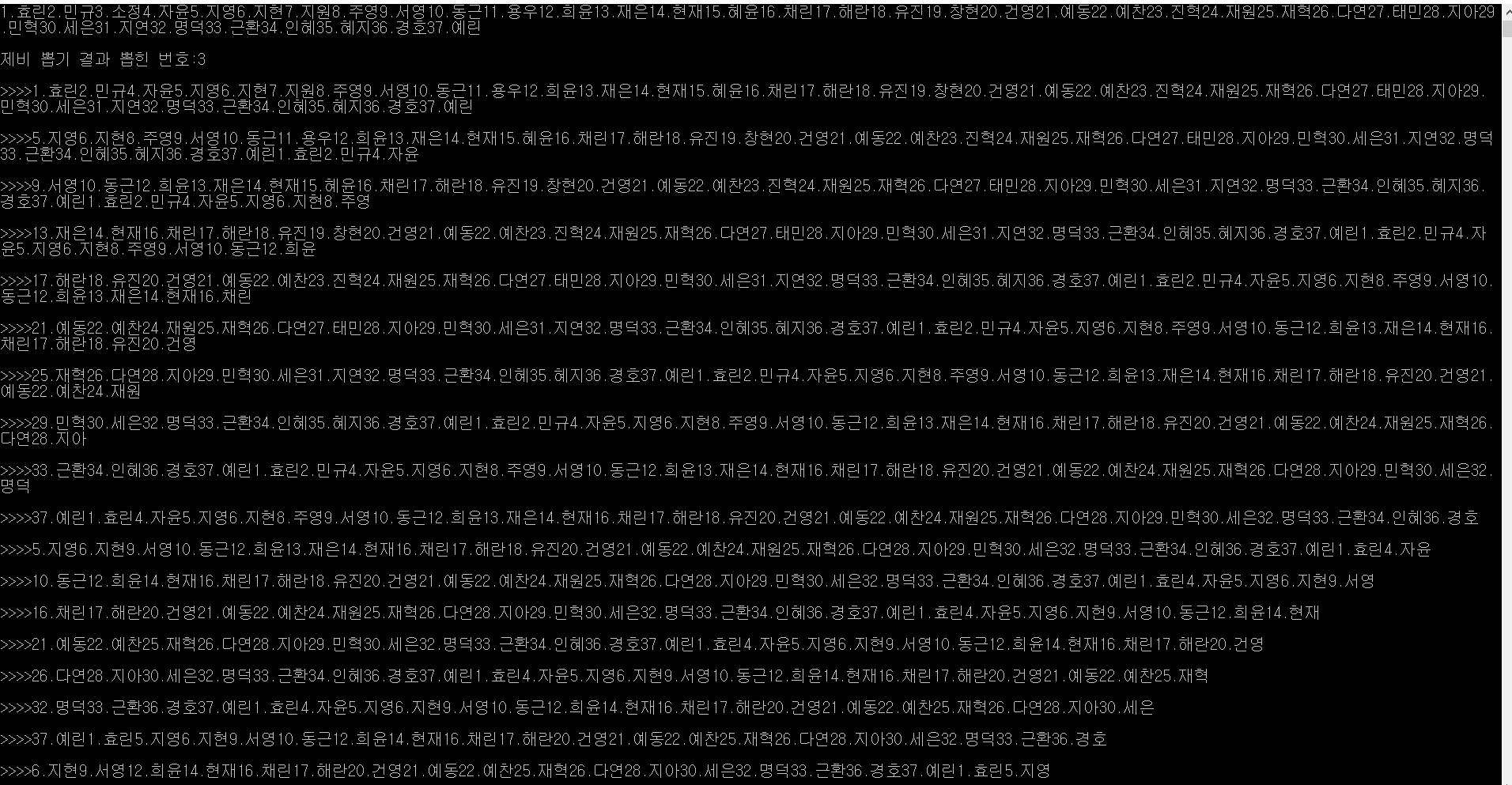
-void del(linkedlist \*list, int nav)

