Programming Example

1.

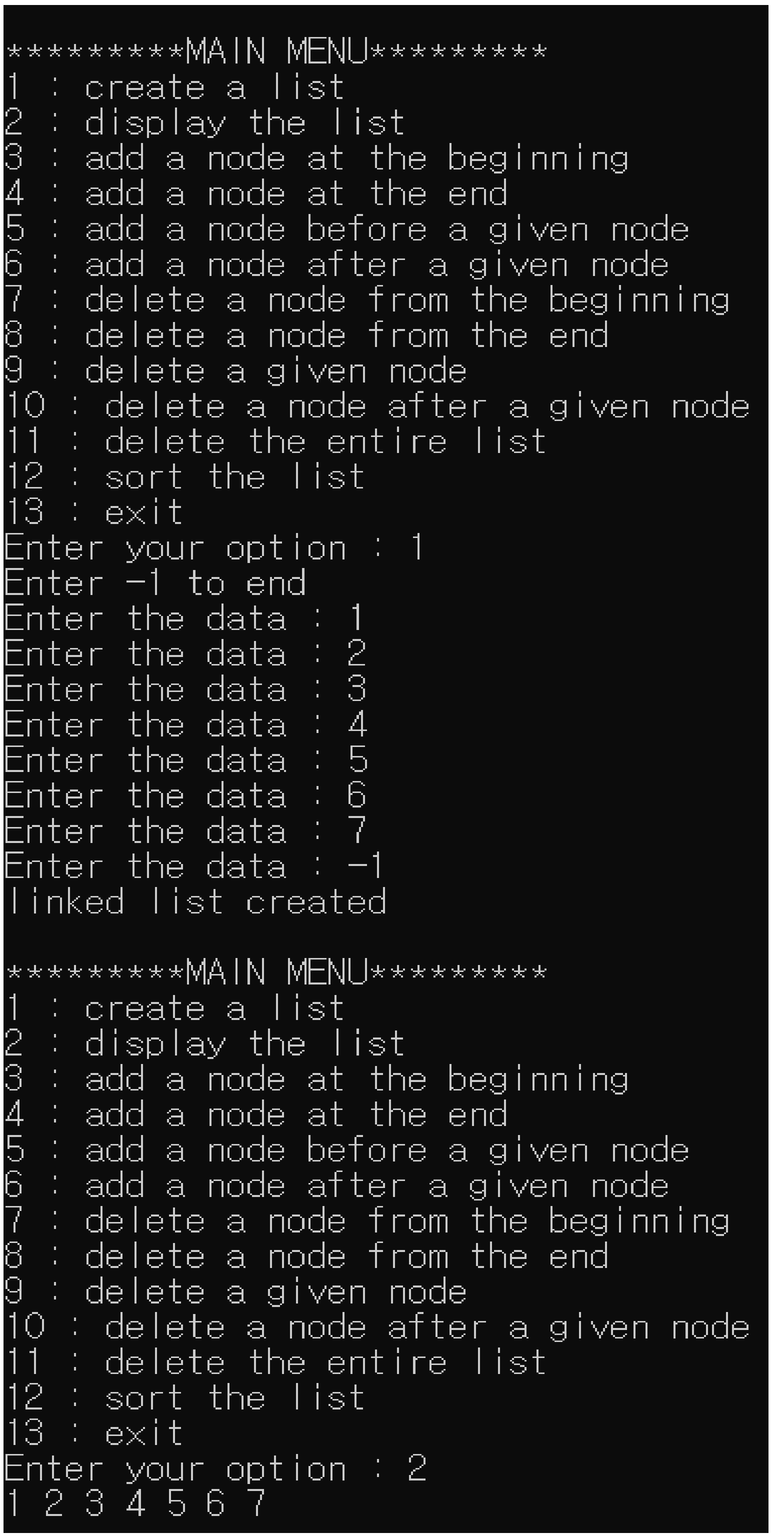
```
#include <stdio.h>
#include <malloc.h>

struct node
{
        int data;
        struct node *next;
};

struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *insert_before(struct node *);
struct node *insert_after(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_node(struct node *);
struct node *delete_after(struct node *);
struct node *delete_list(struct node *);
struct node *sort_list(struct node *);
int main()
{
        int option;
        do{
        printf("\n");
        printf("\n");
        printf("*********MAIN MENU*********\n");
        printf("1 : create a list \n");
        printf("2 : display the list\n");
        printf("3 : add a node at the beginning\n");
        printf("4 : add a node at the end\n");
        printf("5 : add a node before a given node\n");
        printf("6 : add a node after a given node\n");
        printf("7 : delete a node from the beginning\n");
        printf("8 : delete a node from the end\n");
        printf("9 : delete a given node\n");
        printf("10 : delete a node after a given node\n");
        printf("11 : delete the entire list\n");
        printf("12 : sort the list\n");
printf("13 : exit\n");
printf("Enter your option : ");
scanf("%d",&option);
switch(option)
{
          case 1: start=create(start);
                              printf("linked list created");
                              break;

          case 2: start=display(start);
                              break;

          case 3: start=insert_beg(start);
                              break;

          case 4: start=insert_end(start);
                         break;

          case 5: start=insert_before(start);
                         break;

          case 6: start=insert_after(start);
                         break;

          case 7: start=delete_beg(start);
                         break;

          case 8: start=delete_end(start);
                              break;

          case 9: start=delete_node(start);
                              break;

          case 10: start=delete_after(start);
                          break;
          case 11: start=delete_list(start);
                               break;

          case 12: start=sort_list(start);

}

}while(option!=13);
```

```
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d ",ptr->data);
		ptr=ptr->next;
	}
	return start;
}
```

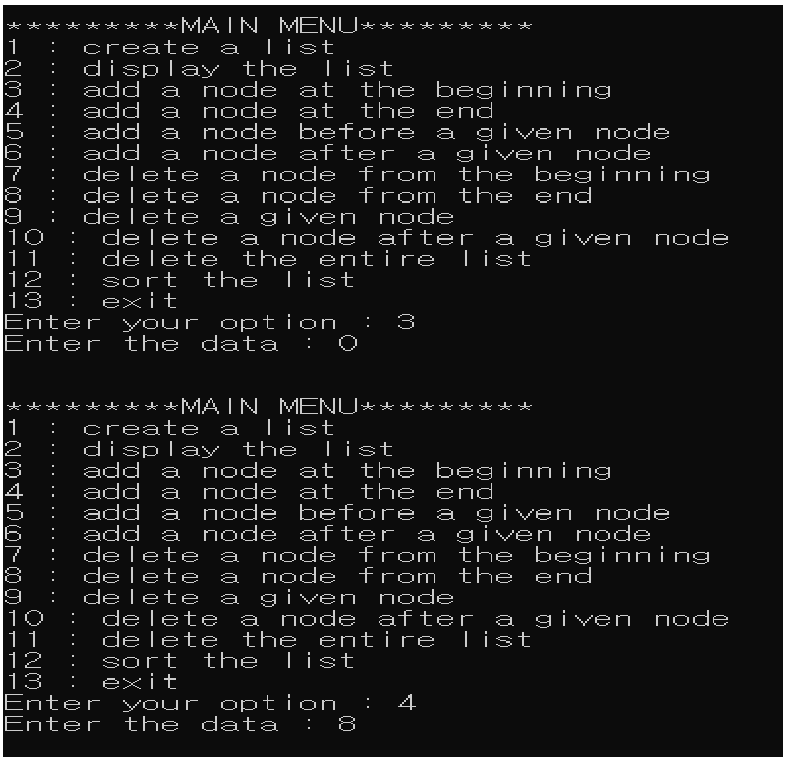
```
struct node *insert_beg(struct node *start)
{
        struct node *new_node;
        int num;
        printf("Enter the data : ");
        scanf("%d",&num);
        new_node=(struct node *)malloc(sizeof(struct node));
        new_node->data=num;
        new_node->next=start;
        start=new_node;
}

struct node *insert_end(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter the data : ");
        scanf("%d",&num);
        new_node=(struct node*)malloc(sizeof(struct node));
        new_node->next=NULL;
        new_node->data=num;
        ptr=start;
        while(ptr->next!=NULL)
                ptr=ptr->next;
        ptr->next=new_node;
        return start;
}

struct node *insert_before(struct node *start)
{
        struct node *new_node, *ptr, *preptr;
        int num,val;
        printf("Enter the data : ");
        scanf("%d",&num);
        printf("Enter the value before which the data has to be inserted : ");
        scanf("%d",&val);
        new_node=(struct node*)malloc(sizeof(struct node));
        new_node->data=num;
    ptr=start;
        while(ptr->data!=val)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=new_node;
        new_node->next=ptr;
        return start;
}

struct node *insert_after(struct node *start)
{
        struct node *new_node, *ptr, *preptr;
        int num,val;
        printf("Enter the data : ");
        scanf("%d",&num);
        printf("Enter the value before which the data has to be inserted : ");
        scanf("%d",&val);
        new_node=(struct node *)malloc(sizeof(struct node));
        new_node->data=num;
        ptr=start;
        preptr=ptr;
        while(preptr->data!=val)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=new_node;
        new_node->next=ptr;
        return start;
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        free(ptr);
        return start;
}

struct node *delete_end(struct node *start)
{
        struct node *ptr, *preptr;

        ptr=start;
        while(ptr->next!=NULL)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=NULL;
        free(ptr);
        return start;
}

struct node *delete_node(struct node *start)
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the value of the node which has to be deleted : ");
        scanf("%d",&val);
        ptr=start;
        if(ptr->data == val)
        {
                start=delete_beg(start);
                return start;
        }
        else
        {
                while(ptr->data!=val)
                {
                        preptr=ptr;
                        ptr=ptr->next;
                }
                preptr->next=ptr->next;
                free(ptr);
                return start;
        }
}

struct node *delete_after(struct node *start)
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the value after which the node has to be deleted");
        scanf("%d",&val);
        ptr=start;
        preptr=ptr;
```

```
		while(preptr->data!=val)
		{
			preptr=ptr;
			ptr=ptr->next;
		}
		preptr->next=ptr->next;
		free(ptr);
		return start;
}

struct node *delete_list(struct node *start)
{
		struct node *ptr;
		if(start!=NULL)
		{
			ptr=start;
			while(ptr!=NULL)
			{
				printf("%d is to be deleted next\n",ptr->data);
				start=delete_beg(ptr);
				ptr=start;
			}
		}
		return start;
}

struct node *sort_list(struct node *start)
{
		struct node *ptr1, *ptr2;
		int temp;
		ptr1=start;
		while(ptr1->next!=NULL)
		{
			ptr2=ptr1->next;
			while(ptr2!=NULL)
			{
				if(ptr1->data > ptr2->data)
				{
					temp=ptr1->data;
					ptr1->data=ptr2->data;
					ptr2->data=temp;
				}
				ptr2=ptr2->next;
```

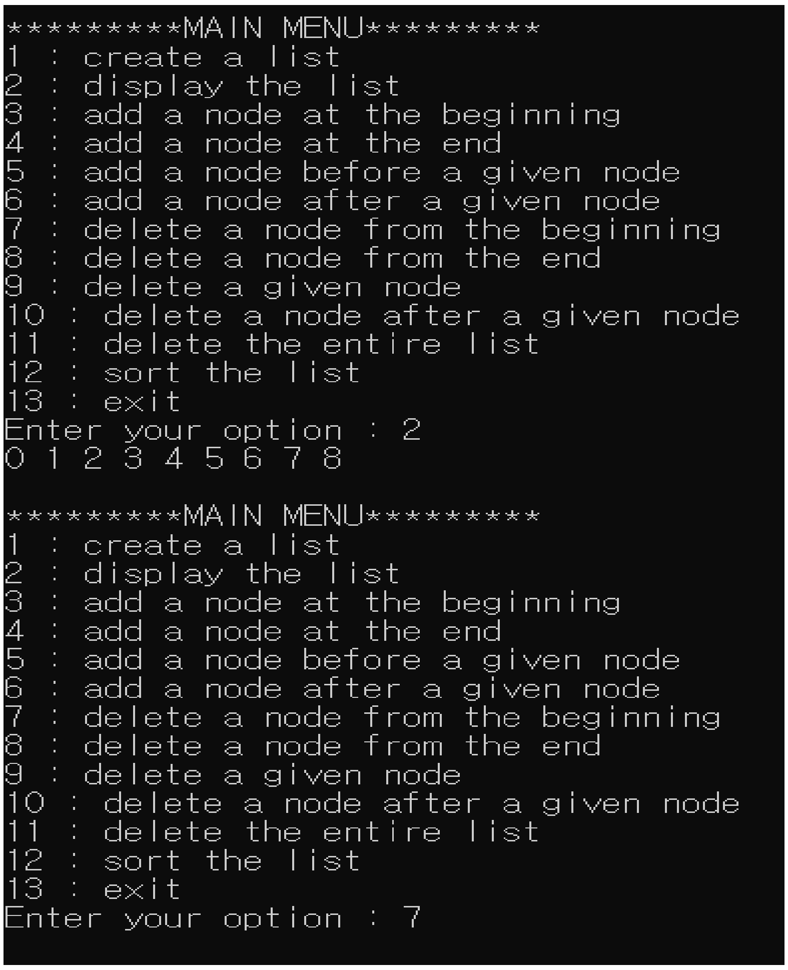
```
                }
                ptr1=ptr1->next;
        }
        return start;
}
```



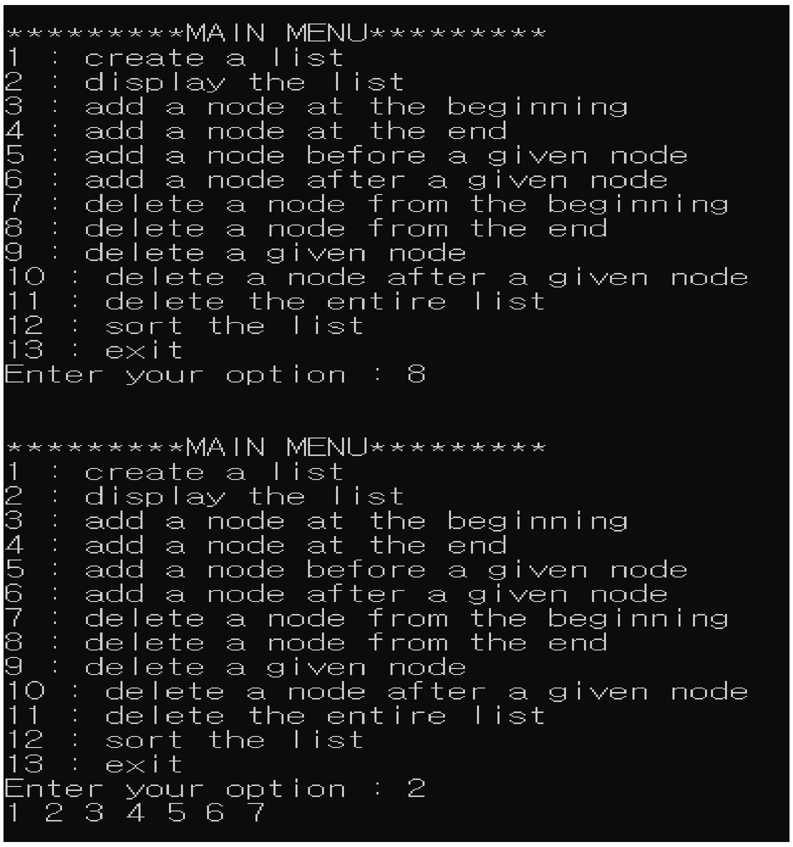
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : 6
Enter the data : 7
Enter the data : -1
linked list created

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 2
1 2 3 4 5 6 7
```



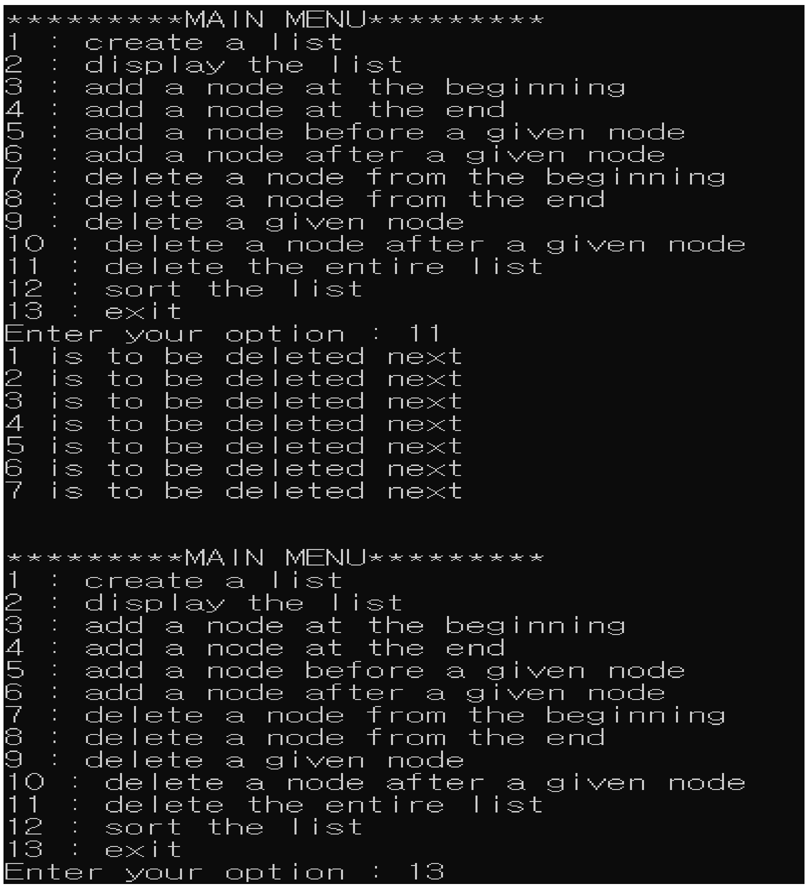
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 3
Enter the data : 0

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 4
Enter the data : 8
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 2
0 1 2 3 4 5 6 7 8

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 7
```



```
*********MAIN MENU*********
1  : create a list
2  : display the list
3  : add a node at the beginning
4  : add a node at the end
5  : add a node before a given node
6  : add a node after a given node
7  : delete a node from the beginning
8  : delete a node from the end
9  : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 8


*********MAIN MENU*********
1  : create a list
2  : display the list
3  : add a node at the beginning
4  : add a node at the end
5  : add a node before a given node
6  : add a node after a given node
7  : delete a node from the beginning
8  : delete a node from the end
9  : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 2
1 2 3 4 5 6 7
```



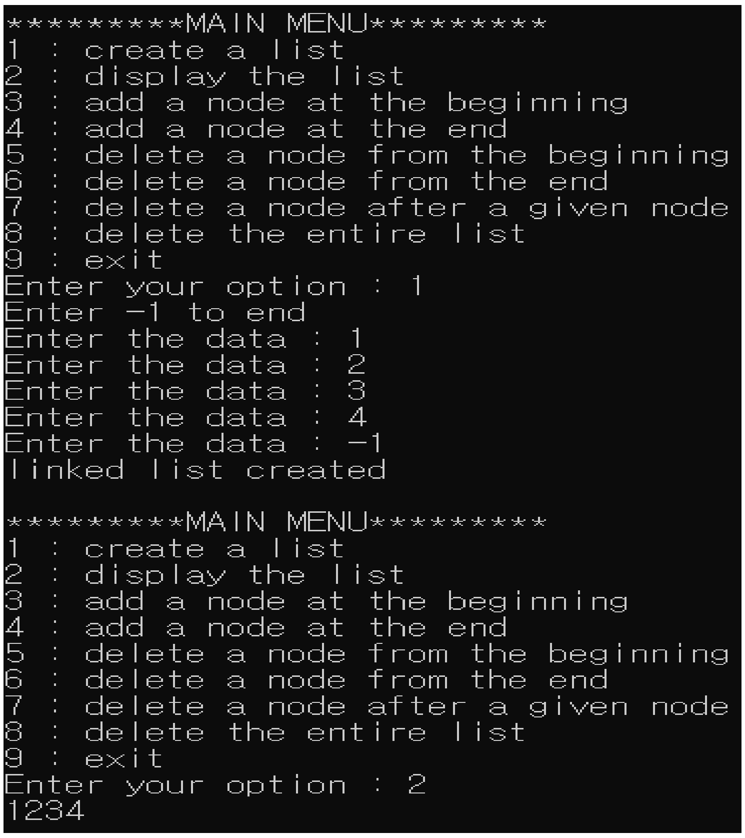
```
*********MAIN MENU*********
1  : create a list
2  : display the list
3  : add a node at the beginning
4  : add a node at the end
5  : add a node before a given node
6  : add a node after a given node
7  : delete a node from the beginning
8  : delete a node from the end
9  : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 11
1 is to be deleted next
2 is to be deleted next
3 is to be deleted next
4 is to be deleted next
5 is to be deleted next
6 is to be deleted next
7 is to be deleted next


*********MAIN MENU*********
1  : create a list
2  : display the list
3  : add a node at the beginning
4  : add a node at the end
5  : add a node before a given node
6  : add a node after a given node
7  : delete a node from the beginning
8  : delete a node from the end
9  : delete a given node
10 : delete a node after a given node
11 : delete the entire list
12 : sort the list
13 : exit
Enter your option : 13
```

2.
```
#include <stdio.h>
#include <malloc.h>

struct node
{
	int data;
	struct node *next;
};

struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_after(struct node *);
struct node *delete_list(struct node *);

int main()
{
	int option;
	do{
	printf("\n");
	printf("\n");
	printf("*********MAIN MENU*********\n");
	printf("1 : create a list \n");
	printf("2 : display the list\n");
	printf("3 : add a node at the beginning\n");
	printf("4 : add a node at the end\n");
	printf("5 : delete a node from the beginning\n");
	printf("6 : delete a node from the end\n");
	printf("7 : delete a node after a given node\n");
	printf("8 : delete the entire list\n");
	printf("9 : exit\n");
	printf("Enter your option : ");
	scanf("%d",&option);
	switch(option)
	{
		case 1: start=create(start);
				printf("linked list created");
				break;
```

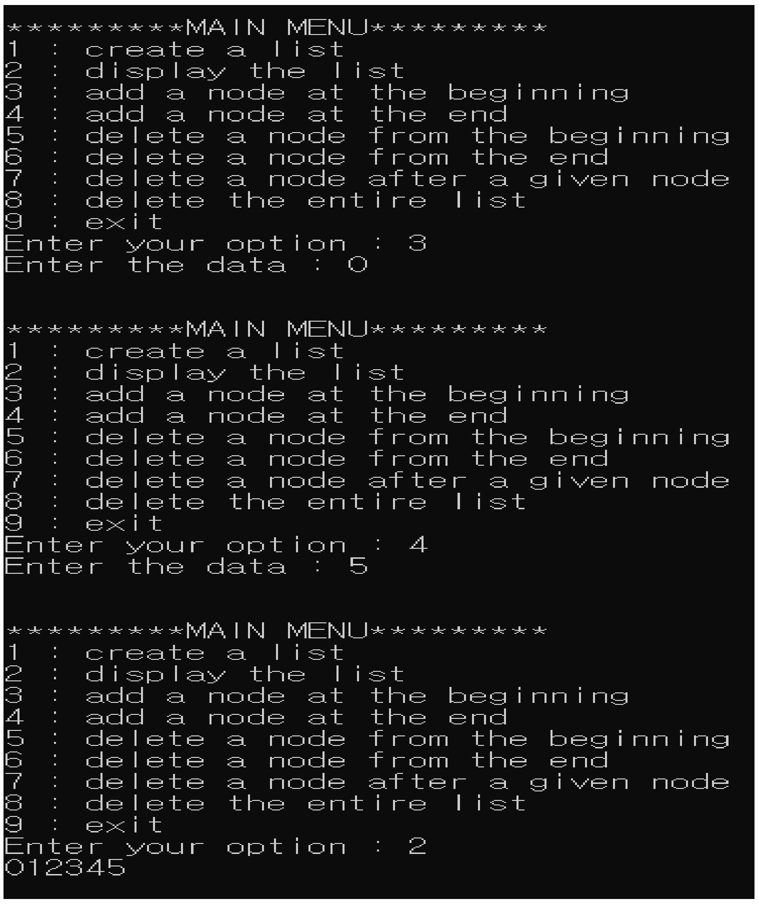
```
                    case 2: start=display(start);
                                        break;

                    case 3: start=insert_beg(start);
                                        break;

                    case 4: start=insert_end(start);
                                   break;

                    case 5: start=delete_beg(start);
                                   break;

                    case 6: start=delete_end(start);
                                        break;


                    case 7: start=delete_after(start);
                                    break;
                    case 8: start=delete_list(start);
                                         break;
          }

          }while(option!=9);
}

struct node *create(struct node *start)
{
          struct node *new_node, *ptr;
          int num;
          printf("Enter -1 to end\n");
          printf("Enter the data : ");
          scanf("%d",&num);
          while(num!=-1)
          {
                    new_node=(struct node*)malloc(sizeof(struct node));
                    new_node->data=num;
                    if(start==NULL)
                    {
                              new_node->next=new_node;
                              start=new_node;
                    }
                    else
```

```
				{
					ptr=start;
					while(ptr->next!=start)
						ptr=ptr->next;
					ptr->next=new_node;
					new_node->next=start;
				}
				printf("Enter the data : ");
				scanf("%d",&num);
		}
		return start;
}

struct node *display(struct node *start)
{
		struct node *ptr;
		ptr=start;
		while(ptr->next!=start)
		{
				printf("%d",ptr->data);
				ptr=ptr->next;
		}
		printf("%d",ptr->data);//마지막값은 짤리기 때문에 따로 추가해준다.
		return start;
}

struct node *insert_beg(struct node *start)
{
		struct node *new_node, *ptr;
		int num;
		printf("Enter the data : ");
		scanf("%d",&num);
		new_node=(struct node *)malloc(sizeof(struct node));
		new_node->data=num;
		new_node->next=start;
		ptr=start;
		while(ptr->next!=start)
				ptr=ptr->next;
		ptr->next=new_node;
		new_node->next=start;
		start=new_node;
		return start;
}
```

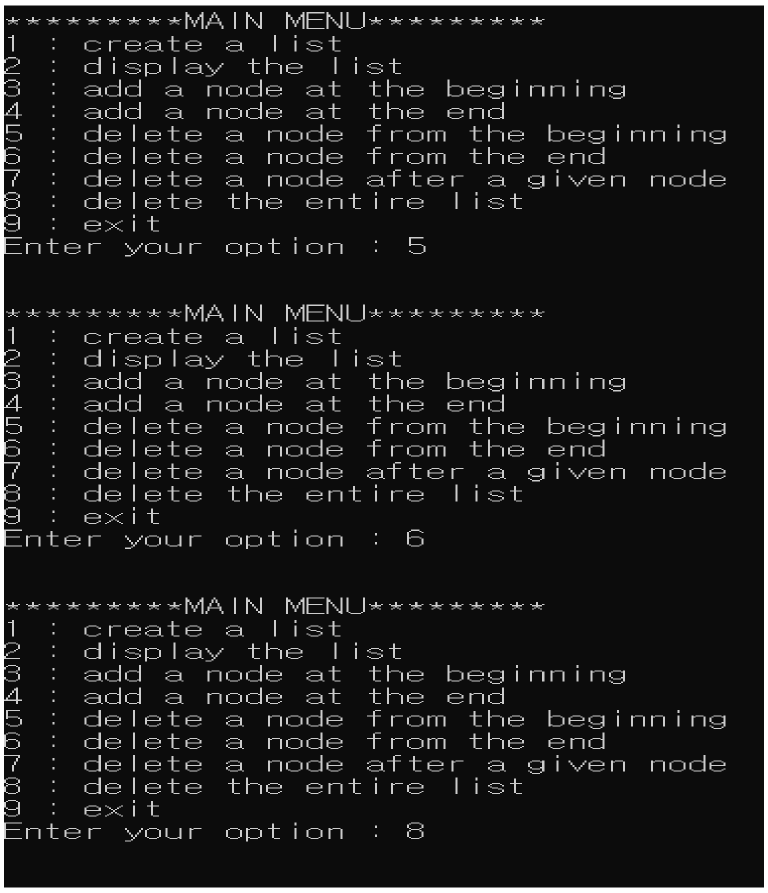
```
struct node *insert_end(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter the data : ");
        scanf("%d",&num);
        new_node=(struct node*)malloc(sizeof(struct node));
        new_node->data=num;
        ptr=start;
        while(ptr->next!=start)
                ptr=ptr->next;
        ptr->next=new_node;
        new_node->next=start;
        start=new_node->next;
        return start;
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
                ptr=ptr->next;
        ptr->next=start->next;
        free(start);
        start=ptr->next;
        return start;
}

struct node *delete_end(struct node *start)
{
        struct node *ptr, *preptr;

        ptr=start;
        while(ptr->next!=start)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=ptr->next;
        free(ptr);
        return start;
}



struct node *delete_after(struct node *start)
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the value after which the node has to be deleted");
        scanf("%d",&val);
        ptr=start;
        preptr=ptr;
        while(preptr->data!=val)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=ptr->next;
        if(ptr==start)
                start=preptr->next;
        free(ptr);
        return start;
}

struct node *delete_list(struct node *start)
{
        struct node *ptr;

        ptr=start;
        while(ptr->next!=start)
        start=delete_end(start);
        free(start);
        return start;
}
```



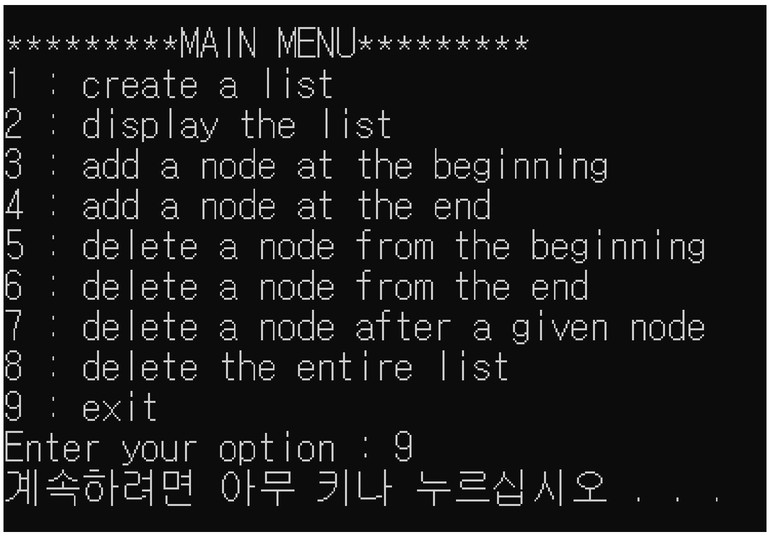
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
linked list created

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 2
1234
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 3
Enter the data : 0


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 4
Enter the data : 5


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 2
012345
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 5

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 6

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 8
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a node after a given node
8 : delete the entire list
9 : exit
Enter your option : 9
계속하려면 아무 키나 누르십시오 . . .
```

3.

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

struct node
{
        int data;
        struct node *next;
        struct node *prev;
};

struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *insert_before(struct node *);
struct node *insert_after(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_before(struct node *);
struct node *delete_after(struct node *);
struct node *delete_list(struct node *);

int main()
{
        int option;
        do{
        printf("\n");
        printf("\n");
        printf("*********MAIN MENU*********\n");
        printf("1 : create a list \n");
        printf("2 : display the list\n");
        printf("3 : add a node at the beginning\n");
        printf("4 : add a node at the end\n");
        printf("5 : add a node before a given node\n");
        printf("6 : add a node after a given node\n");
        printf("7 : delete a node from the beginning\n");
        printf("8 : delete a node from the end\n");
        printf("9 : delete a node before a given node\n");
        printf("10 : delete a node after a given node\n");
        printf("11 : delete the entire list\n");
printf("12 : exit\n");
printf("Enter your option : ");
scanf("%d",&option);
switch(option)
{
        case 1: start=create(start);
                        printf("linked list created");
                        break;

        case 2: start=display(start);
                        break;

        case 3: start=insert_beg(start);
                        break;

        case 4: start=insert_end(start);
                    break;

        case 5: start=insert_before(start);
                    break;

        case 6: start=insert_after(start);
                    break;

        case 7: start=delete_beg(start);
                    break;

        case 8: start=delete_end(start);
                        break;

        case 9: start=delete_before(start);
                        break;

        case 10: start=delete_after(start);
                     break;
        case 11: start=delete_list(start);
                         break;



}

}while(option!=12);
```

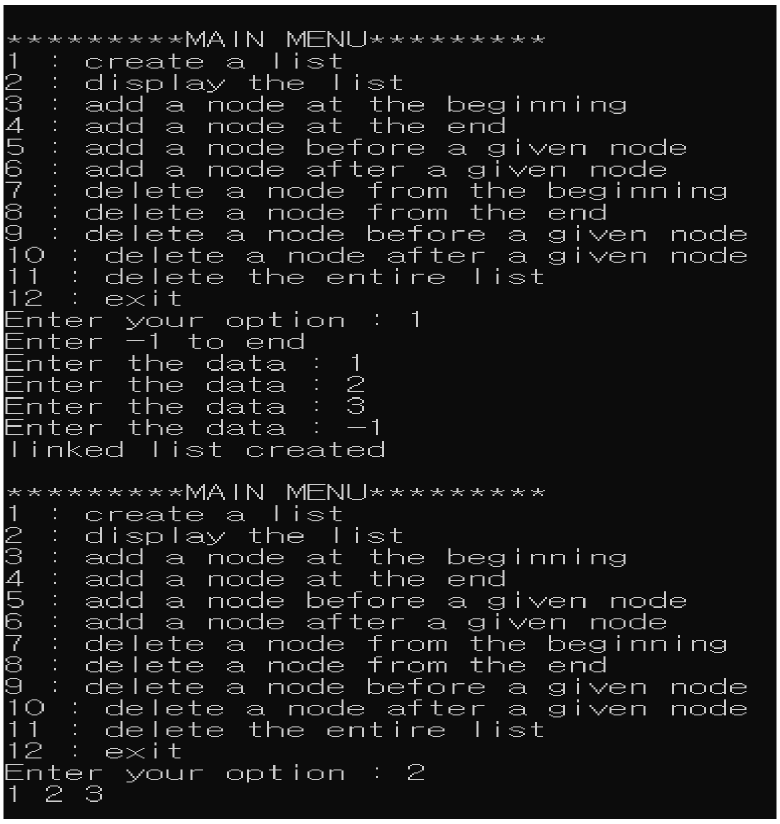
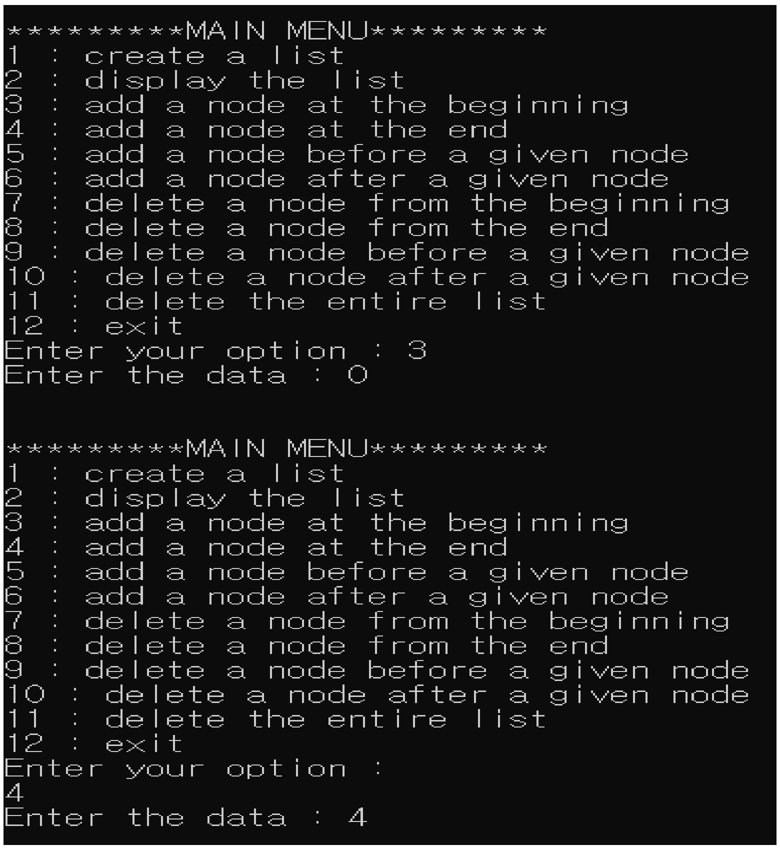
```
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->prev=NULL;
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
			new_node->prev=ptr;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d ",ptr->data);
		ptr=ptr->next;
	}
```

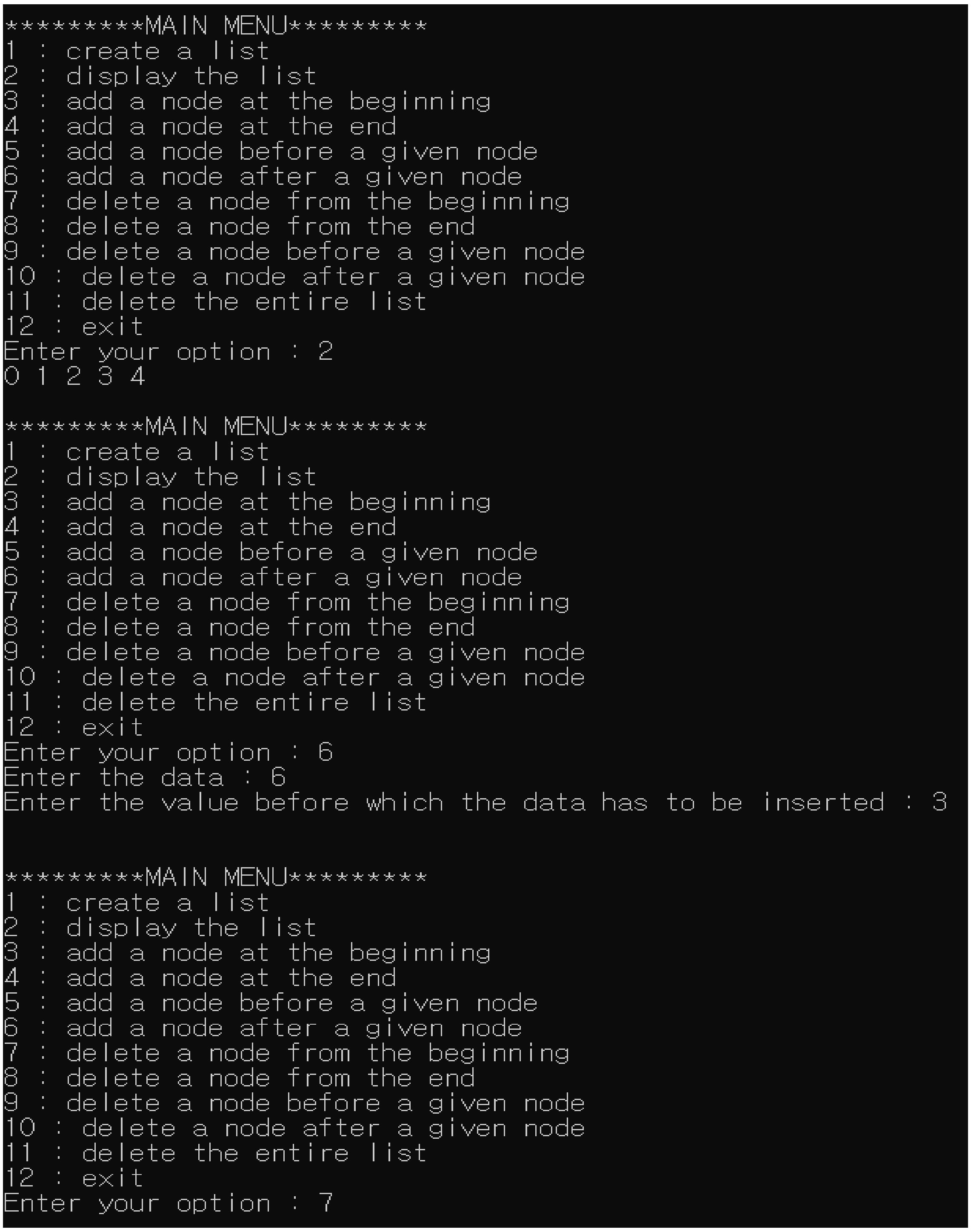
```
		return start;
}

struct node *insert_beg(struct node *start)
{
		struct node *new_node;
		int num;
		printf("Enter the data : ");
		scanf("%d",&num);
		new_node=(struct node *)malloc(sizeof(struct node));
		new_node->data=num;
		start->prev=new_node;
		new_node->next=start;
		new_node->prev=NULL;
		start=new_node;
		return start;
}

struct node *insert_end(struct node *start)
{
		struct node *new_node, *ptr;
		int num;
		printf("Enter the data : ");
		scanf("%d",&num);
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		ptr=start;
		while(ptr->next!=NULL)
			ptr=ptr->next;
		ptr->next=new_node;
		new_node->next=NULL;
		new_node->prev=ptr;
		return start;
}

struct node *insert_before(struct node *start)
{
		struct node *new_node, *ptr;
		int num,val;
		printf("Enter the data : ");
		scanf("%d",&num);
		printf("Enter the value before which the data has to be inserted : ");
		scanf("%d",&val);
```

```
          new_node=(struct node*)malloc(sizeof(struct node));
          new_node->data=num;
     ptr=start;
          while(ptr->data!=val)
          {
                    ptr=ptr->next;
          }
          new_node->next=ptr;
          new_node->prev=ptr->prev;
          ptr->prev->next=new_node;
          ptr->prev=new_node;
          return start;
}

struct node *insert_after(struct node *start)
{
          struct node *new_node, *ptr;
          int num,val;
          printf("Enter the data : ");
          scanf("%d",&num);
          printf("Enter the value before which the data has to be inserted : ");
          scanf("%d",&val);
          new_node=(struct node *)malloc(sizeof(struct node));
          new_node->data=num;
          ptr=start;
          while(ptr->data!=val)
          ptr=ptr->next;
          new_node->prev=ptr;
          new_node->next=ptr->next;
          ptr->next->prev=new_node;
          ptr->next=new_node;
          return start;
}

struct node *delete_beg(struct node *start)
{
          struct node *ptr;
          ptr=start;
          start=start->next;
          start->prev=NULL;
          free(ptr);
          return start;
}
```

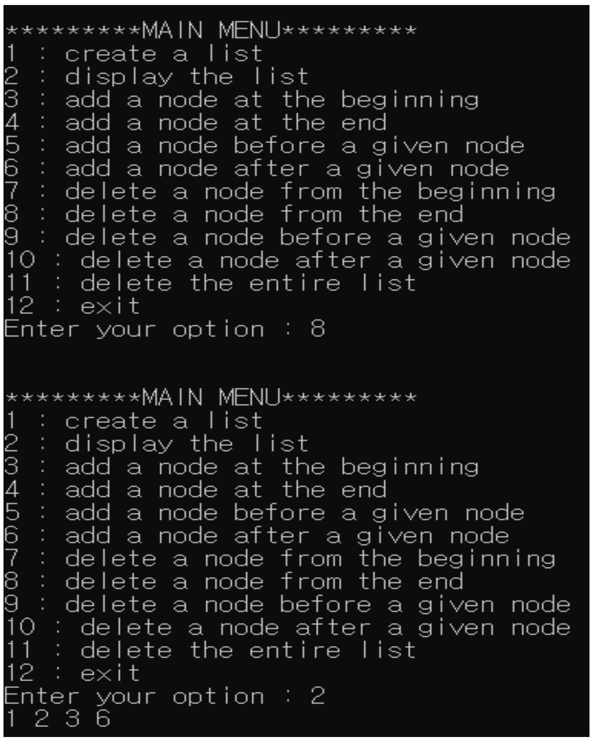
```
struct node *delete_end(struct node *start)
{
          struct node *ptr;

          ptr=start;
          while(ptr->next!=NULL)
                    ptr=ptr->next;
          ptr->prev->next=NULL;
          free(ptr);
          return start;
}

struct node *delete_before(struct node *start)
{
          struct node *ptr, *temp;
          int val;
          printf("Enter the value of the node which has to be deleted : ");
          scanf("%d",&val);
          ptr=start;
          while(ptr->data!=val)
                    ptr=ptr->next;
          temp=ptr->prev;
          if(temp==start)
                    start=delete_beg(start);
          else
          {
                    ptr->prev=temp->prev;
                    temp->prev->next=ptr;
          }
          free(temp);
          return start;
}

struct node *delete_after(struct node *start)
{
          struct node *ptr,*temp;
          int val;
          printf("Enter the value after which the node has to be deleted");
          scanf("%d",&val);
          ptr=start;
          while(ptr->data!=val)
                    ptr=ptr->next;
```

```
        temp=ptr->next;
        ptr->next=temp->next;
        temp->next->prev=ptr;
        free(temp);
        return start;
}

struct node *delete_list(struct node *start)
{
        while(start->next !=NULL)
                start=delete_beg(start);
        free(start);
        start=NULL;
        return start;
}
```

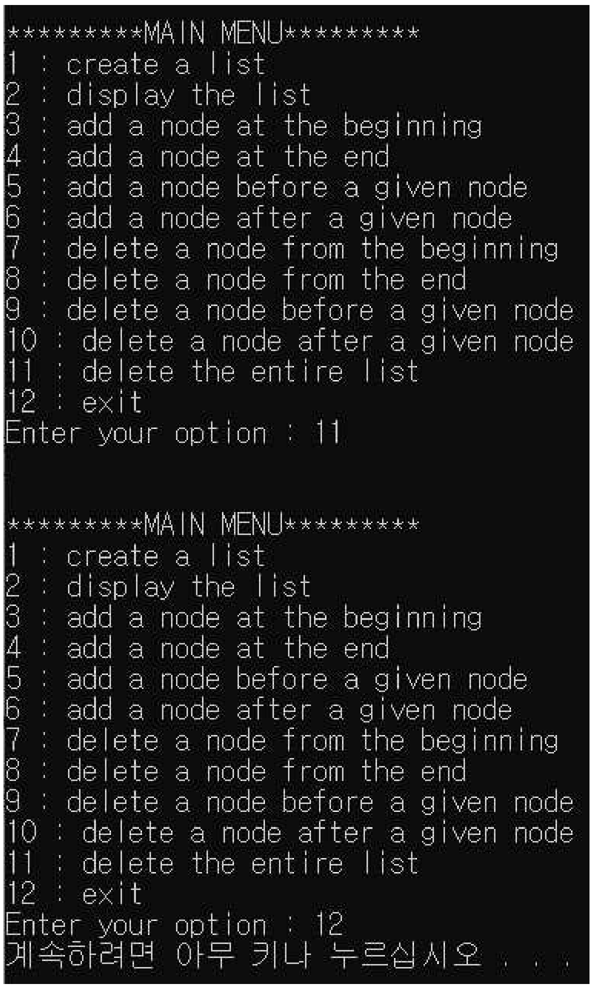

```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
linked list created

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 2
1 2 3
```

```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 3
Enter the data : 0


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option :
4
Enter the data : 4
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 2
0 1 2 3 4

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 6
Enter the data : 6
Enter the value before which the data has to be inserted : 3


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 7
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 8

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 2
1 2 3 6
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 11


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : add a node before a given node
6 : add a node after a given node
7 : delete a node from the beginning
8 : delete a node from the end
9 : delete a node before a given node
10 : delete a node after a given node
11 : delete the entire list
12 : exit
Enter your option : 12
계속하려면 아무 키나 누르십시오 . . .
```

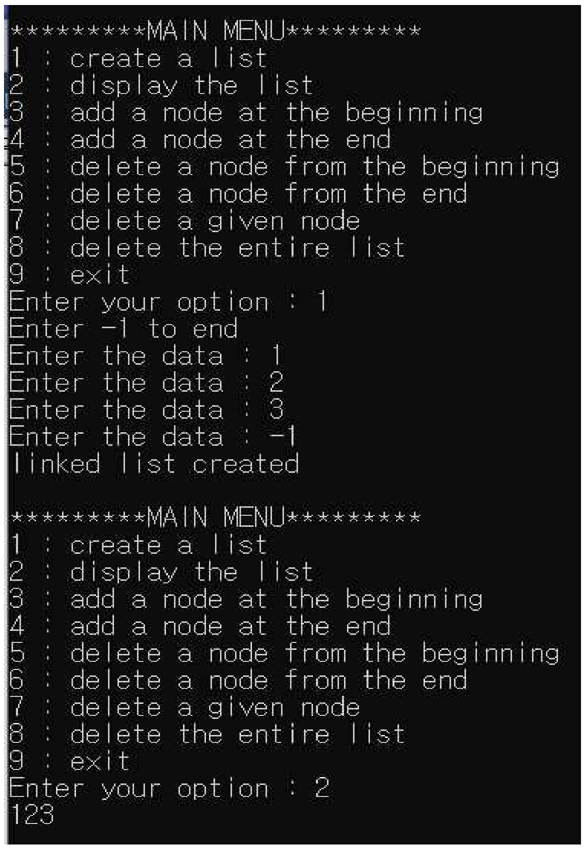
4.
```c
#include <stdio.h>
#include <malloc.h>

struct node
{
	int data;
	struct node *next;
	struct node *prev;
};

struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_node(struct node *);
struct node *delete_list(struct node *);

int main()
{
	int option;
	do{
	printf("\n");
	printf("\n");
	printf("*********MAIN MENU*********\n");
	printf("1 : create a list \n");
	printf("2 : display the list\n");
	printf("3 : add a node at the beginning\n");
	printf("4 : add a node at the end\n");
	printf("5 : delete a node from the beginning\n");
	printf("6 : delete a node from the end\n");
	printf("7 : delete a given node\n");
	printf("8 : delete the entire list\n");
	printf("9 : exit\n");
	printf("Enter your option : ");
	scanf("%d",&option);
	switch(option)
	{
		case 1: start=create(start);
				printf("linked list created");
```

```
                        break;

                case 2: start=display(start);
                        break;

                case 3: start=insert_beg(start);
                        break;

                case 4: start=insert_end(start);
                        break;

                case 5: start=delete_beg(start);
                        break;

                case 6: start=delete_end(start);
                        break;

                case 7: start=delete_node(start);
                        break;

                case 8: start=delete_list(start);
                        break;

        }

        }while(option!=9);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num !=-1)
        {

                if(start == NULL)
                {
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->prev=NULL;
                        new_node->data=num;
                        new_node->next=new_node;//start로 설정하면 null값이 들어간다.
                        start=new_node;
                }
                else
                {
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->data=num;
                        ptr=start;
                        while(ptr->next!=start)
                        ptr=ptr->next;
                        new_node->next=start;
                        start->prev=new_node;
                        new_node->prev=ptr;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;
}

struct node *insert_beg(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter the data : ");
        scanf("%d",&num);
        new_node=(struct node *)malloc(sizeof(struct node));
        ptr=start;
        while(ptr->next!=start)
```