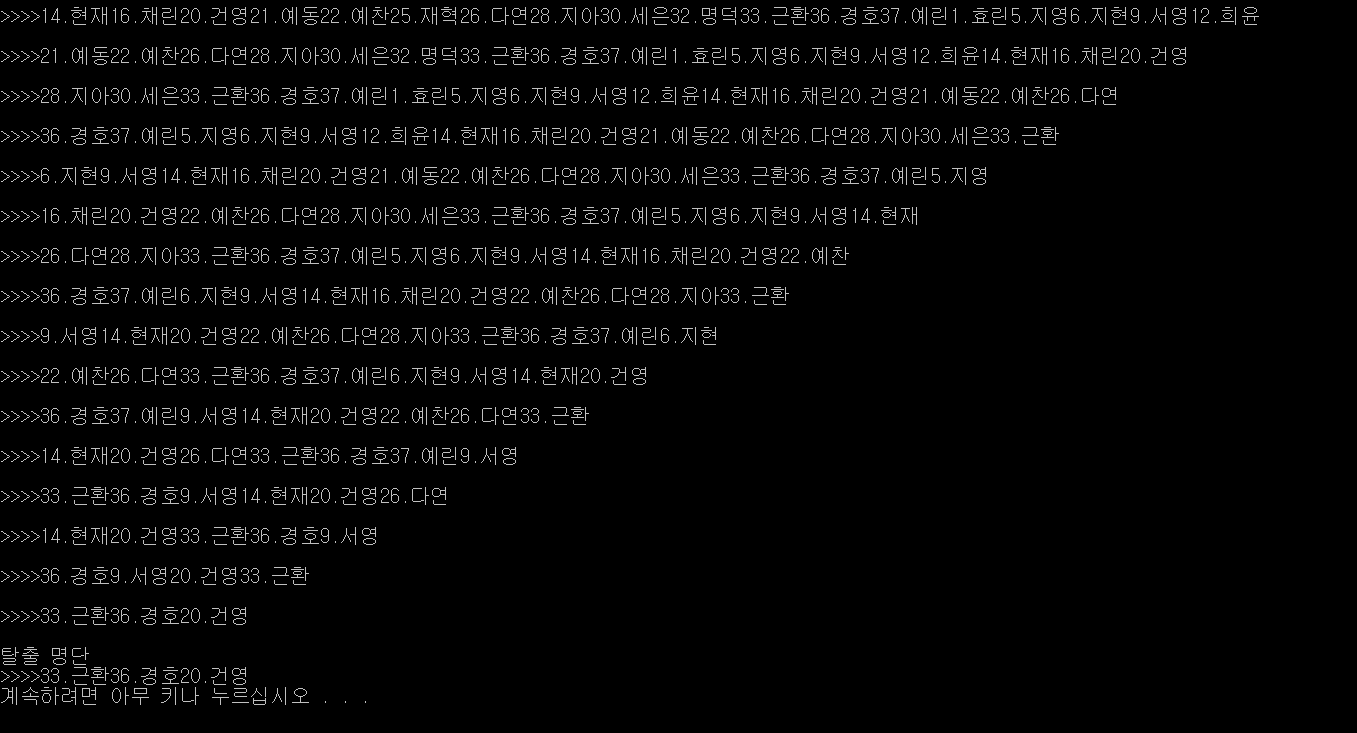
노드를 삭제하는 함수입니다. 리스트를 입력 받고 삭제하고 싶은 위치의 값을 nav로 받은 후 gtnode함수를 이용하여 가져옵니다. 그 후 rem 함수를 이용해서 노드를 삭제하게 됩니다. 이 때 삭제 될 때마다 리스트의 전체 길이는 1씩 감소합니다.