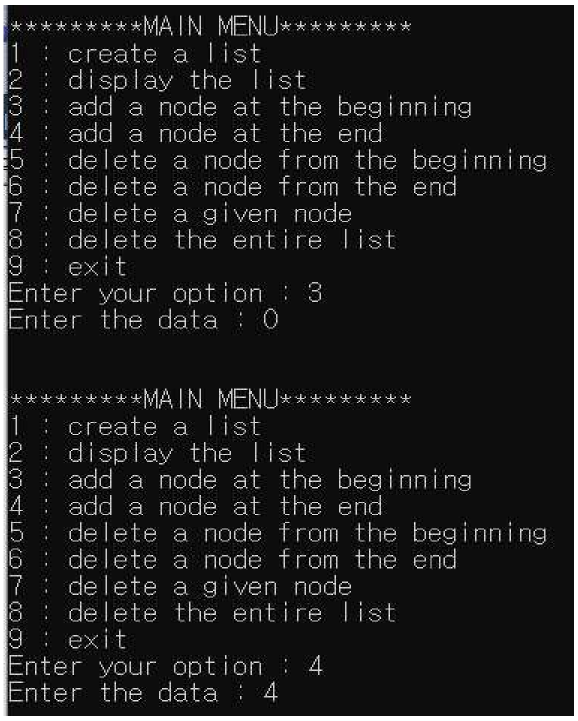
```
                ptr=ptr->next;
        new_node->data=num;

        ptr->next=new_node;
        new_node->next=start;
        start->prev=new_node;
        new_node->prev=ptr;
        start=new_node;
        return start;
}

struct node *insert_end(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter the data : ");
        scanf("%d",&num);
        new_node=(struct node*)malloc(sizeof(struct node));
        new_node->data=num;
        ptr=start;
        while(ptr->next!=start)
                ptr=ptr->next;
        ptr->next=new_node;
        new_node->next=start;
        new_node->prev=ptr;
        start->prev=new_node;
        return start;
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr,*temp;
        ptr=start;
        while(ptr->next!=start)
                ptr=ptr->next;
        ptr->next=start->next;
        temp=start;
        start=start->next;
        start->prev=ptr;
        free(temp);
        return start;
}
```

```
struct node *delete_end(struct node *start)
{
          struct node *ptr;
          ptr=start;

          while(ptr->next!=start)
                    ptr=ptr->next;
          ptr->prev->next=start;
          start->prev=ptr->prev;
          free(ptr);
          return start;
}

struct node *delete_node(struct node *start)
{
          struct node *ptr;
          int val;
          printf("Enter the value of the node which has to be deleted : ");
          scanf("%d",&val);
          ptr=start;

          if(ptr->data==val)
          {
                    start=delete_beg(start);
                    return start;
          }
          else
          {
                    while(ptr->data!=val)
                              ptr=ptr->next;
                    ptr->prev->next=ptr->next;
                    ptr->next=ptr->prev;
          }
          free(ptr);
          return start;
}


struct node *delete_list(struct node *start)
{
          struct node *ptr;
          ptr=start;
          while(ptr->next!=start)
```

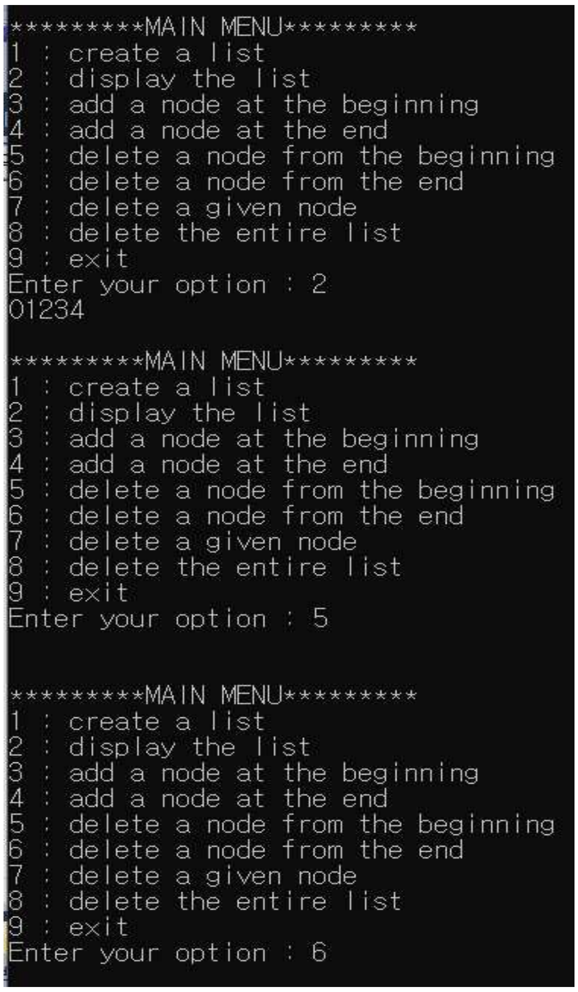
```
                start=delete_end(start);
        free(start);
        return start;
}
```



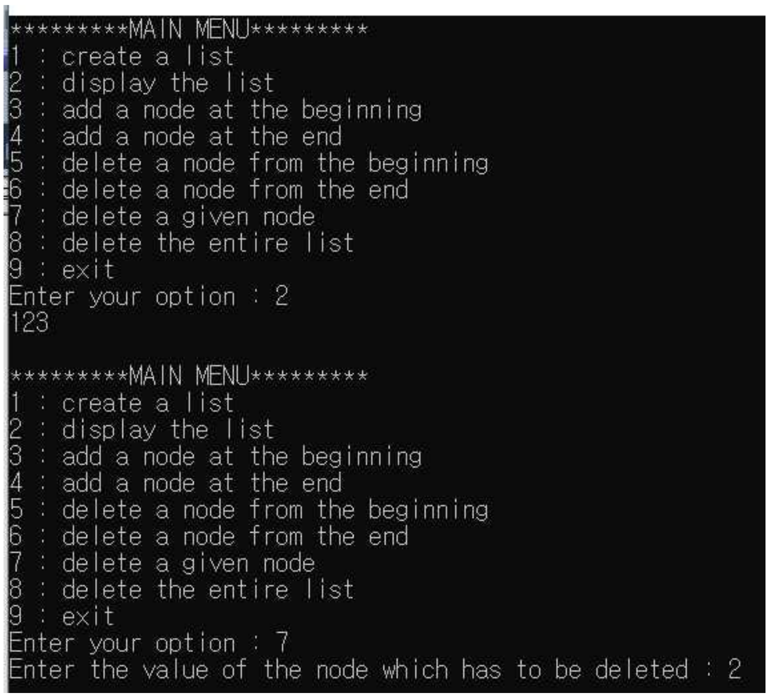
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
linked list created

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 2
123
```



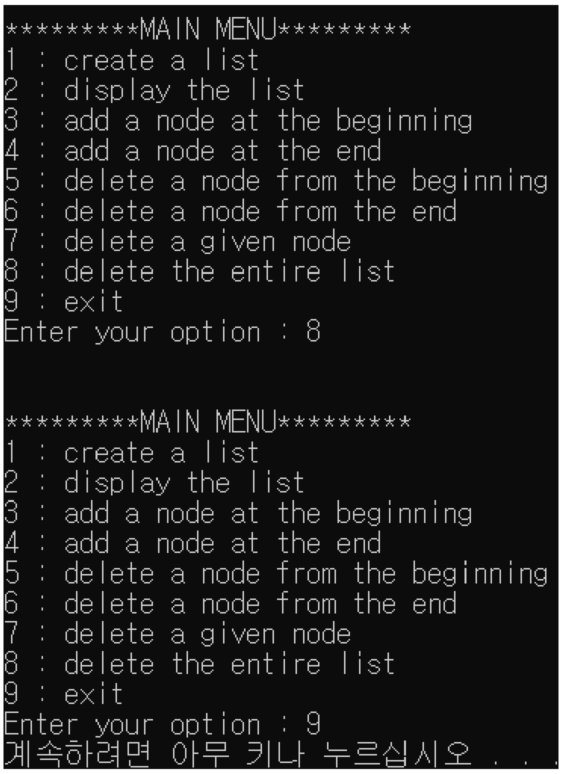
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 3
Enter the data : 0


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 4
Enter the data : 4
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 2
01234

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 5


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 6
```



```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 2
123

*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 7
Enter the value of the node which has to be deleted : 2
```



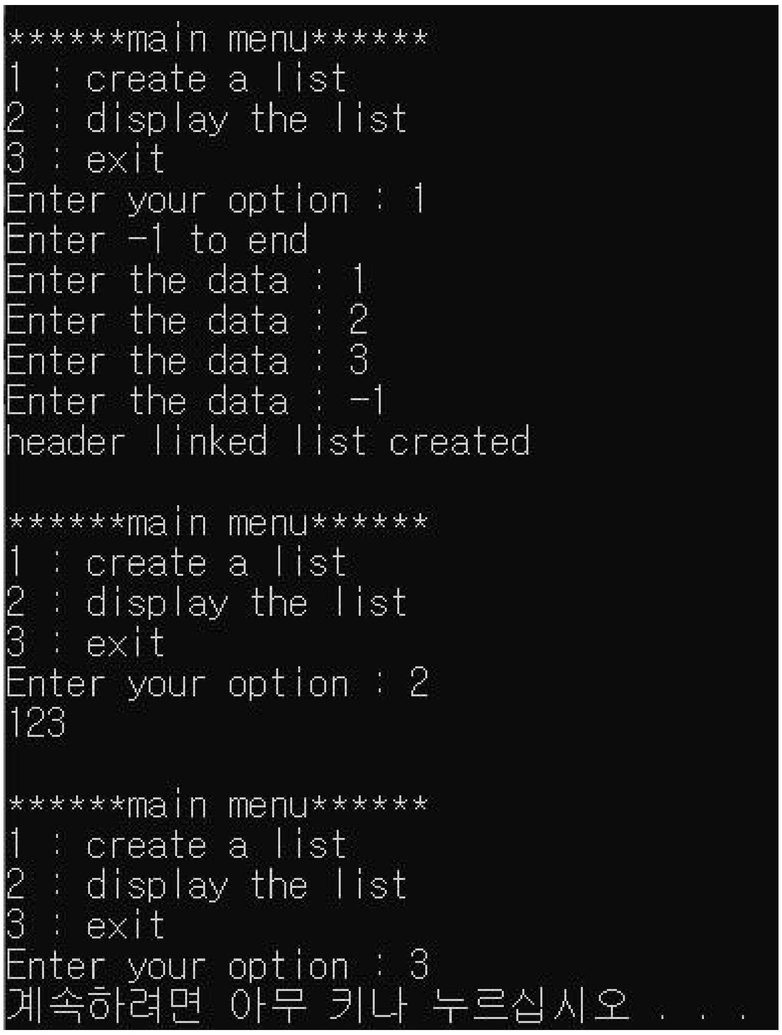
```
*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 8


*********MAIN MENU*********
1 : create a list
2 : display the list
3 : add a node at the beginning
4 : add a node at the end
5 : delete a node from the beginning
6 : delete a node from the end
7 : delete a given node
8 : delete the entire list
9 : exit
Enter your option : 9
계속하려면 아무 키나 누르십시오 . . .
```

5.

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

struct node {
	int data;
	struct node *next;
};

struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);

int main()
{
	int option;

	do
	{	printf("\n\n");
		printf("******main menu******\n");
		printf("1 : create a list\n");
		printf("2 : display the list\n");
		printf("3 : exit\n");
		printf("Enter your option : ");
```

```
                scanf("%d",&option);
                switch(option)
                {
                        case 1: start=create(start);
                                        printf("header linked list created");
                                        break;

                        case 2: start= display(start);
                                        break;
                }

        }while(option !=3);

}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                new_node->next=NULL;
                if(start==NULL)
                {
                        start=(struct node*)malloc(sizeof(struct node));
                        start->next=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
```

```
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start->next;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
```



```
******main menu******
1 : create a list
2 : display the list
3 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
header linked list created

******main menu******
1 : create a list
2 : display the list
3 : exit
Enter your option : 2
123

******main menu******
1 : create a list
2 : display the list
3 : exit
Enter your option : 3
계속하려면 아무 키나 누르십시오 . . .
```

6.

```
#include <stdio.h>
#include <malloc.h>
struct node
```

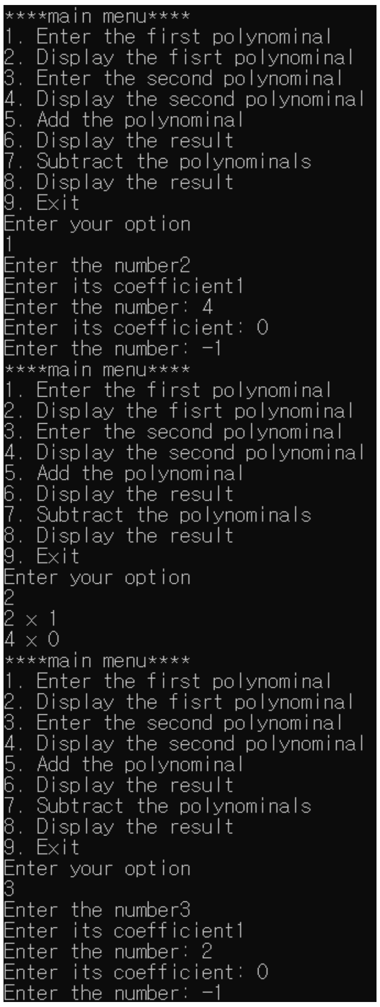
```
{
        int num;
        int coeff;//계수
        struct node *next;
};
struct node *start1=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *start4=NULL;
struct node *create_poly(struct node *);
struct node *display_poly(struct node *);
struct node *sub_poly(struct node *,struct node *, struct node *);
struct node *add_poly(struct node *,struct node *, struct node *);
struct node *add_node(struct node *,struct node *, struct node *);
int main()
{
        int option;
        do
        {
        printf("****main menu****\n");
        printf("1. Enter the first polynominal\n");
        printf("2. Display the fisrt polynominal\n");
        printf("3. Enter the second polynominal\n");
        printf("4. Display the second polynominal\n");
        printf("5. Add the polynominal\n");
        printf("6. Display the result\n");
        printf("7. Subtract the polynominals\n");
        printf("8. Display the result\n");
        printf("9. Exit\n");
        printf("Enter your option\n");
        scanf("%d",&option);
        switch(option)
                {
                        case 1: start1=create_poly(start1);
                                        break;
                        case 2: start1=display_poly(start1);
                                        break;
                        case 3: start2=create_poly(start2);
                                        break;
                        case 4: start2=display_poly(start2);
                                        break;
                        case 5: start3=add_poly(start1,start2,start3);
                                        break;
```

```
                    case 6: start3=display_poly(start3);
                                break;
                    case 7: start4=sub_poly(start1,start2,start4);
                                break;
                    case 8: start4=display_poly(start4);
                                break;

                }
        }while(option!=9);
}


struct node *create_poly(struct node *start)
{
        struct node *new_node, *ptr;
        int n,c;
        printf("Enter the number");
        scanf("%d",&n);
        printf("Enter its coefficient");
        scanf("%d",&c);
        while(n!=-1)
        {
                if(start==NULL)
                {
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->num=n;
                        new_node->coeff=c;
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->num=n;
                        new_node->coeff=c;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the number: ");
                scanf("%d",&n);
                if(n==-1)
                        break;
                printf("Enter its coefficient: ");
                scanf("%d",&c);
        }
        return start;
}

struct node *display_poly(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d x %d \n",ptr->num,ptr->coeff);
                ptr=ptr->next;
        }
        return start;
}

struct node *add_node(struct node *start, int n, int c)
{
        struct node *ptr, *new_node;
        if(start==NULL)
        {
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node->num=n;
                new_node->coeff=c;
                new_node->next=NULL;
                start=new_node;
        }
        else
        {
                ptr=start;
                while(ptr->next!=NULL)
                        ptr=ptr->next;
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node->num=n;
                new_node->coeff=c;
                new_node->next=NULL;
                ptr->next=new_node;
        }
        return start;
```

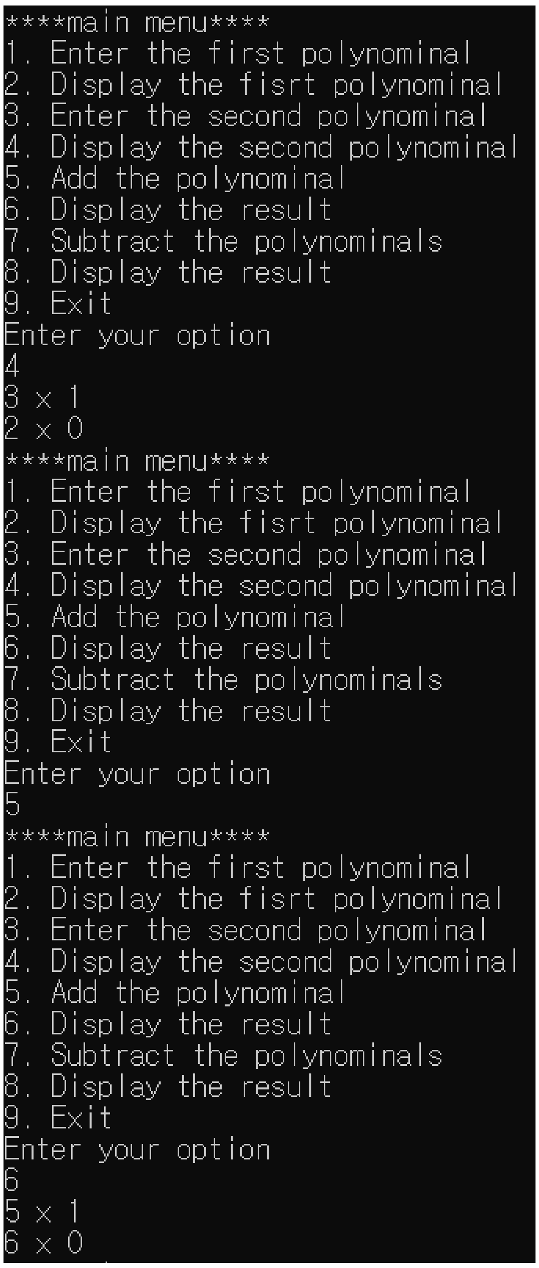
```
}
struct node *add_poly(struct node *start1, struct node *start2, struct node *start3)
{
        struct node *ptr1, *ptr2;
        int sum_num,c;
        ptr1=start1;
        ptr2=start2;
        while(ptr1!=NULL && ptr2!= NULL)
        {
                if(ptr1->coeff==ptr2->coeff)
                {
                        sum_num=ptr1->num+ptr2->num;
                        start3 = add_node(start3,sum_num,ptr1->coeff);
                        ptr1=ptr1->next;
                        ptr2=ptr2->next;
                }
                else if(ptr1->coeff > ptr2->coeff)
                {
                        start3= add_node(start3,ptr1->num,ptr1->coeff);
                        ptr1=ptr1->next;
                }
                else if(ptr1->coeff < ptr2->coeff)
                {
                        start3= add_node(start3,ptr2->num,ptr2->coeff);
                        ptr2=ptr2->next;
                }
        }
        if(ptr1=NULL)
        {
                while(ptr2!=NULL)
                {
                        start3=add_node(start3,ptr2->num,ptr2->coeff);
                }
        }
        if(ptr2==NULL)
        {
                while(ptr1!=NULL)
                {
                        start3=add_node(start3, ptr1->num, ptr1->coeff);
                        ptr1=ptr1->next;
                }
        }
        return start3;
```

```
}

struct node *sub_poly(struct node *start1,struct node *start2,struct node *start4)
{
        struct node *ptr1, *ptr2;
        int sub_num, c;
        ptr1=start1;
        ptr2=start2;
        while(ptr1!=NULL && ptr2!=NULL)
        {
                if(ptr1->coeff==ptr2->coeff)
                {
                sub_num=ptr1->num-ptr2->num;
                start4=add_node(start4,sub_num,ptr1->coeff);
                ptr1=ptr1->next;
                ptr2=ptr2->next;
                }
                else if(ptr1->coeff> ptr2->coeff)
                {
                        start4=add_node(start4,ptr1->num,ptr1->coeff);
                        ptr1=ptr1->next;
                }
                else if(ptr1->coeff<ptr2->coeff)
                {
                        start4=add_node(start4,ptr2->num,ptr2->coeff);
                        ptr2=ptr2->next;
                }
        }
        if(ptr1==NULL)
        {
                while(ptr2!=NULL)
                {
                        start4=add_node(start4, ptr2->num, ptr2->coeff);
                        ptr2=ptr2->next;
                }
        }
        if(ptr2==NULL)
        {
                while(ptr1!=NULL)
                {
                        start4=add_node(start4, ptr1->num, ptr1->coeff);
                        ptr1=ptr1->next;
                }
```

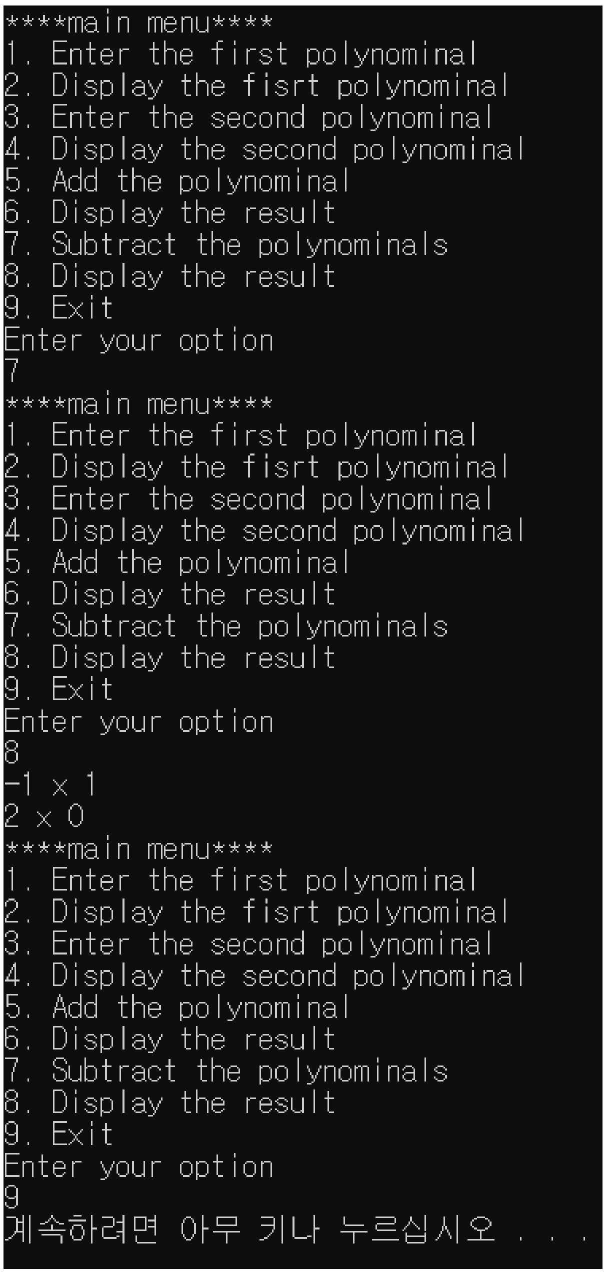
```
		}
		return start4;
	}
```



```
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
1
Enter the number2
Enter its coefficient1
Enter the number: 4
Enter its coefficient: 0
Enter the number: -1
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
2
2 x 1
4 x 0
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
3
Enter the number3
Enter its coefficient1
Enter the number: 2
Enter its coefficient: 0
Enter the number: -1
```



```
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
4
3 x 1
2 x 0
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
5
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
6
5 x 1
6 x 0
```

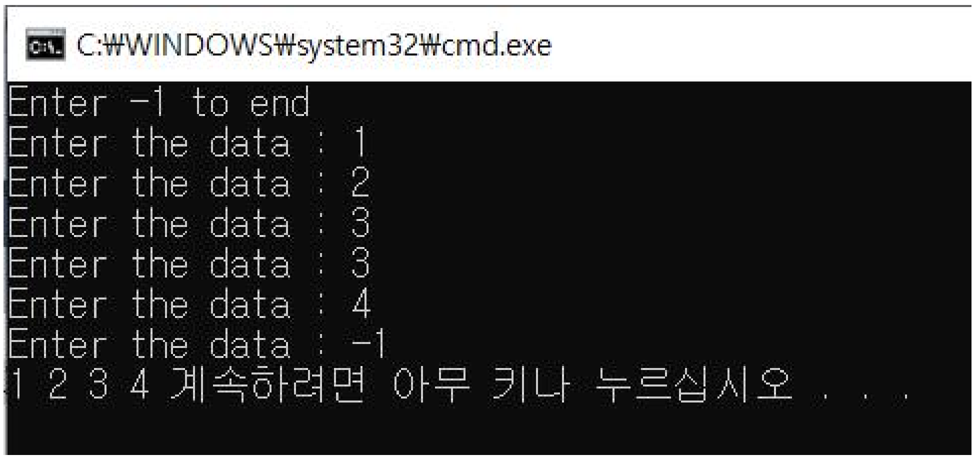


```
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
7
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
8
-1 x 1
2 x 0
****main menu****
1. Enter the first polynominal
2. Display the fisrt polynominal
3. Enter the second polynominal
4. Display the second polynominal
5. Add the polynominal
6. Display the result
7. Subtract the polynominals
8. Display the result
9. Exit
Enter your option
9
계속하려면 아무 키나 누르십시오 . . .
```

Programming Exercises

1.

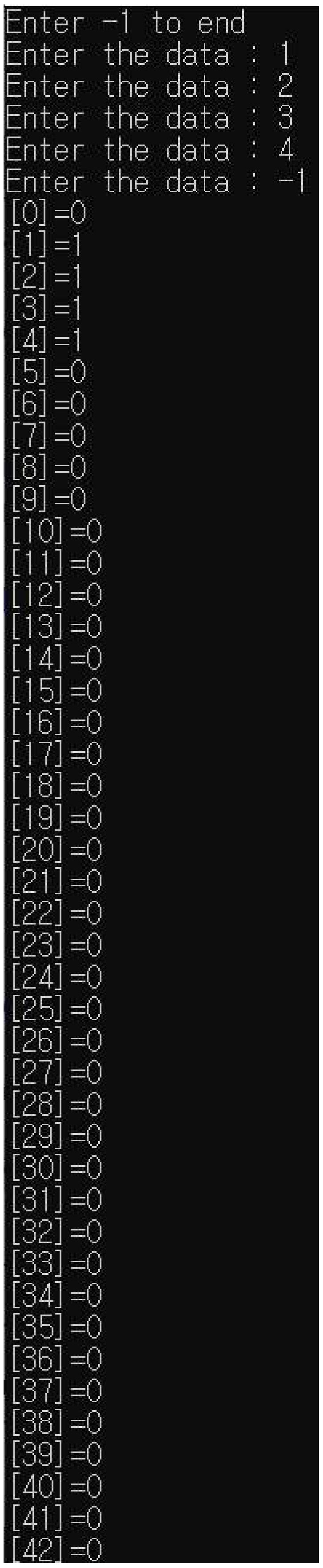
```
#include <stdio.h>
#include <malloc.h>
struct node {
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *);
struct node *check(struct node *);
struct node *display(struct node *);
int main ()
{

        start=create(start);
        start=check(start);
        start=display(start);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}
struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
struct node *check(struct node *start)
{
        struct node *ptr,*temp;
        ptr=start;

        while(ptr->next!=NULL)
        {
                if(ptr->data==ptr->next->data)
                {
                        temp=ptr->next;
                        if(temp->next==NULL)
                        {
                                ptr->next=NULL;
                                free(temp);
                        }
                        else
                        {
                        ptr->next=temp->next;
                        ptr=temp->next;
                        free(temp);
                        }
                }
                else
                ptr=ptr->next;
        }
        return start; }
```



```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 3
Enter the data : 4
Enter the data : -1
1 2 3 4 계속하려면 아무 키나 누르십시오 . . .
```

2.

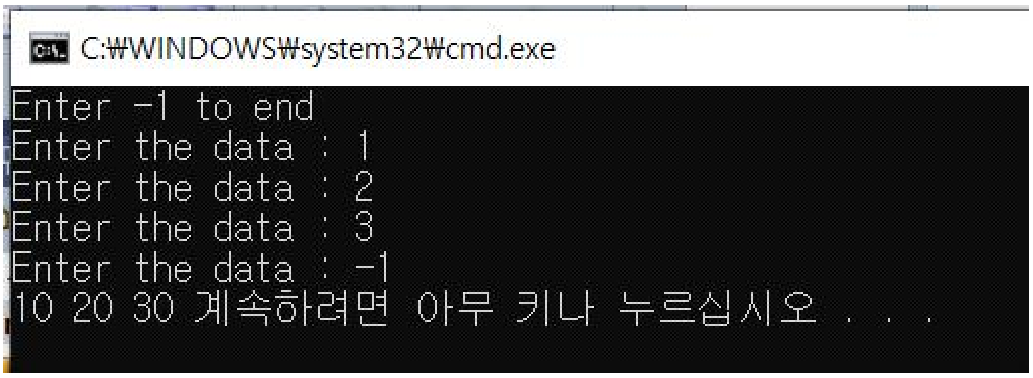
```
#include <stdio.h>
#include <malloc.h>
struct node {
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *);
void check(struct node *start);
int main ()
{

        start=create(start);
        check(start);

}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->next=NULL;
                start=new_node;
            }
            else
            {
                ptr=start;
                while(ptr->next!=NULL)
                    ptr=ptr->next;
                new_node->next=NULL;
                ptr->next=new_node;
            }
            printf("Enter the data : ");
            scanf("%d",&num);
        }
        return start;
}

void check(struct node *start)
{
        struct node *ptr;
        int count[100]={0};
        int i;
        ptr=start;
        while(ptr!=NULL)
        {
            for(i=0;i<100;i++)
            {
                if(ptr->data==i)
                    count[i]++;
            }
            ptr=ptr->next;
        }
        for(i=0;i<100;i++)
        printf("[%d]=%d\n",i,count[i]);

}
```


```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
[0]=0
[1]=1
[2]=1
[3]=1
[4]=1
[5]=0
[6]=0
[7]=0
[8]=0
[9]=0
[10]=0
[11]=0
[12]=0
[13]=0
[14]=0
[15]=0
[16]=0
[17]=0
[18]=0
[19]=0
[20]=0
[21]=0
[22]=0
[23]=0
[24]=0
[25]=0
[26]=0
[27]=0
[28]=0
[29]=0
[30]=0
[31]=0
[32]=0
[33]=0
[34]=0
[35]=0
[36]=0
[37]=0
[38]=0
[39]=0
[40]=0
[41]=0
[42]=0
```

3.

```c
#include <stdio.h>
#include <malloc.h>
struct node {
	int data;
	struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *);
struct node *multi(struct node *);
struct node *display(struct node *);
int main ()
{

	start=create(start);
	start=multi(start);
	start=display(start);

}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			new_node->next=NULL;
			ptr->next=new_node;
```

```
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}
struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
struct node *multi(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                ptr->data=ptr->data*10;
                ptr=ptr->next;
        }
        return start;
}
```

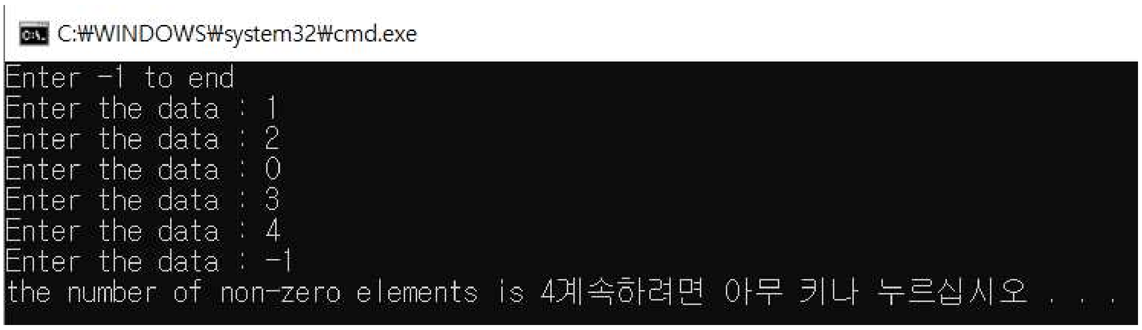


```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
10 20 30 계속하려면 아무 키나 누르십시오 . . .
```

4.

```
#include <stdio.h>
#include <malloc.h>
struct node {
        int data;
        struct node *next;
```

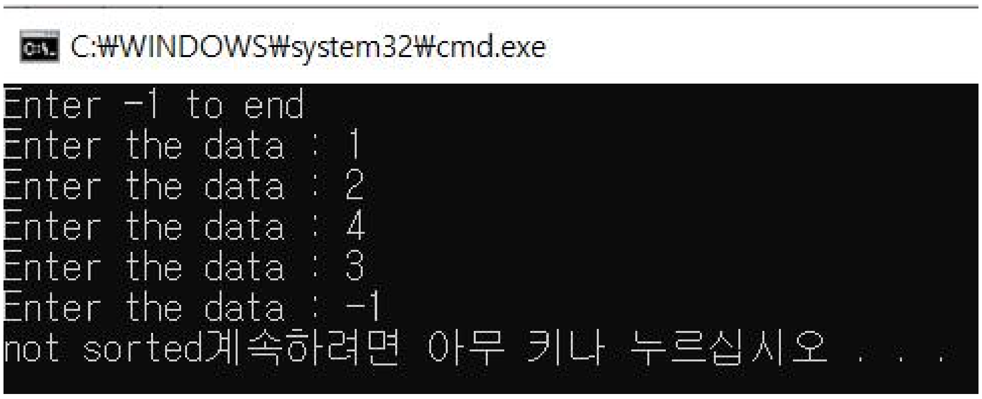
```c
};
struct node *start=NULL;
struct node *create(struct node *);
void multi(struct node *start);
int main ()
{

        start=create(start);
        multi(start);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

void multi(struct node *start)
{
        struct node *ptr;
        int count=0;
        ptr=start;
        while(ptr!=NULL)
        {
                if(ptr->data!=0)
                        count++;
                ptr=ptr->next;
        }
        printf("the number of non-zero elements is %d",count);
}
```



```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 0
Enter the data : 3
Enter the data : 4
Enter the data : -1
the number of non-zero elements is 4계속하려면 아무 키나 누르십시오 . . .
```

5.

```
#include <stdio.h>
#include <malloc.h>
struct node {
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *);
void check(struct node *start);
int main ()
{

        start=create(start);
        check(start);



}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}
void check(struct node *start)
{
        struct node *ptr;
        int flag=1;
        ptr=start;
        while(ptr->next!=NULL)
        {
                if(ptr->data < ptr->next->data)
                        flag=1;
                else
                {
                        flag=0;
                        break;
                }
                ptr=ptr->next;
        }
        if(flag==1)printf("sorted");
        else printf("not sorted");
}
```


C:₩WINDOWS₩system32₩cmd.exe

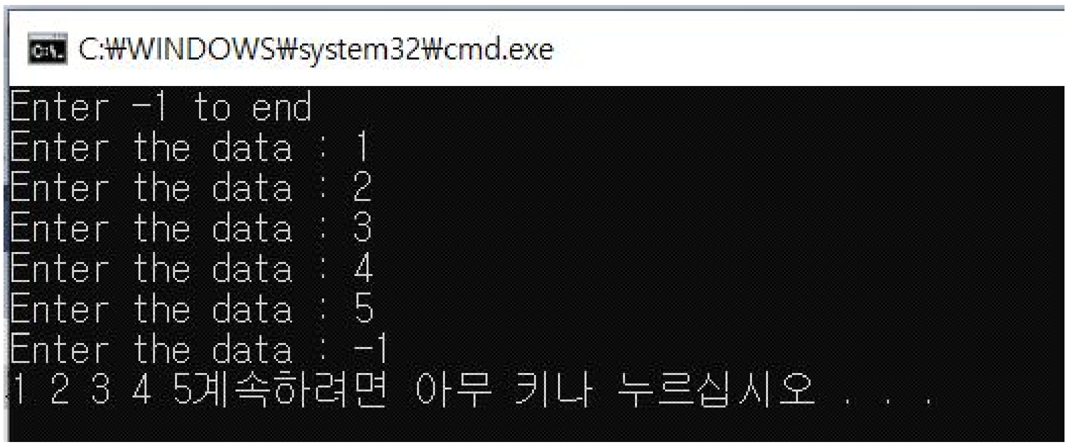
```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 4
Enter the data : 3
Enter the data : -1
not sorted계속하려면 아무 키나 누르십시오 . . .
```

6.

```
#include <stdio.h>
#include <malloc.h>
struct node {
	int data;
	struct node *next;
};
struct save {
	struct node *ad1;
	struct node *ad2;
};
struct node *start=NULL;
struct node *start2=NULL;
struct save create(struct node *start,struct node *start2);
struct node *display(struct node *start);
struct node *add_node(struct node *start2);
struct node *copy(struct node *start, struct node *start2);
int main ()
{
	struct save s1;
	s1=create(start,start2);
	s1.ad2=copy(s1.ad1,s1.ad2);
	s1.ad2=display(s1.ad2);




}

struct save create(struct node *start,struct node *start2)
{
	struct save s1;
	struct node *new_node, *ptr;
	int num;
```

```
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        start2=add_node(start2);
                        new_node->next=new_node;
                        start=new_node;

                }
                else
                {
                        start2=add_node(start2);
                        ptr=start;
                        while(ptr->next!=start)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=start;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        s1.ad1=start;
        s1.ad2=start2;
        return s1;
}
struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;
}
```

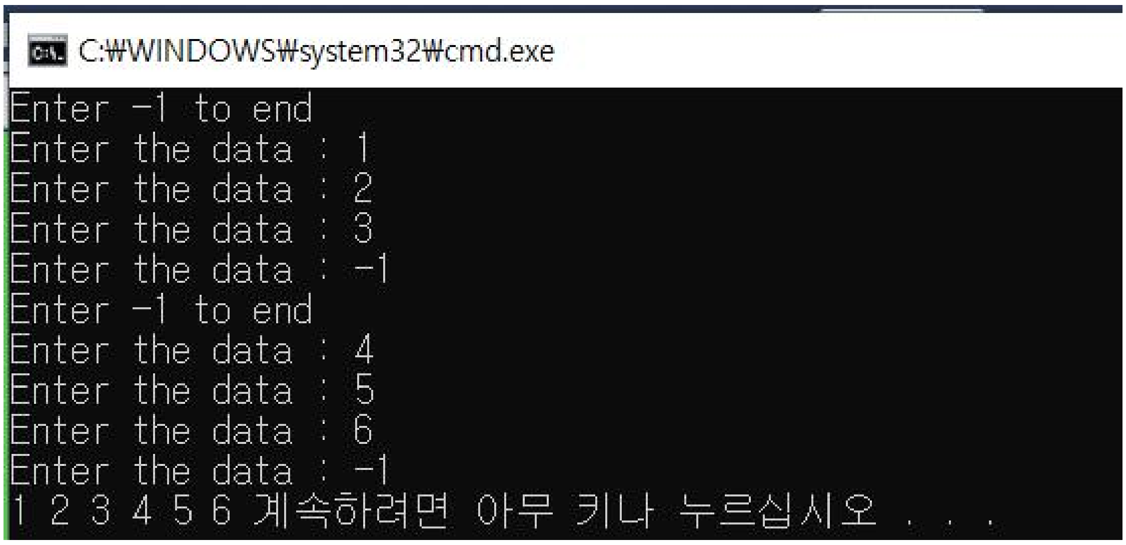
```
struct node *add_node(struct node *start2)
{
        struct node *new_node, *ptr;
        if(start2==NULL)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->next=new_node;
                new_node->data=0;
                start2=new_node;
        }
        else
        {       new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=0;
                ptr=start2;
                while(ptr->next!=start2)
                        ptr=ptr->next;
                ptr->next=new_node;
                new_node->next=start2;
        }
        return start2;
}

struct node *copy(struct node *start, struct node *start2)
{
        struct node *ptr1, *ptr2;
        ptr1=start;
        ptr2=start2;
        while(ptr1->next!=start)
        {
                ptr2->data=ptr1->data;
                ptr1=ptr1->next;
                ptr2=ptr2->next;
        }
        ptr2->data=ptr1->data;

        return start2;
}
```



```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
1 2 3 4 5계속하려면 아무 키나 누르십시오 . . .
```

7.

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

struct node{
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *merge(struct node *start,struct node *start2,struct node *start3);
int main()
{
        start=create(start);
        start2=create(start2);
        start3=merge(start,start2,start3);
        display(start3);
}

struct node *create(struct node *start)
{
        int num;
        struct node *new_node,*ptr;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {

                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                        ptr=ptr->next;
                        new_node->next=NULL;
                        ptr->next=new_node;

                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *merge(struct node *start,struct node *start2, struct node *start3)
{
        struct node *new_node, *ptr1, *ptr2,*ptr3;
        ptr1=start;
        ptr2=start2;

        while(ptr2!=NULL)
        {
                if(start3==NULL)
                {
                        new_node=(struct node *)malloc(sizeof(struct node));
                        new_node->data=ptr1->data;
                        ptr1=ptr1->next;
                        new_node->next=NULL;
                        start3=new_node;
                }
                else
                {

                        while(ptr2!=NULL)
                        {
                                if(ptr1!=NULL)
                                {       new_node=(struct   node   *)malloc(sizeof(struct
node));

                                        new_node->data=ptr1->data;
                                        ptr1=ptr1->next;
                                        break;
                                }

                                if(ptr1==NULL&&ptr2!=NULL)
                                {
                                        new_node=(struct   node   *)malloc(sizeof(struct
node));

                                        new_node->data=ptr2->data;
                                        ptr2=ptr2->next;
                                        break;
                                }
                        }
                        ptr3=start3;
                        while(ptr3->next!=NULL)
                        ptr3=ptr3->next;
                        new_node->next=NULL;
                        ptr3->next=new_node;



                }

        }
        return start3;
}
```

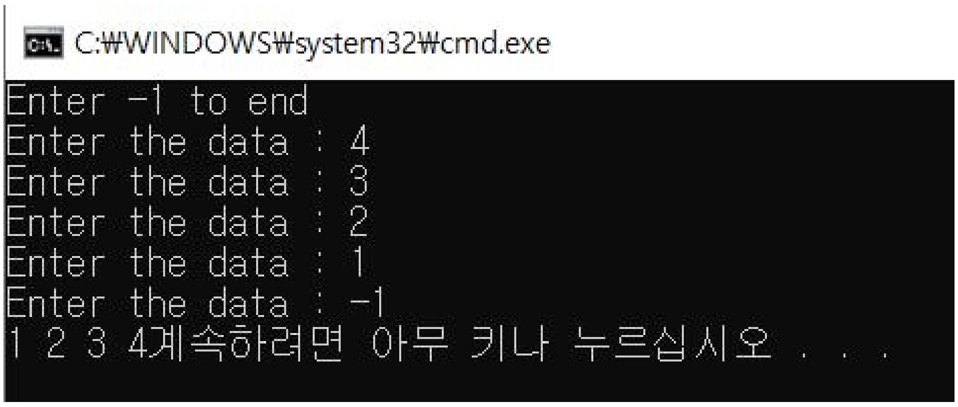

C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
Enter -1 to end
Enter the data : 4
Enter the data : 5
Enter the data : 6
Enter the data : -1
1 2 3 4 5 6 계속하려면 아무 키나 누르십시오 . . .
```

8.

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

struct node{
   int data;
   struct node *next;
   struct node *prev;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *sort(struct node *start);
int main()
{
         start=create(start);
         start=sort(start);
         start=display(start);
}
struct node *create(struct node *start)
{
         struct node *new_node, *ptr;
         int num;
         printf("Enter -1 to end\n");
         printf("Enter the data : ");
         scanf("%d",&num);
         while(num!=-1)
         {        new_node=(struct node*)malloc(sizeof(struct node));
                  new_node->data=num;
                  if(start==NULL)
```

```
                {
                        new_node->prev=NULL;
                        new_node->next=new_node;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=start)
                        ptr=ptr->next;
                        new_node->prev=ptr;
                        ptr->next=new_node;
                        new_node->next=start;
                        start->prev=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;
}

struct node *sort(struct node *start)
{
        struct node *ptr1, *ptr2;
        int temp;
        ptr1=start;
        while(ptr1->next!=start)
        {
                ptr2=ptr1->next;
                while(ptr2!=start)
                {
                        if(ptr1->data > ptr2->data)
                        {
                                temp=ptr1->data;
                                ptr1->data=ptr2->data;
                                ptr2->data=temp;
                        }
                        ptr2=ptr2->next;
                }
                ptr1=ptr1->next;
        }
        return start;
}
```



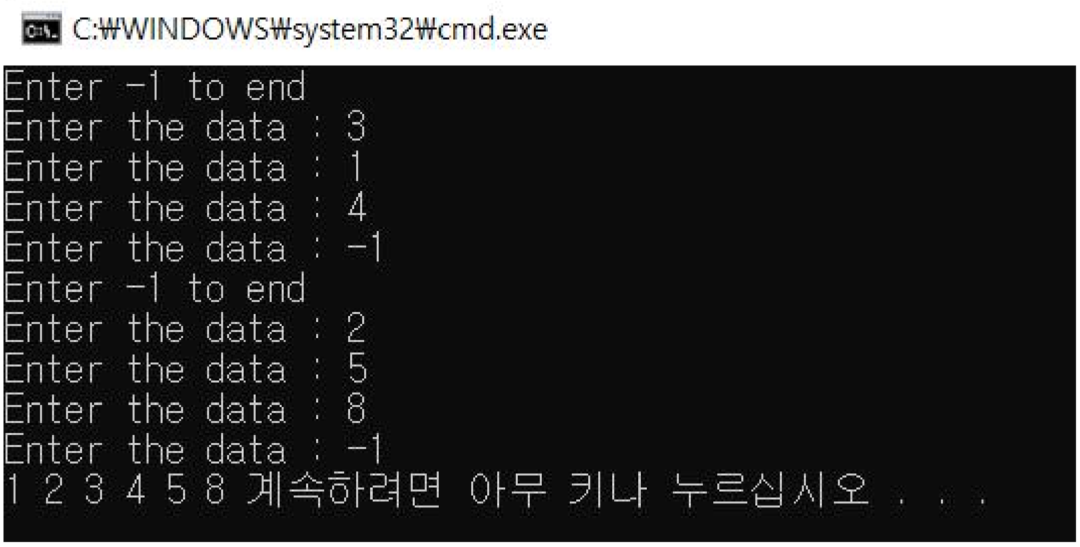
```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 4
Enter the data : 3
Enter the data : 2
Enter the data : 1
Enter the data : -1
1 2 3 4계속하려면 아무 키나 누르십시오 . . .
```

9.