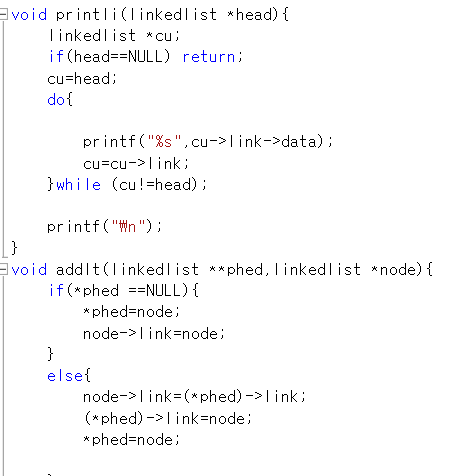
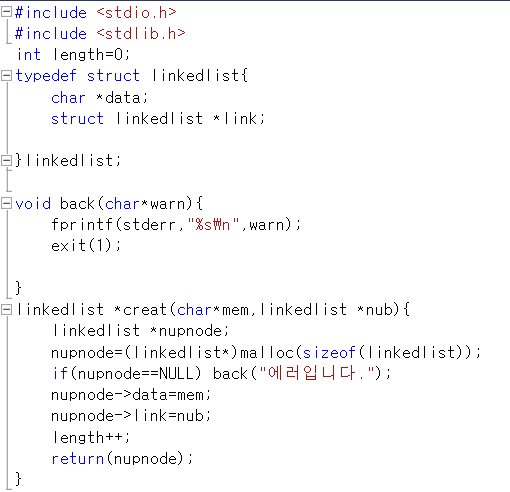
-int main(void)

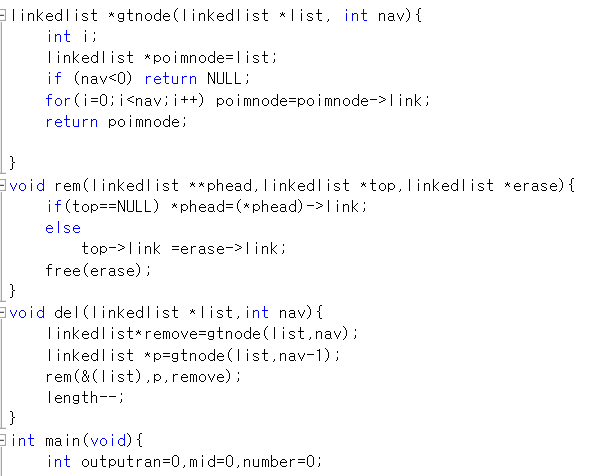
random함수를 이용하여 outputran에 그 수를 저장합니다. mid는 중간에 사용될 변수로 outputran 만큼씩 더해지는 역할을 합니다. 이 때 head를 유지하게 되면 mid가 가진 수가 인덱스 로서의 역할을 못하게 되어 원하는 위치의 노드가 삭제되지 않습니다. 그래서 ouhead라는 노드를 두어서 del함수가 수행되기전에 원래 ouhead 위치보다 1 큰 수의 노드를 ouhead라고 두고 del함수가 수행된 후 그 노드를 가리키게 하여 head 위치에 변화를 두어서 outputran만큼씩 삭제되게 하려고 하였으나 잘못된 노드가 삭제되는 부분에 대해서 아직 해결하지 못하였습니다.

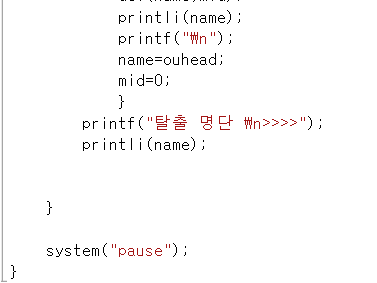
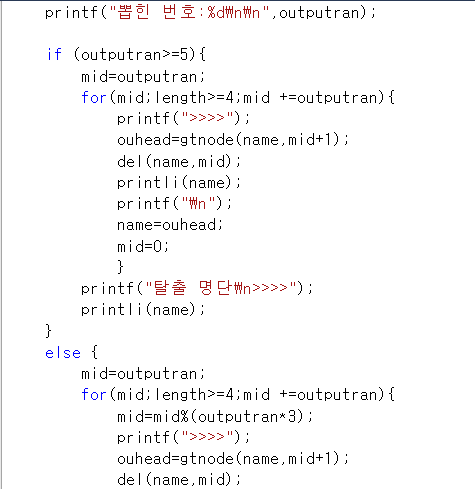
1. 실행화면











5번-3,4

1. 함수에 대한 설명

위의 5번-1,2 에서 와 동일한 함수명과 함수기능을 하게 프로그램을 짜보았습니다. 단지 singly cicular list를 doubly circular list로 바꾸어 구현한 것 말고는 차이점이 없습니다. 하지만 오류를 찾지 못하여 실행되지 않습니다.

1. 실행화면