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

struct node{
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *merge(struct node *start,struct node *start2,struct node *start3);
struct node *sort(struct node *start);
int main()
{
        start=create(start);
        start2=create(start2);
        start3=merge(start,start2,start3);
```

```
		start3=sort(start3);
		display(start3);
}

struct node *create(struct node *start)
{
		int num;
		struct node *new_node,*ptr;
		printf("Enter -1 to end\n");
		printf("Enter the data : ");
		scanf("%d",&num);
		while(num!=-1)
		{
				new_node=(struct node *)malloc(sizeof(struct node));
				new_node->data=num;
				if(start==NULL)
				{

						new_node->next=NULL;
						start=new_node;
				}
				else
				{
						ptr=start;
						while(ptr->next!=NULL)
						ptr=ptr->next;
						new_node->next=NULL;
						ptr->next=new_node;

				}
				printf("Enter the data : ");
				scanf("%d",&num);
		}
		return start;
}

struct node *display(struct node *start)
{
		struct node *ptr;
		ptr=start;
		while(ptr!=NULL)
		{
				printf("%d ",ptr->data);
```

```
                ptr=ptr->next;
        }
        return start;
}

struct node *merge(struct node *start,struct node *start2, struct node *start3)
{
        struct node *new_node, *ptr1, *ptr2,*ptr3;
        ptr1=start;
        ptr2=start2;

        while(ptr2!=NULL)
        {
                if(start3==NULL)
                {
                        new_node=(struct node *)malloc(sizeof(struct node));
                        new_node->data=ptr1->data;
                        ptr1=ptr1->next;
                        new_node->next=NULL;
                        start3=new_node;
                }
                else
                {

                        while(ptr2!=NULL)
                        {
                                if(ptr1!=NULL)
                                {       new_node=(struct    node    *)malloc(sizeof(struct
node));
                                        new_node->data=ptr1->data;
                                        ptr1=ptr1->next;
                                        break;
                                }

                                if(ptr1==NULL&&ptr2!=NULL)
                                {
                                        new_node=(struct    node    *)malloc(sizeof(struct
node));
                                        new_node->data=ptr2->data;
                                        ptr2=ptr2->next;
                                        break;
                                }
                        }
                        ptr3=start3;
                        while(ptr3->next!=NULL)
                        ptr3=ptr3->next;
                        new_node->next=NULL;
                        ptr3->next=new_node;



                }

        }
        return start3;
}

struct node *sort(struct node *start)
{
        struct node *ptr1, *ptr2;
        int temp;
        ptr1=start;
        while(ptr1->next!=NULL)
        {
                ptr2=ptr1->next;
                while(ptr2!=NULL)
                {
                        if(ptr1->data>ptr2->data)
                        {
                                temp=ptr1->data;
                                ptr1->data=ptr2->data;
                                ptr2->data=temp;
                        }
                        ptr2=ptr2->next;
                }
                ptr1=ptr1->next;
        }
        return start; }
```


C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 3
Enter the data : 1
Enter the data : 4
Enter the data : -1
Enter -1 to end
Enter the data : 2
Enter the data : 5
Enter the data : 8
Enter the data : -1
1 2 3 4 5 8 계속하려면 아무 키나 누르십시오 . . .
```

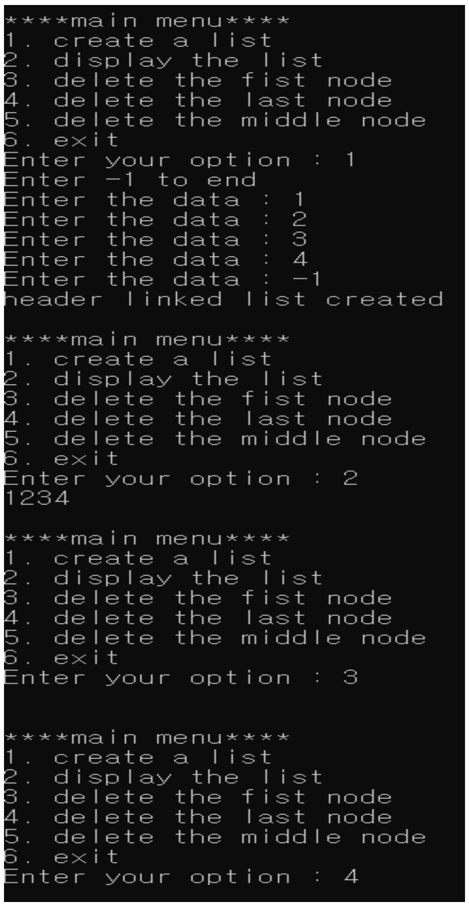
10.

```
#include <stdio.h>
#include <malloc.h>
struct node{
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_beg(struct node *start);
struct node *delete_end(struct node *start);
struct node *delete_node(struct node *start);
int main()
{
        int option;
        do
        {
        printf("\n\n");
        printf("****main menu****\n");
        printf("1. create a list\n");
        printf("2. display the list\n");
        printf("3. delete the fist node\n");
        printf("4. delete the last node\n");
        printf("5. delete the middle node\n");
        printf("6. exit\n");
        printf("Enter your option : ");
        scanf("%d",&option);
        switch(option)
```

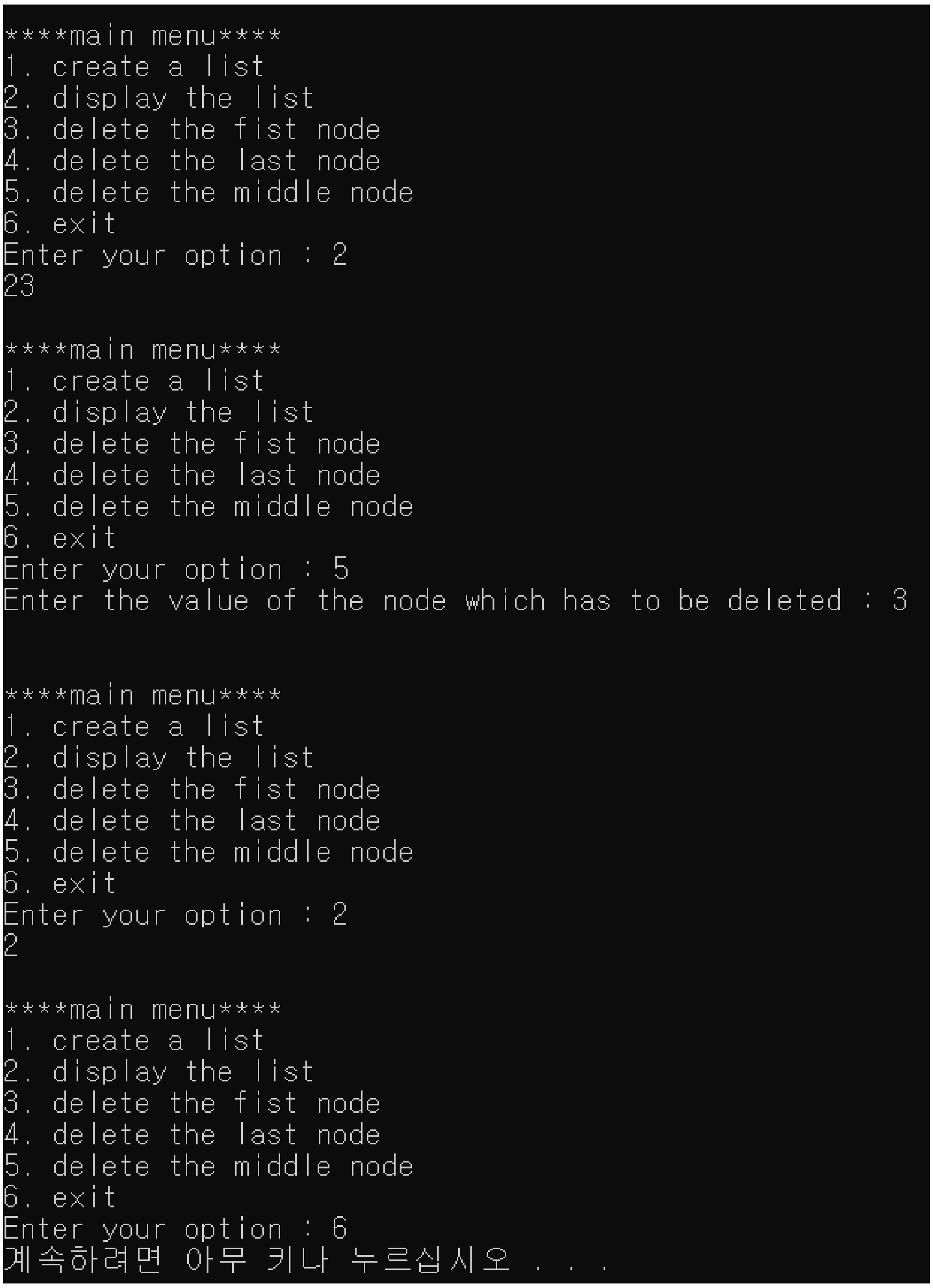
```
          {
                    case 1:start=create(start);
                    printf("header linked list created");
                    break;

                    case 2:start=display(start);
                    break;

                    case 3:start=delete_beg(start);
                    break;

                    case 4:start=delete_end(start);
                    case 5:start=delete_node(start);
          }

          }while(option !=6);

}

struct node *create(struct node *start)
{
          struct node *new_node, *ptr;
          int num;
          printf("Enter -1 to end\n");
          printf("Enter the data : ");
          scanf("%d",&num);
          while(num!=-1)
          {
                    new_node=(struct node*)malloc(sizeof(struct node));
                    new_node->data=num;
                    new_node->next=NULL;
                    if(start==NULL)
                    {
                              start=(struct node*)malloc(sizeof(struct node));
                              start->next=new_node;
                    }
                    else
                    {
                              ptr=start;
                              while(ptr->next!=NULL)
                                        ptr=ptr->next;
                                        ptr->next=new_node;
                    }
```

```
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}
struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start->next;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        free(ptr);
        return start;
}

struct node *delete_end(struct node *start)
{
        struct node *ptr, *preptr;
        ptr=start;
        while(ptr->next!=NULL)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=NULL;
        free(ptr);
        return start;
}

struct node *delete_node(struct node *start)
{
        struct node *ptr, *preptr;
```

```
          int val;
          printf("Enter the value of the node which has to be deleted : ");
          scanf("%d",&val);
          ptr=start;
          if(ptr->data==val)
          {
                    start=delete_beg(start);
                    return start;
          }
          else
          {
                    while(ptr->data!=val)
                    {
                              preptr=ptr;
                              ptr=ptr->next;
                    }
                    preptr->next=ptr->next;
                    free(ptr);
                    return start;
          }
}
```



```
****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
header linked list created

****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 2
1234

****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 3


****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 4
```

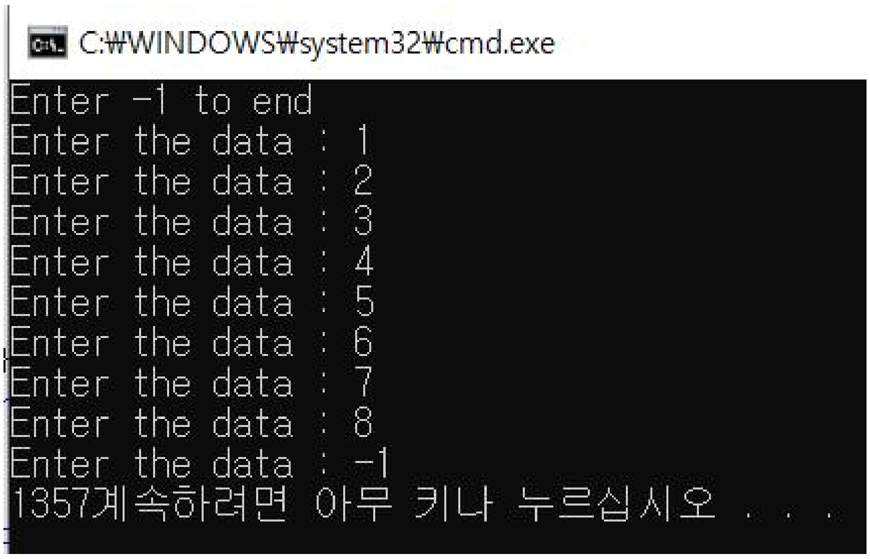


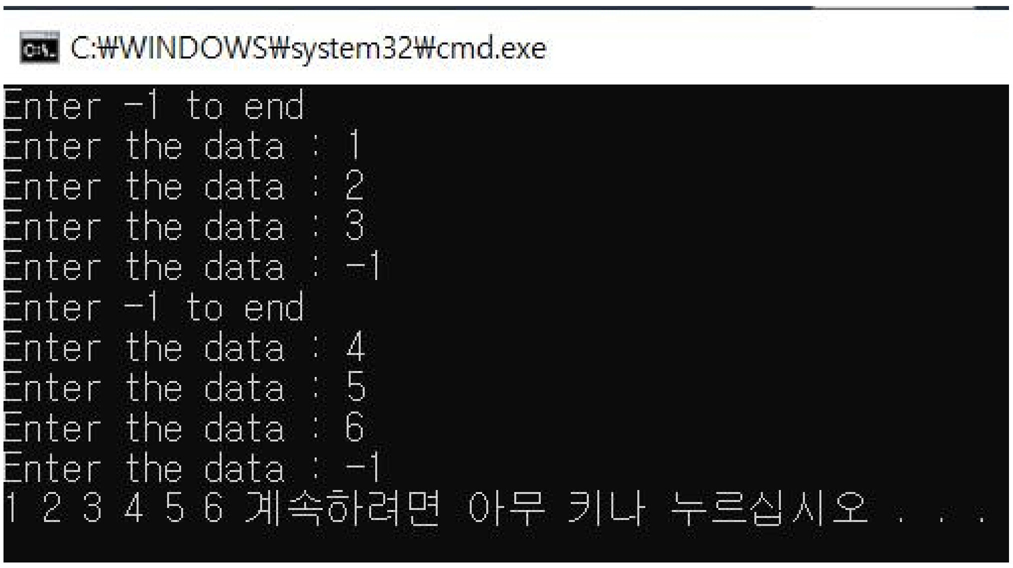
```
****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 2
23

****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 5
Enter the value of the node which has to be deleted : 3


****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 2
2

****main menu****
1. create a list
2. display the list
3. delete the fist node
4. delete the last node
5. delete the middle node
6. exit
Enter your option : 6
계속하려면 아무 키나 누르십시오 . . .
```

11.

```
#include<stdio.h>
#include <malloc.h>
struct node
```

```
{
	int data;
	struct node *next;
};
struct node *start=NULL;
struct node *display(struct node *start);
struct node *create(struct node *start);
int main()
{
	start=create(start);
	start=display(start);
}
struct node *create(struct node *start)
{
	struct node *new_node,*ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

struct node *display(struct node *start)
```

```
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
                ptr=ptr->next;
        }
        return start;
}
```

C:₩WINDOWS₩system32₩cmd.exe
```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : 6
Enter the data : 7
Enter the data : 8
Enter the data : -1
1357계속하려면 아무 키나 누르십시오 . . .
```
12.
```
#include <stdio.h>
#include <malloc.h>

struct node{
        int data;
        struct node *next;
        struct node *prev;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *merge(struct node *start,struct node *start2,struct node *start3);
int main()
{
        start=create(start);
        start2=create(start2);
        start3=merge(start,start2,start3);
        display(start3);
}

struct node *create(struct node *start)
{
        int num;
        struct node *new_node,*ptr;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->prev=NULL;
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                        ptr=ptr->next;
                        new_node->next=NULL;
                        new_node->prev=NULL;
                        ptr->next=new_node;

                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *merge(struct node *start,struct node *start2, struct node *start3)
{
        struct node *new_node, *ptr1, *ptr2,*ptr3;
        ptr1=start;
        ptr2=start2;

        while(ptr2!=NULL)
        {
                if(start3==NULL)
                {
                        new_node=(struct node *)malloc(sizeof(struct node));
                        new_node->data=ptr1->data;
                        ptr1=ptr1->next;
                        new_node->next=NULL;
                        start3=new_node;
                }
                else
                {

                        while(ptr2!=NULL)
                        {
                                if(ptr1!=NULL)
                                {       new_node=(struct    node    *)malloc(sizeof(struct
node));
                                        new_node->data=ptr1->data;
                                        ptr1=ptr1->next;
                                        break;
                                }

                                if(ptr1==NULL&&ptr2!=NULL)
                                {
                                        new_node=(struct    node    *)malloc(sizeof(struct
node));
                                        new_node->data=ptr2->data;
                                        ptr2=ptr2->next;
                                        break;
                                }
                        }
                        ptr3=start3;
                        while(ptr3->next!=NULL)
                        ptr3=ptr3->next;
                        new_node->next=NULL;
                        ptr3->next=new_node;



                }

        }
        return start3;
}
```

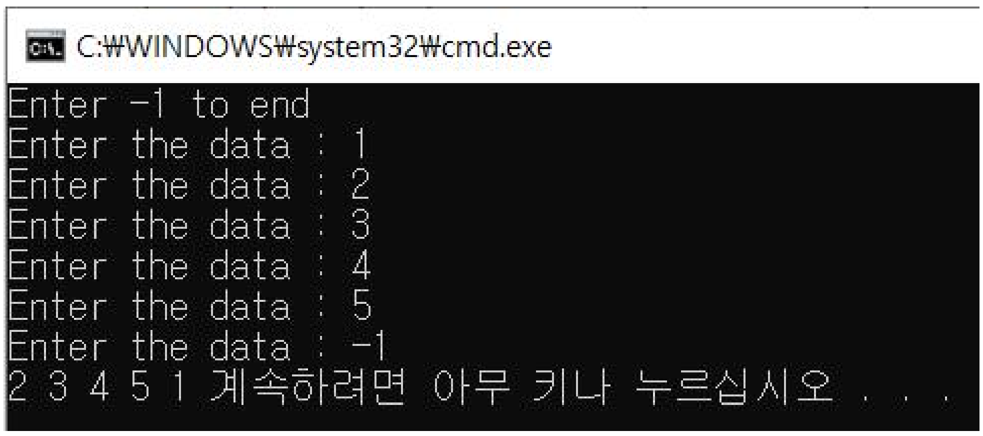


```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
Enter -1 to end
Enter the data : 4
Enter the data : 5
Enter the data : 6
Enter the data : -1
1 2 3 4 5 6 계속하려면 아무 키나 누르십시오 . . .
```

13.

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

struct node{
        int data;
        struct node *next;
        struct node *prev;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *put(struct node *start,int *num);
struct node *delete_beg(struct node *start, int *num);
int main()
{
	int a;
	start=create(start);
	start=delete_beg(start,&a);
	start=put(start,&a);
	display(start);
}

struct node *create(struct node *start)
{
	int num;
	struct node *new_node,*ptr;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node *)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->prev=NULL;
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
			ptr=ptr->next;
			new_node->next=NULL;
			new_node->prev=NULL;
			ptr->next=new_node;

		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}
```

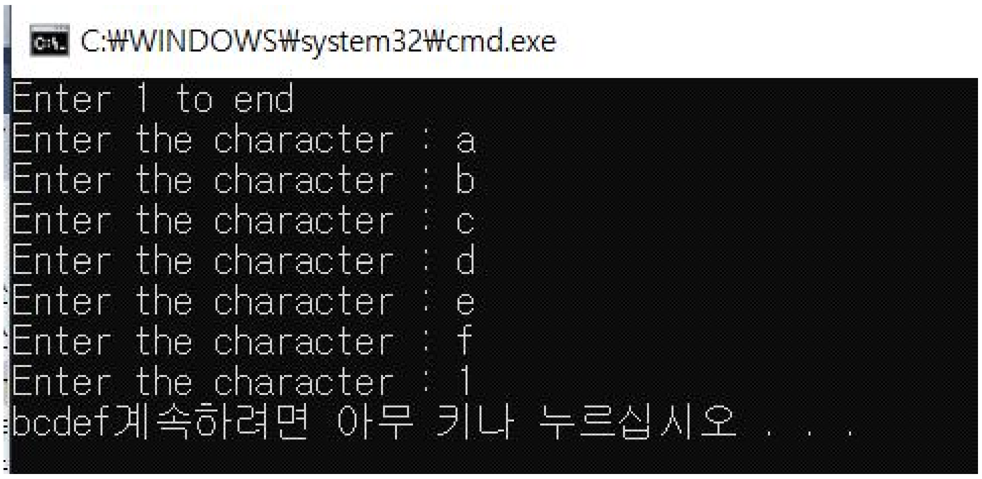
```
struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *delete_beg(struct node *start, int *num)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        start->prev=NULL;
        *num=ptr->data;
        free(ptr);
        return start;
}

struct node *put(struct node *start,int *num)
{
        struct node *new_node, *ptr;
        new_node = (struct node *)malloc(sizeof(struct node));
        new_node->data=*num;
        ptr=start;
        while(ptr->next!=NULL)
                ptr=ptr->next;
        ptr->next=new_node;
        new_node->prev=ptr;
        new_node->next=NULL;
        return start;
}
```


C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
2 3 4 5 1 계속하려면 아무 키나 누르십시오 . . .
```

14-(a).

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


struct node {

        char ch;
        struct node *next;
};
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_beg(struct node *start);
struct node *start=NULL;

int main()
{
        start=create(start);
        start=delete_beg(start);
        start=display(start);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        char ch;
        printf("Enter 1 to end\n");
        printf("Enter the character : ");
        scanf("%c",&ch);
        while(ch!='1')
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->ch=ch;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=NULL;
                }
                fflush(stdin);
                printf("Enter the character : ");
                scanf("%c",&ch);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%c",ptr->ch);
                ptr=ptr->next;
        }
        return start;

}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        free(ptr);
        return start;
}
```

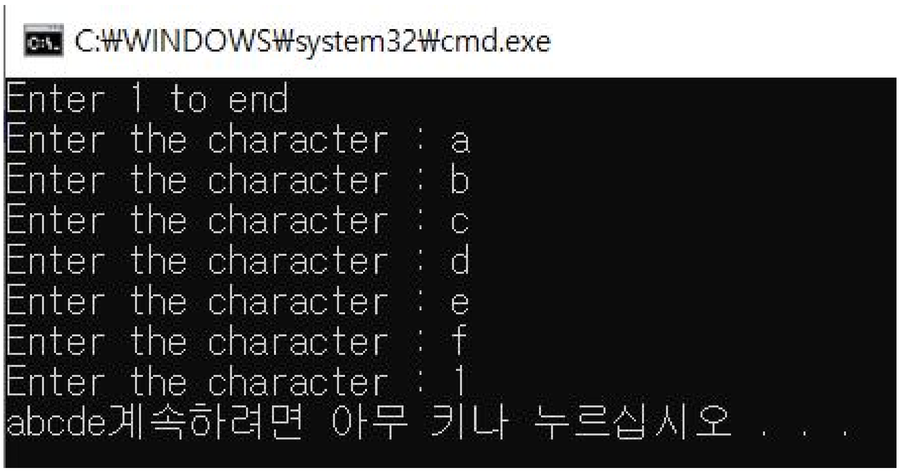


```
Enter 1 to end
Enter the character : a
Enter the character : b
Enter the character : c
Enter the character : d
Enter the character : e
Enter the character : f
Enter the character : 1
bcdef계속하려면 아무 키나 누르십시오 . . .
```

14-(b).

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


struct node {

	char ch;
	struct node *next;
};
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_end(struct node *start);
struct node *start=NULL;

int main()
{
	start=create(start);
	start=delete_end(start);
	start=display(start);
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	char ch;
	printf("Enter 1 to end\n");
	printf("Enter the character : ");
	scanf("%c",&ch);
	while(ch!='1')
	{
```

```
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->ch=ch;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=NULL;
                }
                fflush(stdin);
                printf("Enter the character : ");
                scanf("%c",&ch);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%c",ptr->ch);
                ptr=ptr->next;
        }
        return start;

}

struct node *delete_end(struct node *start)
{
        struct node *ptr, *preptr;
        ptr=start;
        while(ptr->next!=NULL)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=NULL;
        free(ptr);
        return start;
}
```



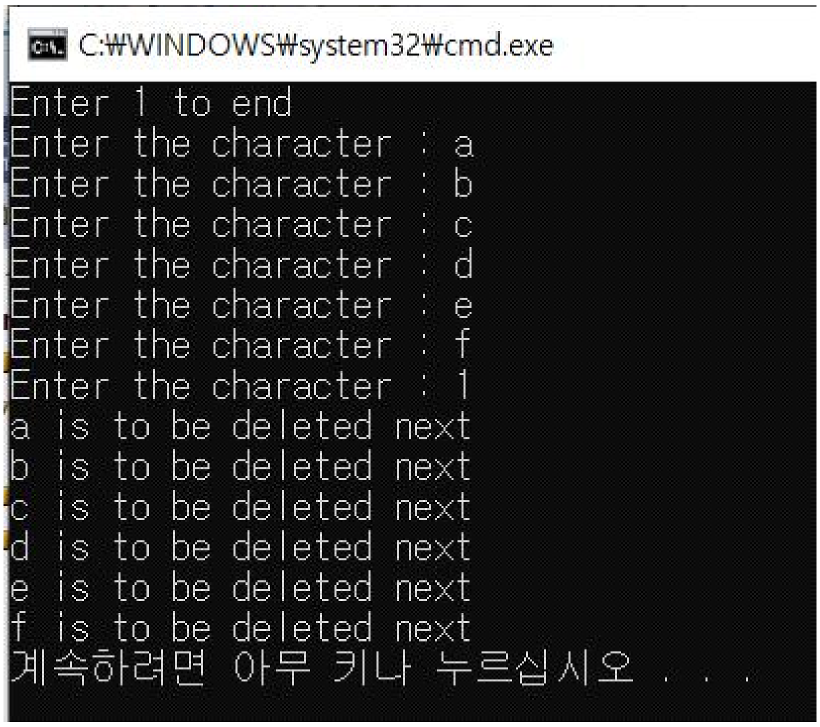
```
C:\WINDOWS\system32\cmd.exe
Enter 1 to end
Enter the character : a
Enter the character : b
Enter the character : c
Enter the character : d
Enter the character : e
Enter the character : f
Enter the character : 1
abcde계속하려면 아무 키나 누르십시오 . . .
```

14-(c).

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


struct node {

        char ch;
        struct node *next;
};
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_all(struct node *start);
struct node *delete_beg(struct node *start);
struct node *start=NULL;

int main()
{
        start=create(start);
        start=delete_all(start);
        start=display(start);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
```

```
	char ch;
	printf("Enter 1 to end\n");
	printf("Enter the character : ");
	scanf("%c",&ch);
	while(ch!='1')
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->ch=ch;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
		}
		fflush(stdin);
		printf("Enter the character : ");
		scanf("%c",&ch);
	}
	return start;
}

struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%c",ptr->ch);
		ptr=ptr->next;
	}
	return start;

}

struct node *delete_beg(struct node *start)
{
```

```
          struct node *ptr;
          ptr=start;
          start=start->next;
          free(ptr);
          return start;
}

struct node *delete_all(struct node *start)
{
          struct node *ptr;
          if(start!=NULL){
                    ptr=start;
                    while(ptr!=NULL)
                    {
                              printf("%c is to be deleted next\n",ptr->ch);
                              start=delete_beg(ptr);
                              ptr=start;
                    }

          }
}
```



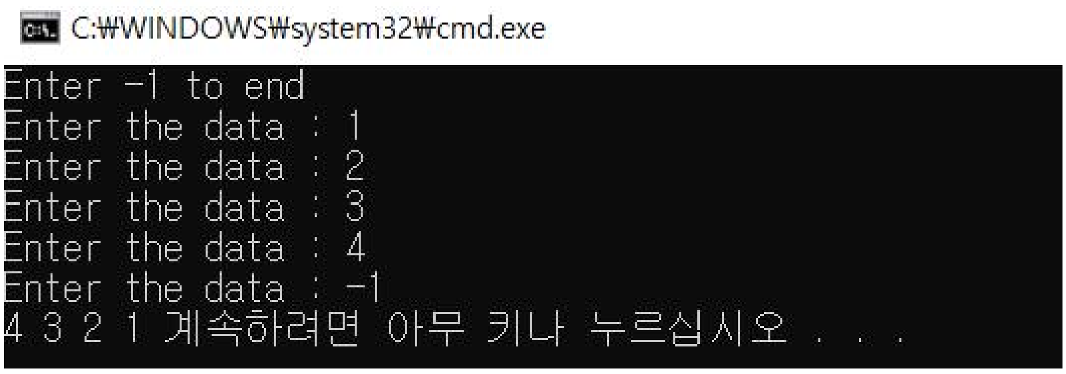
```
C:₩WINDOWS₩system32₩cmd.exe
Enter 1 to end
Enter the character : a
Enter the character : b
Enter the character : c
Enter the character : d
Enter the character : e
Enter the character : f
Enter the character : 1
a is to be deleted next
b is to be deleted next
c is to be deleted next
d is to be deleted next
e is to be deleted next
f is to be deleted next
계속하려면 아무 키나 누르십시오 . . .
```

15.

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

```
struct node {
		int data;
		struct node *next;
};
struct node *start=NULL;
struct node* delete_beg(struct node *start, int *num);
struct node *create(struct node *start,int *p1);
struct node *display(struct node *start);
int main()
{
		int num[10];
		int i,put;
		int count,*p1,*p2;
		int print[10];
		p1=&count;//노드갯수 몇개인지 셈
		p2=&put;
		start=create(start,p1);
		for(i=0;i<count-1;i++)
			{
				start=delete_beg(start,p2);
				print[i]=put;
			}
		for(i=count-2;i>=0;i--)
		{
			printf("%d ",print[i]);
		}
}

struct node *create(struct node *start,int *p1)
{
		int num,count=1;
		struct node *new_node,*ptr;

		printf("Enter -1 to end\n");
		printf("Enter the data : ");
		scanf("%d",&num);
		while(num!=-1)
		{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->data=num;
			if(start==NULL)
			{
```

```
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
                count++;
        }
        *p1=count;
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
}

struct node* delete_beg(struct node *start, int *num)
{

        struct node *ptr;
        ptr=start;
        start=start->next;
        *num=ptr->data;
        free(ptr);
        return start;
}
```

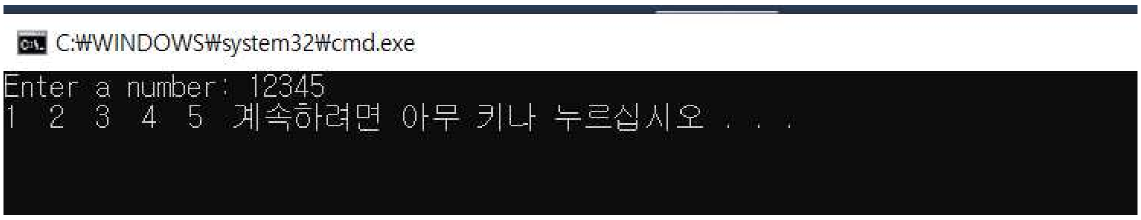

C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
4 3 2 1 계속하려면 아무 키나 누르십시오 . . .
```

16.

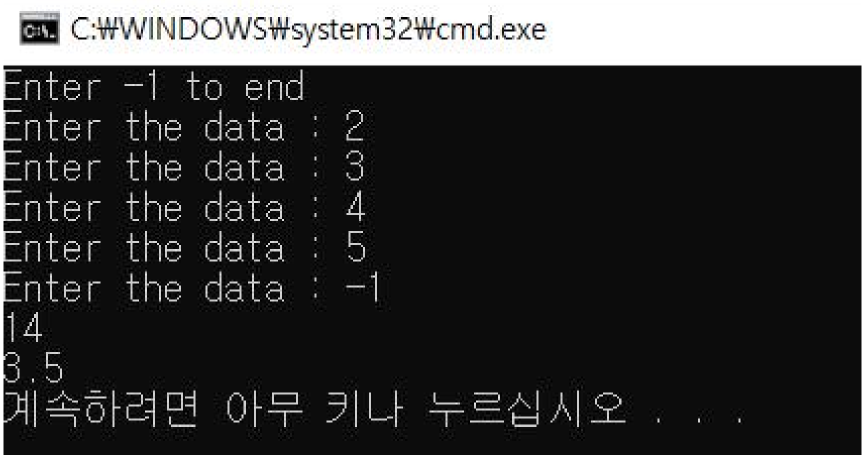
```
#include <stdio.h>
#include <malloc.h>
struct node
{
        struct node *next;
        int data;
};
struct node *start=NULL;
struct node *create(struct node *start,int a);
struct node *display(struct node *start);
int main()
{
         int num,temp,factor=1;
         int a[10],i=0,j;
  printf("Enter a number: ");
  scanf("%d",&num);
  temp=num;

  while(temp){
      temp=temp/10;
      factor = factor*10;
  }

  while(factor>1){
      factor = factor/10;
       a[i]=num/factor;
           i++;
      num = num % factor;
  }
  for(j=0;j<i;j++)
 start=create(start,a[j]);
  display(start);
```

```c
}
struct node *create(struct node *start,int a)
{

        struct node *new_node, *ptr;

        if(start==NULL)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=a;
                new_node->next=NULL;
                start=new_node;
        }
        else
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=a;
                ptr=start;
                while(ptr->next!=NULL)
                        ptr=ptr->next;
                ptr->next=new_node;
                new_node->next=NULL;
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d   ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
```


```
C:₩WINDOWS₩system32₩cmd.exe
Enter a number: 12345
1  2  3  4  5  계속하려면 아무 키나 누르십시오 . . .
```

17.
```c
#include <stdio.h>
#include <malloc.h>
struct node
{
        struct node *next;
        int data;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *add(struct node *start,struct node *start2,struct node *start3);
int main()
{
        int sum,count;
        start=create(start,&sum,&count);

        printf("sum=%d\n",sum);
        printf("mean=%.1lf\n",sum/(double)count);

}
struct node *create(struct node *start, int *sum, int *count)
{

        struct node *new_node, *ptr;
        int num;
        *sum=0;
        *count=0;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                if(start==NULL)
                {
```

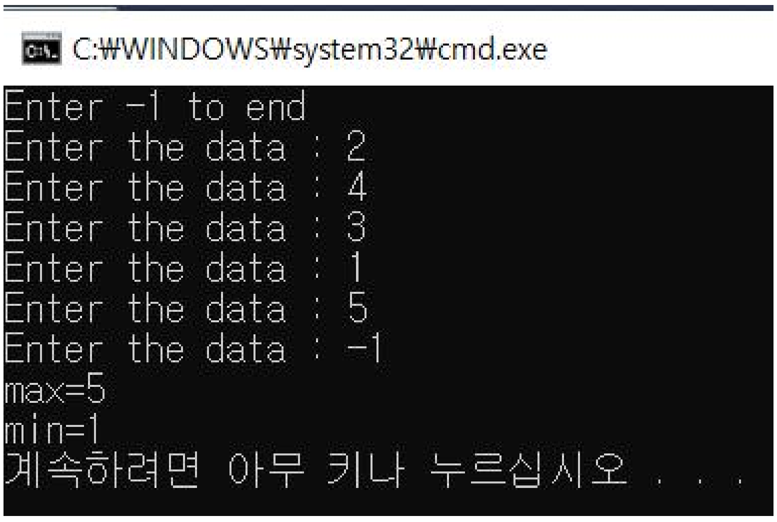
```
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                *sum=*sum+new_node->data;
                new_node->next=NULL;
                start=new_node;

                }
                else
                {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                *sum=*sum+new_node->data;
                ptr=start;
                while(ptr->next!=NULL)
                        ptr=ptr->next;
                ptr->next=new_node;
                new_node->next=NULL;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
                *count=*count+1;
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d   ",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
```


C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
14
3.5
계속하려면 아무 키나 누르십시오 . . .
```

18.

```
#include <stdio.h>
#include <malloc.h>
struct node
{
	struct node *next;
	int data;
};
struct value
{
	int max;
	int min;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct value sort(struct node *start);
int main()
{
	struct value v1;
	start=create(start);
	v1=sort(start);
	printf("max=%d\n",v1.max);
	printf("min=%d\n",v1.min);

}
struct node *create(struct node *start)
{

	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		if(start==NULL)
		{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		new_node->next=NULL;
		start=new_node;
		}
		else
		{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		ptr=start;
		while(ptr->next!=NULL)
			ptr=ptr->next;
		ptr->next=new_node;
		new_node->next=NULL;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d  ",ptr->data);
		ptr=ptr->next;
	}
	return start;
}

struct value sort(struct node *start)
{
	struct node *ptr1;
	struct value v1;
	int max,min;

	ptr1=start->next;
	max=start->data;
	min=start->data;
	while(ptr1!=NULL)
	{
		if(max<ptr1->data)
		{
			max=ptr1->data;
		}
		ptr1=ptr1->next;
	}
	ptr1=start->next;
	while(ptr1!=NULL)
	{
		if(min>ptr1->data)
		{
			min=ptr1->data;
		}
		ptr1=ptr1->next;
	}
	v1.max=max;
	v1.min=min;
	return v1;
}
```

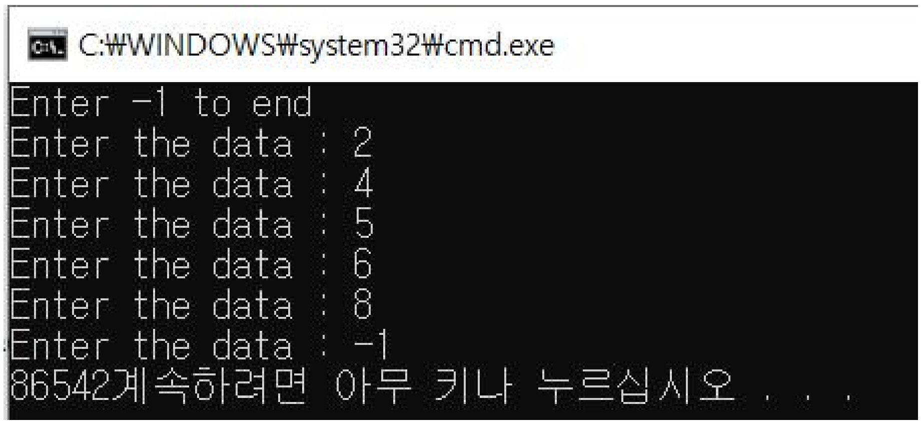


```
Enter -1 to end
Enter the data : 2
Enter the data : 4
Enter the data : 3
Enter the data : 1
Enter the data : 5
Enter the data : -1
max=5
min=1
계속하려면 아무 키나 누르십시오 . . .
```

19.

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

```
struct node{
        int data;
        struct node *prev;
        struct node *next;
        int pos;
};
struct node *start=NULL;
struct node *create(struct node *start, int *count);
struct node *display(struct node *start);
struct node *inter(struct node *start,int count);

int main()
{
        int count;
        start=create(start,&count);
        start=inter(start,count);
        display(start);
}
struct node *create(struct node *start, int *count)
{
        struct node *new_node, *ptr;
        int num;
        int pos=1;
        *count=0;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                if(start==NULL)
                {
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->pos=pos;
                        pos++;
                        new_node->data=num;
                        new_node->prev=NULL;
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->data=num;
                        new_node->pos=pos;
                        pos++;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->prev=ptr;
                        new_node->next=NULL;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
                *count=*count+1;
    }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *inter(struct node *start,int count)
{
        int i=1,j=count;
        int temp;
        struct node *ptr;
        struct node *ptr2;
        ptr=start;
        ptr2=start;
        while(ptr2->next!=NULL)
                ptr2=ptr2->next;

        while(ptr!=NULL)
        {
                if(i<j)
                {
                        j=count-i+1;
                if(ptr->pos==i && ptr2->pos==j)
                {
                        temp=ptr->data;
                        ptr->data=ptr2->data;
                        ptr2->data=temp;
                }
                ptr=ptr->next;
                ptr2=ptr2->prev;
                        i++;
                }
                else
                        break;
        }

        return start;
}
```



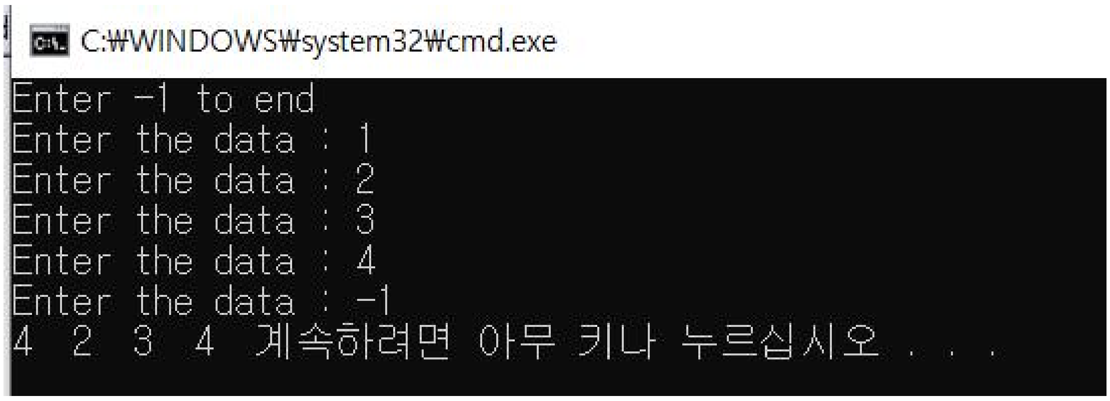
```
C:\WINDOWS\system32\cmd.exe
Enter -1 to end
Enter the data : 2
Enter the data : 4
Enter the data : 5
Enter the data : 6
Enter the data : 8
Enter the data : -1
86542계속하려면 아무 키나 누르십시오 . . .
```

20.

```
#include <stdio.h>
#include <malloc.h>
struct node
{
        struct node *next;
        int data;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *modify(struct node *start, int plus);
int main()
```

```
{
        int plus;
        start=create(start,&plus);
        start=modify(start,plus);
        display(start);
}
struct node *create(struct node *start, int *plus)
{

        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                if(start==NULL)
                {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                new_node->next=NULL;
                start=new_node;

                }
                else
                {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                *plus=new_node->data;
                ptr=start;
                while(ptr->next!=NULL)
                        ptr=ptr->next;
                ptr->next=new_node;
                new_node->next=NULL;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
```

```
          struct node *ptr;
          ptr=start;
          while(ptr!=NULL)
          {
                    printf("%d  ",ptr->data);
                    ptr=ptr->next;
          }
          return start;
}

struct node *modify(struct node *start, int plus)
{
          struct node *ptr;
          ptr=start;
          ptr->data=plus;
          return start;
}
```



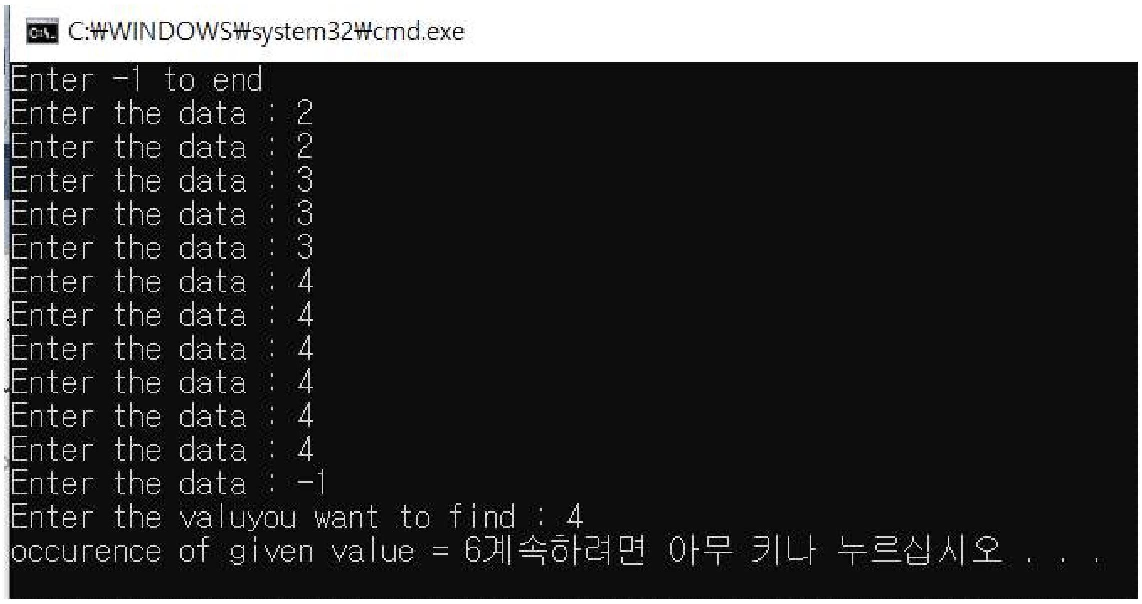
C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
4  2  3  4  계속하려면 아무 키나 누르십시오 . . .
```

21.

```
#include <stdio.h>
#include <malloc.h>
struct node {
          int data;
          struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *);
void check(struct node *start);
int main ()
{

          start=create(start);
          check(start);
```

```
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			new_node->next=NULL;
			ptr->next=new_node;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

void check(struct node *start)
{
	struct node *ptr;
	int count[100]={0};
	int i,num;
	ptr=start;
	while(ptr!=NULL)
	{
		for(i=0;i<100;i++)
		{
			if(ptr->data==i)
```

```
                        count[i]++;
                }
                ptr=ptr->next;
        }
        printf("Enter the valuyou want to find : ");
        scanf("%d",&num);
        printf("occurence of given value = %d",count[num]);


}
```


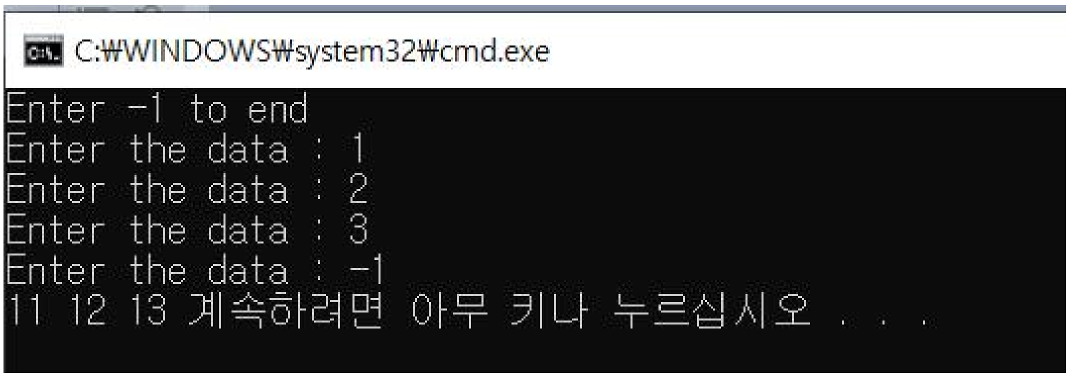
```
C:\WINDOWS\system32\cmd.exe
Enter -1 to end
Enter the data : 2
Enter the data : 2
Enter the data : 3
Enter the data : 3
Enter the data : 3
Enter the data : 4
Enter the data : 4
Enter the data : 4
Enter the data : 4
Enter the data : 4
Enter the data : 4
Enter the data : -1
Enter the valuyou want to find : 4
occurence of given value = 6계속하려면 아무 키나 누르십시오 . . .
```

22.

```
#include <stdio.h>
#include <malloc.h>
struct node {
        int data;
        struct node *next;
        struct node *prev;
};
struct node *start=NULL;
struct node *create(struct node *);
struct node *plus(struct node *);
struct node *display(struct node *);
int main ()
{

        start=create(start);
        start=plus(start);
        start=display(start);
```

```c
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{
			new_node->next=NULL;
			new_node->prev=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			new_node->next=NULL;
			new_node->prev=ptr;
			ptr->next=new_node;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}
struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d ",ptr->data);
		ptr=ptr->next;
	}
```

```
        return start;
}
struct node *plus(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                ptr->data=ptr->data+10;
                ptr=ptr->next;
        }
        return start;
}
```



C:₩WINDOWS₩system32₩cmd.exe

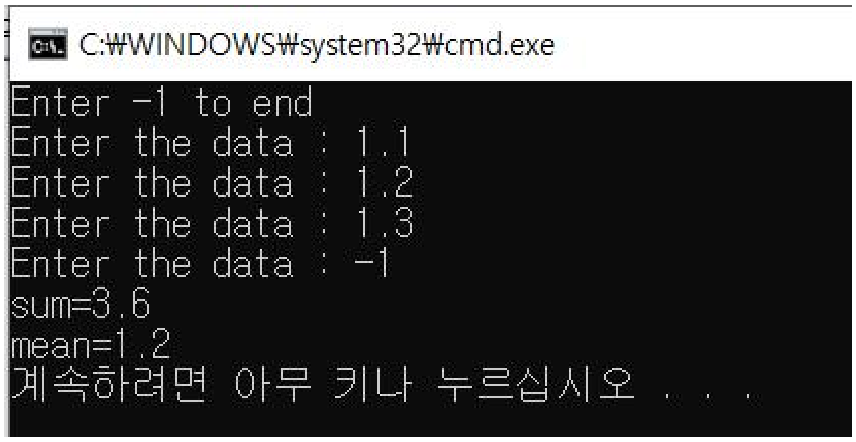
```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
11 12 13 계속하려면 아무 키나 누르십시오 . . .
```

23.

```
#include <stdio.h>
#include <malloc.h>
struct node
{
        struct node *next;
        double data;
};
struct node *start=NULL;
struct node *start2=NULL;
struct node *start3=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *add(struct node *start,struct node *start2,struct node *start3);
int main()
{
        double sum;
        int count;
        start=create(start,&sum,&count);

        printf("sum=%.1lf\n",sum);
```

```
		printf("mean=%.1lf\n",sum/count);

}
struct node *create(struct node *start, double *sum, int *count)
{

		struct node *new_node, *ptr;
		double num;
		*sum=0.0;
		*count=0;
		printf("Enter -1 to end\n");
		printf("Enter the data : ");
		scanf("%lf",&num);
		while(num!=-1)
		{
			if(start==NULL)
			{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->data=num;
			*sum=*sum+new_node->data;
			new_node->next=NULL;
			start=new_node;

			}
			else
			{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->data=num;
			*sum=*sum+new_node->data;
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
			}
			printf("Enter the data : ");
			scanf("%lf",&num);
			*count=*count+1;
		}
		return start;
}

struct node *display(struct node *start)
```

```
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d  ",ptr->data);
		ptr=ptr->next;
	}
	return start;
}
```



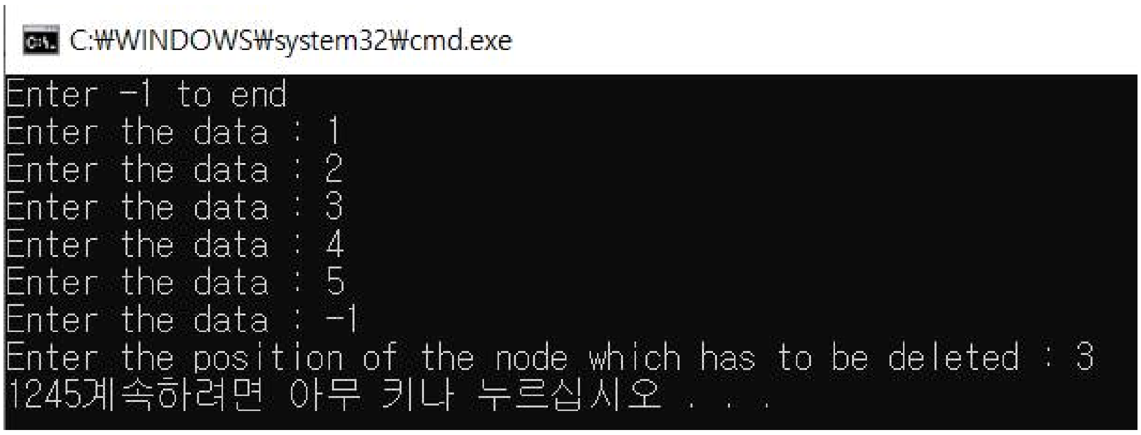
```
C:\WINDOWS\system32\cmd.exe
Enter -1 to end
Enter the data : 1.1
Enter the data : 1.2
Enter the data : 1.3
Enter the data : -1
sum=3.6
mean=1.2
계속하려면 아무 키나 누르십시오 . . .
```

24.

```
#include<stdio.h>
#include <malloc.h>
struct node
{
	int data;
	struct node *next;
	int pos;
};
struct node *start=NULL;
struct node *display(struct node *start);
struct node *create(struct node *start);
struct node *delete_node(struct node *start);
struct node *delete_beg(struct node *start);
int main()
{
	start=create(start);
	start=delete_node(start);
	start=display(start);

}
struct node *create(struct node *start)
```

```
{
	struct node *new_node,*ptr;
	int num;
	int pos=1;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		new_node->pos=pos;
		pos++;
		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
		}
		printf("Enter the data : ");
		scanf("%d",&num);
	}
	return start;
}

struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start;
	while(ptr!=NULL)
	{
		printf("%d",ptr->data);
		ptr=ptr->next;
	}
	return start;
}
```

```
struct node *delete_node(struct node *start)
{
	struct node *ptr, *preptr;
	int val;
	printf("Enter the position of the node which has to be deleted : ");
	scanf("%d",&val);
	ptr=start;
	if(ptr->pos==val)
	{
		start=delete_beg(start);
		return start;
	}
	else
	{
		while(ptr->pos!=val)
		{
			preptr=ptr;
			ptr=ptr->next;
		}
		preptr->next=ptr->next;
		free(ptr);
		return start;
	}
}

struct node *delete_beg(struct node *start)
{
	struct node *ptr;
	ptr=start;
	start=start->next;
	free(ptr);
	return start;
}
```



```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
Enter the position of the node which has to be deleted : 3
1245계속하려면 아무 키나 누르십시오 . . .
```

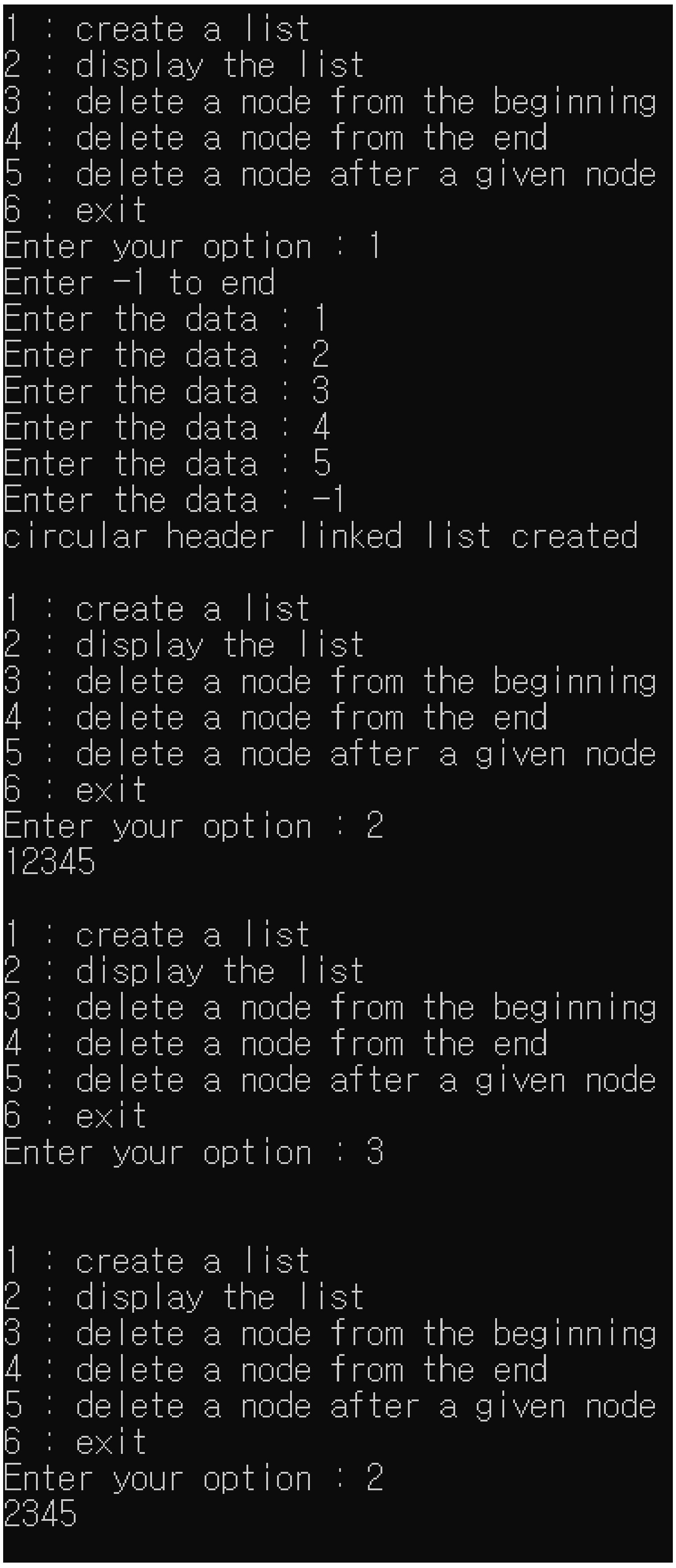
25.

```
#include <stdio.h>
#include <malloc.h>
struct node
{
	int data;
	struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_beg(struct node *start);
struct node *delete_end(struct node *start);
struct node *delete_after(struct node *start);

int main()
{
	int option;
	do{
	printf("\n\n");
	printf("1 : create a list\n");
	printf("2 : display the list\n");
	printf("3 : delete a node from the beginning\n");
	printf("4 : delete a node from the end\n");
	printf("5 : delete a node after a given node\n");
	printf("6 : exit\n");
	printf("Enter your option : ");
	scanf("%d",&option);
		switch(option)
		{
			case 1: start=create(start);
			printf("circular header linked list created");
```

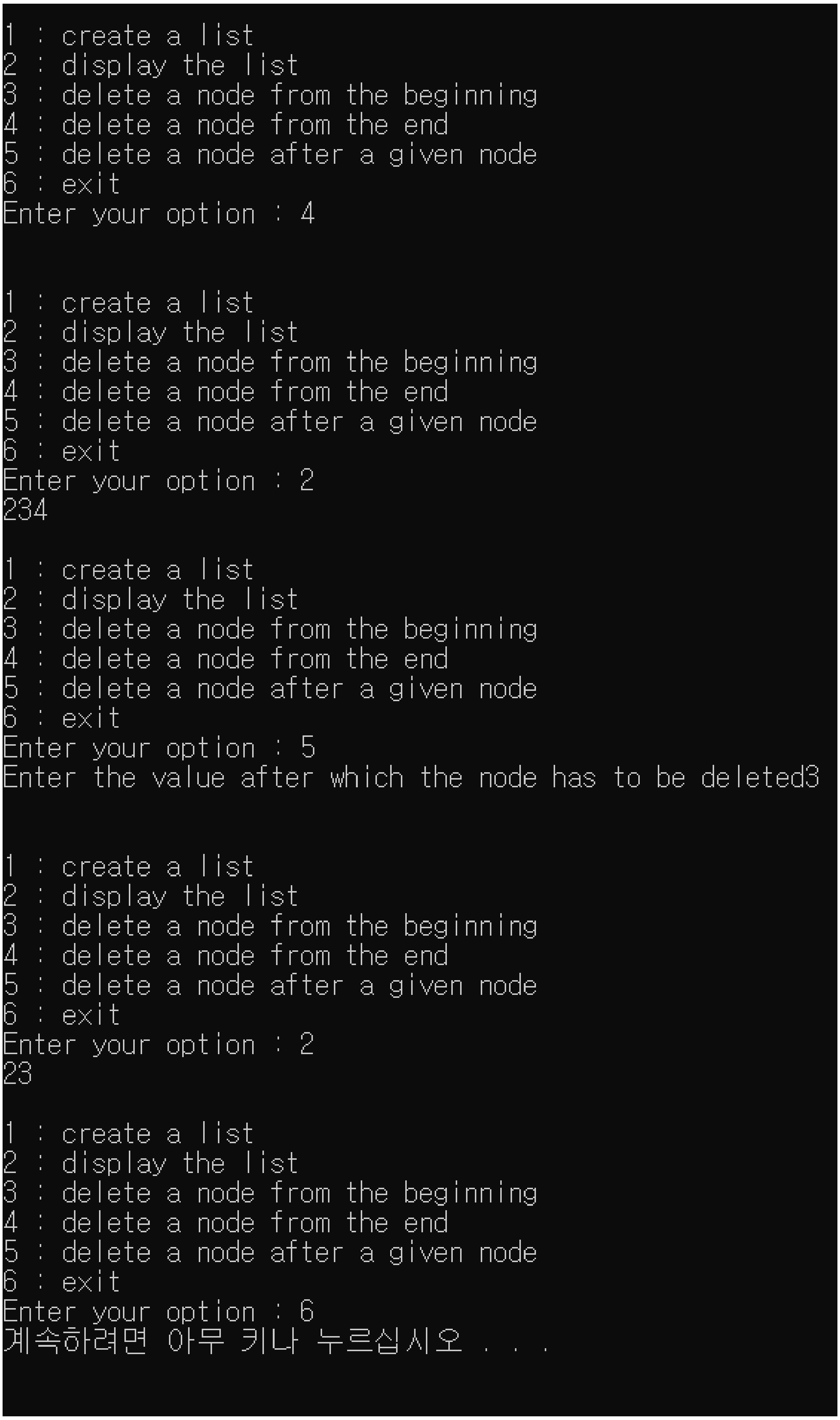
```
				break;
				case 2: start=display(start);
				break;
				case 3: start=delete_beg(start);
				break;
				case 4: start=delete_end(start);
				break;
				case 5: start=delete_after(start);
				break;
		}

		}while(option!=6);
}
struct node *create(struct node *start)
{
		struct node *new_node, *ptr;
		int num;
		printf("Enter -1 to end\n");
		printf("Enter the data : ");
		scanf("%d",&num);
		while(num!=-1)
		{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->data=num;

			if(start==NULL)
			{
				start=(struct node*)malloc(sizeof(struct node));
				start->next=new_node;
				new_node->next=start;

			}
			else
			{
				ptr=start;
				while(ptr->next!=start)
					ptr=ptr->next;
				ptr->next=new_node;
				new_node->next=start;
			}
			printf("Enter the data : ");
			scanf("%d",&num);
		}
```

```
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start->next;
        while(ptr->next!=start)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr, *temp;
        ptr=start->next;
        while(ptr->next!=start)
                ptr=ptr->next;
        temp=start->next;
        start->next=start->next->next;
        free(temp);
        return start;
}

struct node *delete_end(struct node *start)
{
        struct node *ptr, *preptr;
        ptr=start;
        while(ptr->next!=start)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=ptr->next;
        free(ptr);
        return start;
}

struct node *delete_after(struct node *start)
```

```
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the value after which the node has to be deleted");
        scanf("%d",&val);
        ptr=start;
        preptr=ptr;
        while(preptr->data!=val)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=ptr->next;
        if(ptr==start)
                start=preptr->next;
        free(ptr);
        return start;
}
```



```
1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
circular header linked list created

1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 2
12345

1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 3


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 2
2345
```

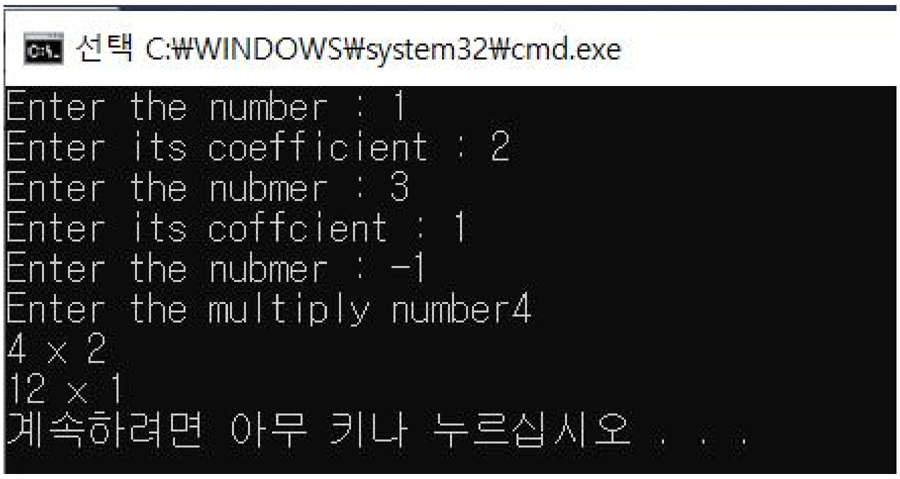


```
1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 4


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 2
234

1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 5
Enter the value after which the node has to be deleted3


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 2
23

1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 6
계속하려면 아무 키나 누르십시오 . . .
```

26.
```
#include <stdio.h>
#include <malloc.h>

struct node
{
	int num;
	int coeff;
	struct node *next;
};

struct node *start1=NULL;


struct node *create_poly(struct node *start);
struct node *display(struct node *start);
struct node *multi(struct node *start);
int main()
{
	start1=create_poly(start1);
	start1=multi(start1);
	start1=display(start1);
}

struct node *create_poly(struct node *start)
{
	struct node *new_node, *ptr;
	int n,c;
	printf("Enter the number : ");
	scanf("%d",&n);
	printf("Enter its coefficient : ");
	scanf("%d",&c);
	while(n!=-1)
	{
		if(start==NULL)
		{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->num=n;
			new_node->coeff=c;
			new_node->next=NULL;
			start=new_node;
		}
		else
```

```
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        new_node=(struct node*)malloc(sizeof(struct node));
                        new_node->num=n;
                        new_node->coeff=c;
                        new_node->next=NULL;
                        ptr->next=new_node;
                }
                printf("Enter the nubmer : ");
                scanf("%d",&n);
                if(n==-1)
                        break;
                printf("Enter its coffcient : ");
                scanf("%d",&c);
        }

        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d x %d\n",ptr->num,ptr->coeff);
                ptr=ptr->next;
        }
        return start;

}

struct node *multi(struct node *start)
{
        struct node *ptr;
        int num;
        printf("Enter the multiply number");
        scanf("%d",&num);
        ptr=start;
        while(ptr!=NULL)
        {
```

```
		ptr->num=ptr->num*num;
		ptr=ptr->next;
	}
	return start;
}
```

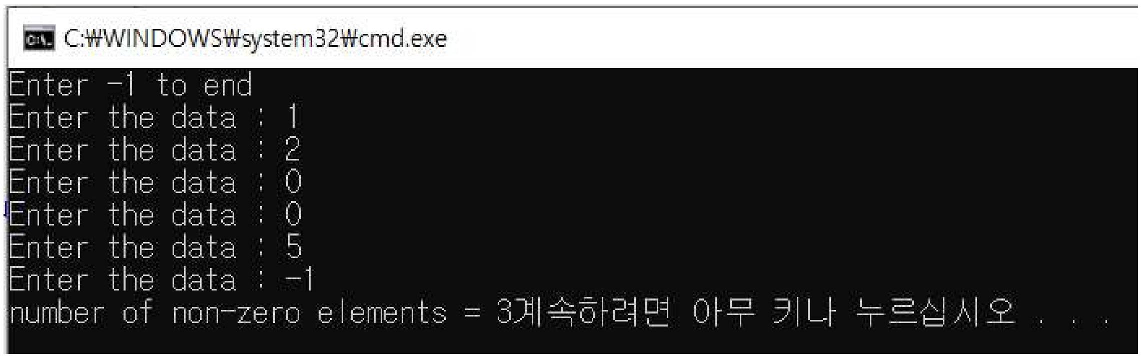


```
선택 C:₩WINDOWS₩system32₩cmd.exe
Enter the number : 1
Enter its coefficient : 2
Enter the nubmer : 3
Enter its coffcient : 1
Enter the nubmer : -1
Enter the multiply number4
4 x 2
12 x 1
계속하려면 아무 키나 누르십시오 . . .
```

27.

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

struct node
{
	int data;
	struct node *next;
};

struct node *start1=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);

int main()
{
	int count;
	start1=create(start1,&count);
	printf("number of non-zero elements = %d",count);
}

struct node *create(struct node *start,int*count)
{
	struct node *new_node, *ptr;
	int num;
	*count=0;
```

```
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(new_node->data!=0)
                {
                        *count=*count+1;
                }
                if(start==NULL)
                {
                        new_node->next=new_node;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=start)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=start;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;

}
```


C:₩WINDOWS₩system32₩cmd.exe

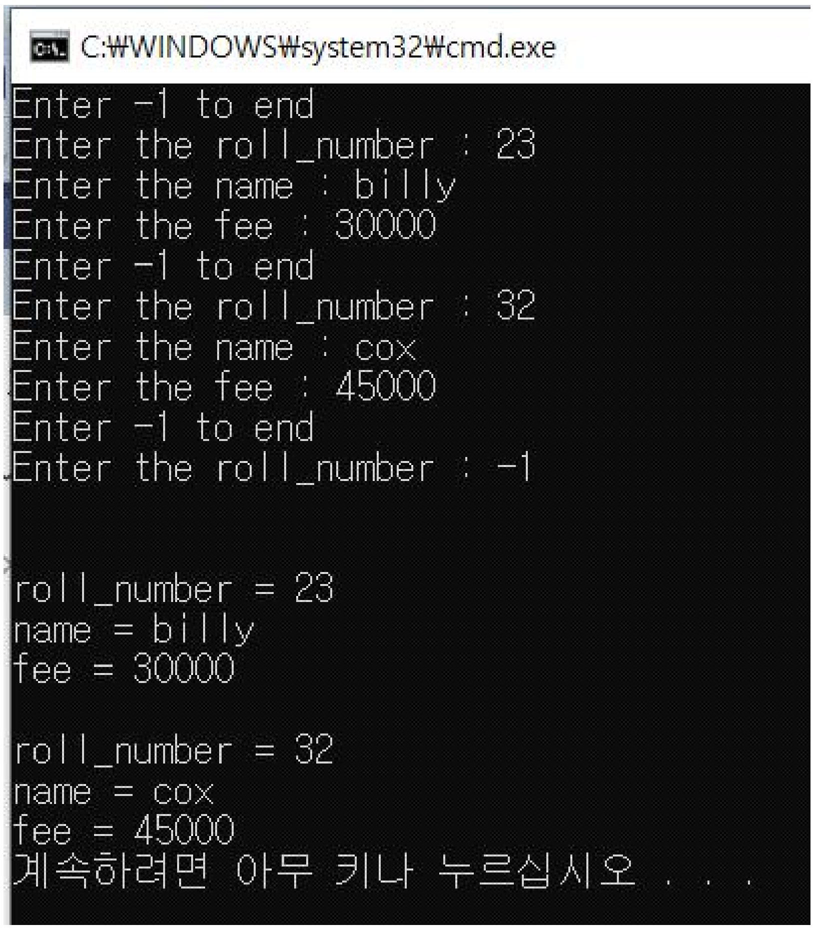
```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 0
Enter the data : 0
Enter the data : 5
Enter the data : -1
number of non-zero elements = 3계속하려면 아무 키나 누르십시오 . . .
```

28.

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
struct node
{
        int roll_no;
        char name[10];
        int fee;
        struct node *next;
};

struct node *start1=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);

int main()
{
        int count;
        start1=create(start1);
        printf("\n\n");
        display(start1);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int roll;
        char name[10];
        int fee;
        printf("Enter -1 to end\n");
        printf("Enter the roll_number : ");
        scanf("%d",&roll);
        printf("Enter the name : ");
```

```
		scanf("%s",name);
		printf("Enter the fee : ");
		scanf("%d",&fee);
		while(roll!=-1)
		{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->roll_no=roll;
			strcpy(new_node->name,name);
			new_node->fee=fee;

			if(start==NULL)
			{
				new_node->next=new_node;
				start=new_node;
			}
			else
			{
				ptr=start;
				while(ptr->next!=start)
					ptr=ptr->next;
				ptr->next=new_node;
				new_node->next=start;
			}
			printf("Enter -1 to end\n");
			printf("Enter the roll_number : ");
			scanf("%d",&roll);
			if(roll==-1)
				break;
			printf("Enter the name : ");
			scanf("%s",name);
			printf("Enter the fee : ");
			scanf("%d",&fee);
		}
		return start;
}

struct node *display(struct node *start)
{
		struct node *ptr;
		ptr=start;
		while(ptr->next!=start)
		{
			printf("roll_number = %d\n",ptr->roll_no);
```

```
                printf("name = %s\n",ptr->name);
                printf("fee = %d\n",ptr->fee);
                printf("\n");
                ptr=ptr->next;
        }
                printf("roll_number = %d\n",ptr->roll_no);
                printf("name = %s\n",ptr->name);
                printf("fee = %d\n",ptr->fee);
        return start;

}
```



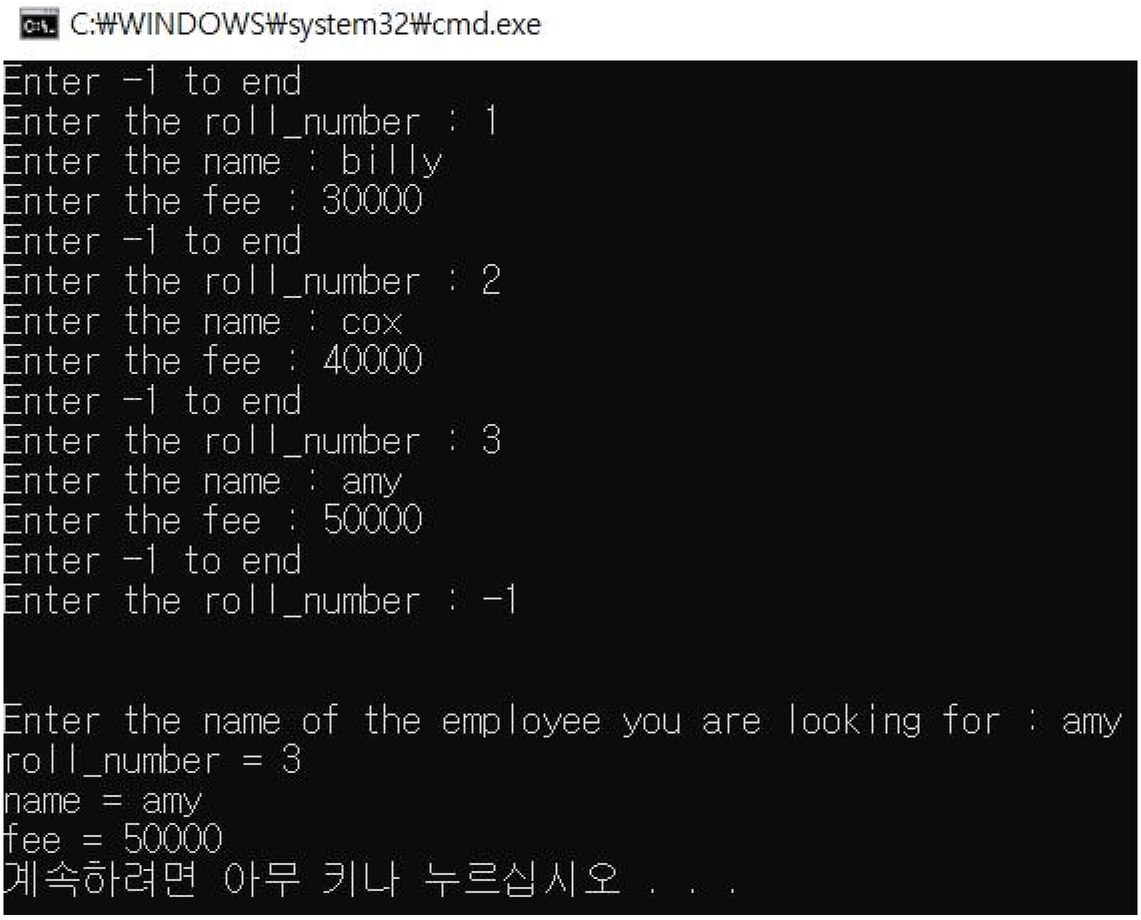
C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the roll_number : 23
Enter the name : billy
Enter the fee : 30000
Enter -1 to end
Enter the roll_number : 32
Enter the name : cox
Enter the fee : 45000
Enter -1 to end
Enter the roll_number : -1


roll_number = 23
name = billy
fee = 30000

roll_number = 32
name = cox
fee = 45000
계속하려면 아무 키나 누르십시오 . . .
```

29.

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
struct node
{
        int roll_no;
        char name[10];
        int fee;
        struct node *next;
};

struct node *start1=NULL;
struct node *create(struct node *start);
struct node *given_display(struct node *start);

int main()
{
        int count;
        start1=create(start1);
        printf("\n\n");
        given_display(start1);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int roll;
        char name[10];
        int fee;
        printf("Enter -1 to end\n");
        printf("Enter the roll_number : ");
        scanf("%d",&roll);
        printf("Enter the name : ");
        scanf("%s",name);
        printf("Enter the fee : ");
        scanf("%d",&fee);
        while(roll!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->roll_no=roll;
                strcpy(new_node->name,name);
                new_node->fee=fee;

                if(start==NULL)
                {
                        new_node->next=new_node;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=start)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=start;
                }
                printf("Enter -1 to end\n");
                printf("Enter the roll_number : ");
                scanf("%d",&roll);
                if(roll==-1)
                        break;
                printf("Enter the name : ");
                scanf("%s",name);
                printf("Enter the fee : ");
                scanf("%d",&fee);
        }
        return start;
}

struct node *given_display(struct node *start)
{
        struct node *ptr;
        char name[10];
        ptr=start;

        printf("Enter the name of the employee you are looking for : ");
        scanf("%s",name);
        while(ptr->next!=start)
        {
                if(strcmp(ptr->name,name)==0)
                {
                        printf("roll_number = %d\n",ptr->roll_no);
                        printf("name = %s\n",ptr->name);
                        printf("fee = %d\n",ptr->fee);
                        printf("\n");
                }
                ptr=ptr->next;
        }
        if(ptr->next==start)
        {
                if(strcmp(ptr->name,name)==0)
```

```
			{
				printf("roll_number = %d\n",ptr->roll_no);
				printf("name = %s\n",ptr->name);
				printf("fee = %d\n",ptr->fee);
			}
		}
		return start;

}
```



C:₩WINDOWS₩system32₩cmd.exe

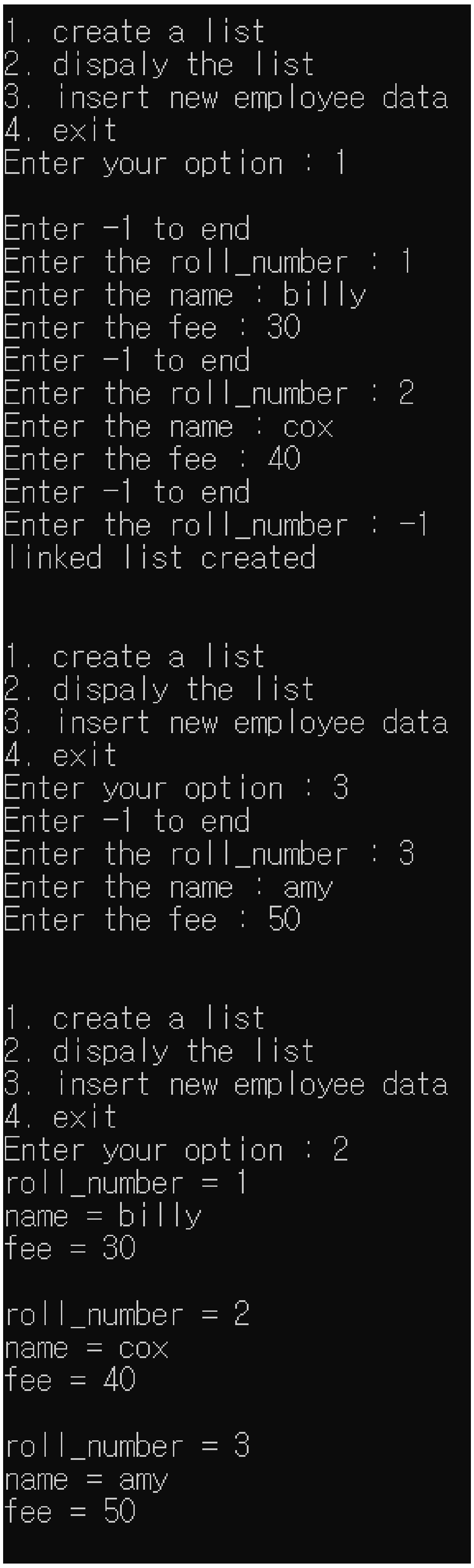
```
Enter -1 to end
Enter the roll_number : 1
Enter the name : billy
Enter the fee : 30000
Enter -1 to end
Enter the roll_number : 2
Enter the name : cox
Enter the fee : 40000
Enter -1 to end
Enter the roll_number : 3
Enter the name : amy
Enter the fee : 50000
Enter -1 to end
Enter the roll_number : -1


Enter the name of the employee you are looking for : amy
roll_number = 3
name = amy
fee = 50000
계속하려면 아무 키나 누르십시오 . . .
```

30.

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
struct node
{
	int roll_no;
	char name[10];
	int fee;
	struct node *next;
};
```

```
struct node *start1=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *insert_end(struct node *start);

int main()
{
	int option;
	do
	{
		printf("\n\n");
		printf("1. create a list\n");
		printf("2. dispaly the list\n");
		printf("3. insert new employee data\n");
		printf("4. exit\n");
		printf("Enter your option : ");
		scanf("%d",&option);
		switch(option)
		{
			case 1:start1=create(start1);
				printf("linked list created\n");
				break;
			case 2:start1=display(start1);
				break;
			case 3:start1=insert_end(start1);
				break;
		}
	}while(option!=4);
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int roll;
	char name[10];
	int fee;
	printf("\n");
	printf("Enter -1 to end\n");
	printf("Enter the roll_number : ");
	scanf("%d",&roll);
	printf("Enter the name : ");
	scanf("%s",name);
```

```
		printf("Enter the fee : ");
		scanf("%d",&fee);
		while(roll!=-1)
		{
			new_node=(struct node*)malloc(sizeof(struct node));
			new_node->roll_no=roll;
			strcpy(new_node->name,name);
			new_node->fee=fee;

			if(start==NULL)
			{
				new_node->next=new_node;
				start=new_node;
			}
			else
			{
				ptr=start;
				while(ptr->next!=start)
					ptr=ptr->next;
				ptr->next=new_node;
				new_node->next=start;
			}
			printf("Enter -1 to end\n");
			printf("Enter the roll_number : ");
			scanf("%d",&roll);
			if(roll==-1)
				break;
			printf("Enter the name : ");
			scanf("%s",name);
			printf("Enter the fee : ");
			scanf("%d",&fee);
		}
		return start;
}

struct node *display(struct node *start)
{
		struct node *ptr;
		ptr=start;
		while(ptr->next!=start)
		{
			printf("roll_number = %d\n",ptr->roll_no);
			printf("name = %s\n",ptr->name);
			printf("fee = %d\n",ptr->fee);
			printf("\n");
			ptr=ptr->next;
		}
			printf("roll_number = %d\n",ptr->roll_no);
			printf("name = %s\n",ptr->name);
			printf("fee = %d\n",ptr->fee);
		return start;


}

struct node *insert_end(struct node *start)
{
		struct node *ptr,*new_node;
		int roll,fee;
		char name[10];
		printf("Enter -1 to end\n");
		printf("Enter the roll_number : ");
		scanf("%d",&roll);
		printf("Enter the name : ");
		scanf("%s",name);
		printf("Enter the fee : ");
		scanf("%d",&fee);
		ptr=start;
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->roll_no=roll;
		strcpy(new_node->name,name);
		new_node->fee=fee;
		while(ptr->next!=start)
			ptr=ptr->next;
		ptr->next=new_node;
		new_node->next=start;
		return start;

}
```

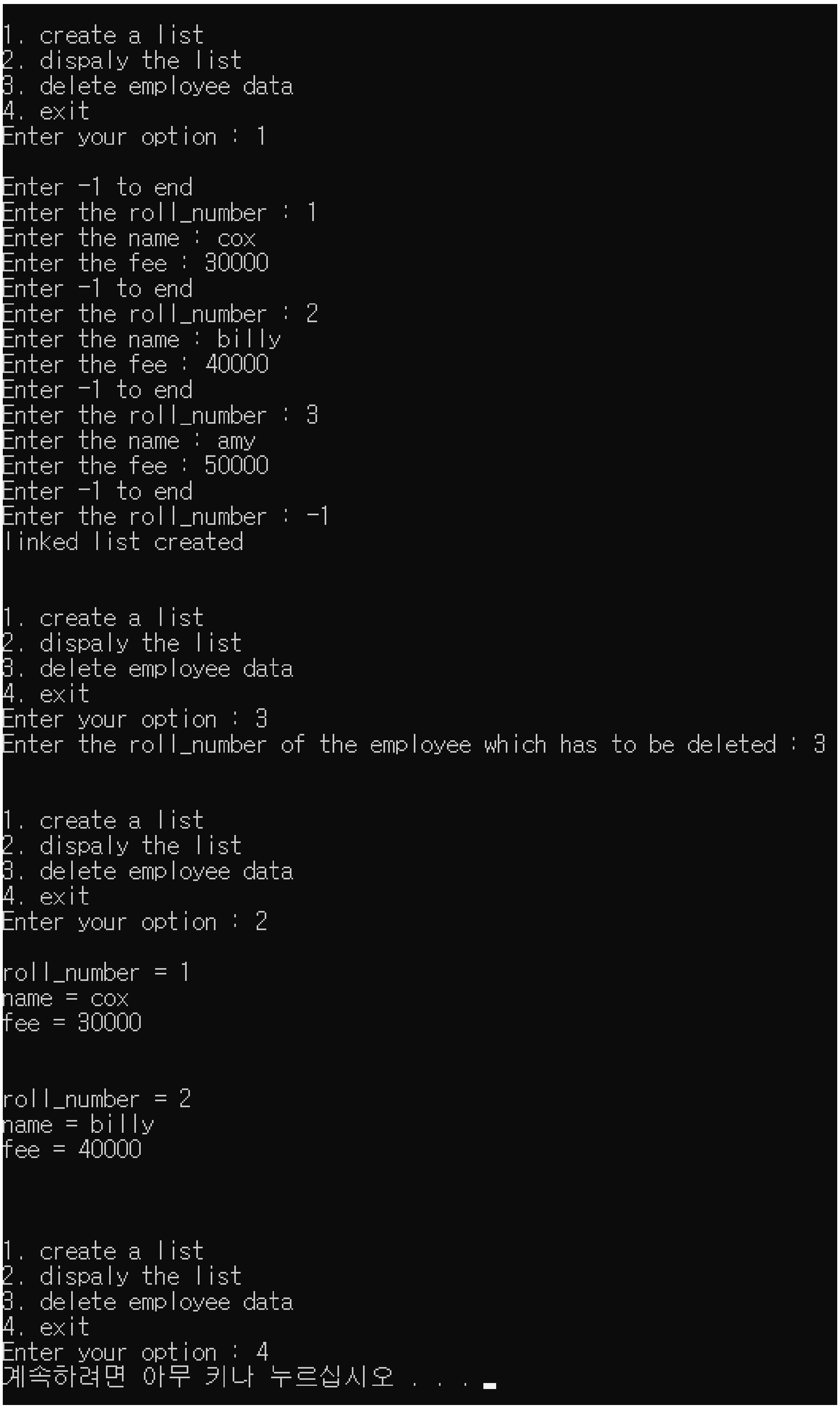


```
1. create a list
2. dispaly the list
3. insert new employee data
4. exit
Enter your option : 1

Enter -1 to end
Enter the roll_number : 1
Enter the name : billy
Enter the fee : 30
Enter -1 to end
Enter the roll_number : 2
Enter the name : cox
Enter the fee : 40
Enter -1 to end
Enter the roll_number : -1
linked list created


1. create a list
2. dispaly the list
3. insert new employee data
4. exit
Enter your option : 3
Enter -1 to end
Enter the roll_number : 3
Enter the name : amy
Enter the fee : 50


1. create a list
2. dispaly the list
3. insert new employee data
4. exit
Enter your option : 2
roll_number = 1
name = billy
fee = 30

roll_number = 2
name = cox
fee = 40

roll_number = 3
name = amy
fee = 50
```

31.
```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
struct node
{
        int roll_no;
        char name[10];
        int fee;
        struct node *next;
};

struct node *start1=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_node(struct node *start);
struct node *delete_beg(struct node *start);


int main()
{
        int option;
        do
        {
                printf("\n\n");
                printf("1. create a list\n");
                printf("2. dispaly the list\n");
                printf("3. delete employee data\n");
                printf("4. exit\n");
                printf("Enter your option : ");
                scanf("%d",&option);
                switch(option)
                {
                        case 1:start1=create(start1);
                                printf("linked list created\n");
                                break;
                        case 2:start1=display(start1);
                                break;
                        case 3:start1=delete_node(start1);
                                break;

                }
        }while(option!=4);
```

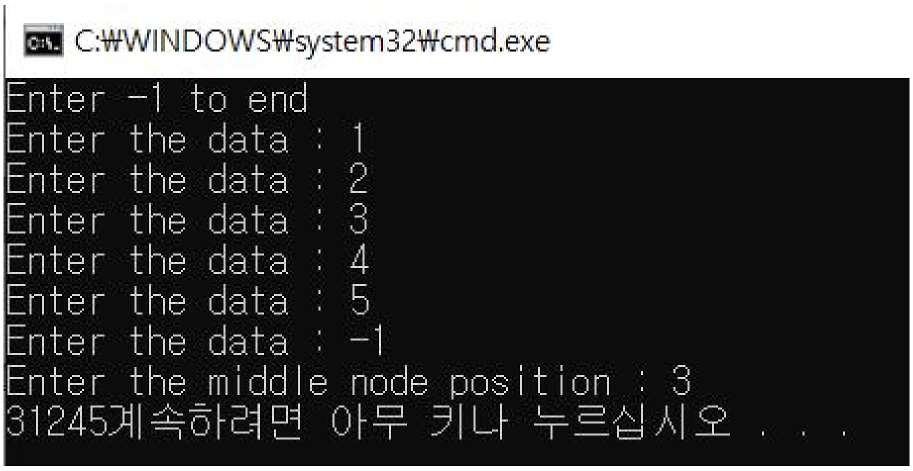
```
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int roll;
	char name[10];
	int fee;
	printf("\n");
	printf("Enter -1 to end\n");
	printf("Enter the roll_number : ");
	scanf("%d",&roll);
	printf("Enter the name : ");
	scanf("%s",name);
	printf("Enter the fee : ");
	scanf("%d",&fee);
	while(roll!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->roll_no=roll;
		strcpy(new_node->name,name);
		new_node->fee=fee;

		if(start==NULL)
		{
			new_node->next=NULL;
			start=new_node;
		}
		else
		{
			ptr=start;
			while(ptr->next!=NULL)
				ptr=ptr->next;
			ptr->next=new_node;
			new_node->next=NULL;
		}
		printf("Enter -1 to end\n");
		printf("Enter the roll_number : ");
		scanf("%d",&roll);
		if(roll==-1)
			break;
		printf("Enter the name : ");
		scanf("%s",name);
```

```
                printf("Enter the fee : ");
                scanf("%d",&fee);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("\n");
                printf("roll_number = %d\n",ptr->roll_no);
                printf("name = %s\n",ptr->name);
                printf("fee = %d\n",ptr->fee);
                printf("\n");
                ptr=ptr->next;
        }
        return start;


}

struct node *delete_node(struct node *start)
{
        struct node *ptr,*preptr;
        int roll;
        printf("Enter the roll_number of the employee which has to be deleted : ");
        scanf("%d",&roll);
        ptr=start;
        if(ptr->roll_no==roll)
        {
                start=delete_beg(start);
                return start;
        }
        else
        {
                while(ptr->roll_no!=roll)
                {
                        preptr=ptr;
                        ptr=ptr->next;
                }
                preptr->next=ptr->next;
                free(ptr);
                return start;
        }
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        free(ptr);
        return start;
}
```



```
1. create a list
2. dispaly the list
3. delete employee data
4. exit
Enter your option : 1

Enter -1 to end
Enter the roll_number : 1
Enter the name : cox
Enter the fee : 30000
Enter -1 to end
Enter the roll_number : 2
Enter the name : billy
Enter the fee : 40000
Enter -1 to end
Enter the roll_number : 3
Enter the name : amy
Enter the fee : 50000
Enter -1 to end
Enter the roll_number : -1
linked list created


1. create a list
2. dispaly the list
3. delete employee data
4. exit
Enter your option : 3
Enter the roll_number of the employee which has to be deleted : 3


1. create a list
2. dispaly the list
3. delete employee data
4. exit
Enter your option : 2

roll_number = 1
name = cox
fee = 30000


roll_number = 2
name = billy
fee = 40000



1. create a list
2. dispaly the list
3. delete employee data
4. exit
Enter your option : 4
계속하려면 아무 키나 누르십시오 . . .
```

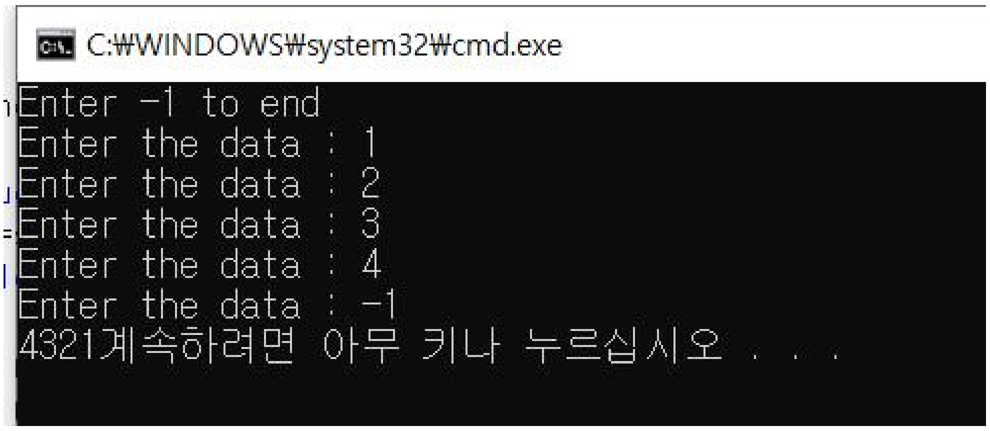
32.
```
#include<stdio.h>
#include <malloc.h>
struct node
{
        int data;
        struct node *next;
        struct node *prev;
        int pos;
};
struct node *start=NULL;
struct node *display(struct node *start);
struct node *re_set(struct node *start);
struct node *create(struct node *start);

int main()
{
        start=create(start);
        start=re_set(start);
        start=display(start);

}
struct node *create(struct node *start)
{
        struct node *new_node,*ptr;
        int num;
        int pos=1;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->pos=pos;
                new_node->data=num;

                pos++;
                if(start==NULL)
                {
                        new_node->prev=NULL;
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->prev=ptr;
                        new_node->next=NULL;

                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *re_set(struct node *start)
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the middle node position : ");
        scanf("%d",&val);
        ptr=start;
        while(ptr->pos!=val)
        {
                        preptr=ptr;
                        ptr=ptr->next;
        }
                preptr->next=ptr->next;
                ptr->next->prev=preptr;
                ptr->next=start;
                start->prev=ptr;
                start=ptr;
                return start;
}
```



```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
Enter the middle node position : 3
31245계속하려면 아무 키나 누르십시오 . . .
```

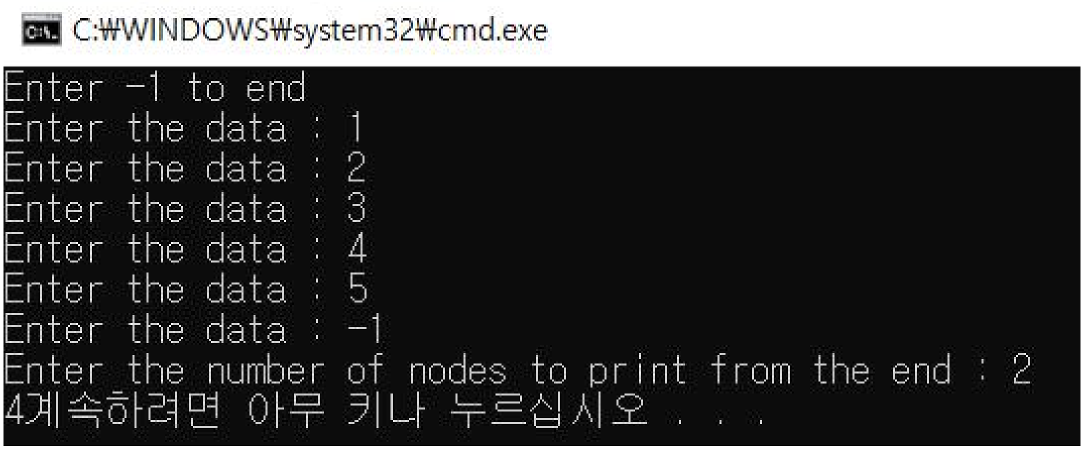
33.

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

struct node {
        int data;
        struct node *next;
};

struct node *start=NULL;
struct node *prev=NULL;
struct node *next=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *reverse(struct node *start,struct node *prev, struct node *next);
int main()
{
        start=create(start);
        start=reverse(start,prev,next);
        start=display(start);
}
struct node *create(struct node *start)
{
        int num;
        struct node *new_node , *ptr;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=NULL;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *reverse(struct node *start,struct node *prev, struct node *next)
{
        struct node *ptr=start;
        struct node *ptr2=prev;
        while(ptr->next!=NULL)
        {
                next=ptr->next;
```

```
            ptr->next=ptr2  ;
            ptr2=ptr;
            ptr=next;
        }
        ptr->next=ptr2;
        return ptr;
}
```


```
C:₩WINDOWS₩system32₩cmd.exe
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : -1
4321계속하려면 아무 키나 누르십시오 . . .
```

34.
```
#include <stdio.h>
#include <malloc.h>
struct node
{
        int data;
        struct node *next;
        int pos;
};
struct node *start=NULL;
struct node *display(struct node *start);
struct node *create(struct node *start);
struct node *delete_node(struct node *start);
struct node *delete_beg(struct node *start);
int main()
{       int count;
        start=create(start,&count);
        start=display(start,count);

}
struct node *create(struct node *start, int *count)
{
        struct node *new_node,*ptr;
        int num;
        int pos=1;
        *count=0;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                new_node->pos=pos;
                pos++;
                if(start==NULL)
                {
                        new_node->next=NULL;
                        start=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=NULL;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
                *count=*count+1;
        }
        return start;
}

struct node *display(struct node *start, int count)
{
        struct node *ptr;
        int num;

        printf("Enter the number of nodes to print from the end : ");
        scanf("%d",&num);
        num=(count+1)-num;
        ptr=start;
        while(ptr!=NULL)
        {
                if(ptr->pos==num)
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}

struct node *delete_node(struct node *start)
{
        struct node *ptr, *preptr;
        int val;
        printf("Enter the position of the node which has to be deleted : ");
        scanf("%d",&val);
        ptr=start;
        if(ptr->pos==val)
        {
                start=delete_beg(start);
                return start;
        }
        else
        {
                while(ptr->pos!=val)
                {
                        preptr=ptr;
                        ptr=ptr->next;
                }
                preptr->next=ptr->next;
                free(ptr);
                return start;
        }
}

struct node *delete_beg(struct node *start)
{
        struct node *ptr;
        ptr=start;
        start=start->next;
        free(ptr);
        return start;
}
```



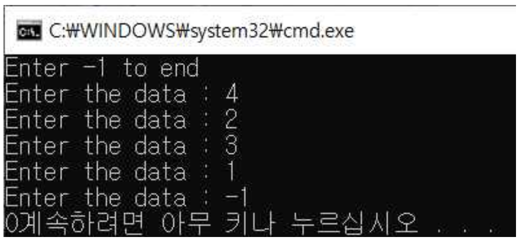
C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
Enter the number of nodes to print from the end : 2
4계속하려면 아무 키나 누르십시오 . . .
```

35.

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

struct node{
	int data;
	struct node *next;

};
struct node *start=NULL;
struct node *create(struct node *start);
void isSorted(struct node *start);
int main ()
{
	start=create(start);
	isSorted(start);
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		if(start==NULL)
		{	new_node->next=NULL;
			start=new_node;
```

```
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=NULL;

                }
                printf("Enter the data : ");
                scanf("%d", &num);

        }
        return start;
}

void isSorted(struct node *start)
{
        struct node *ptr, *preptr;
        int temp;
        int flag;

        ptr=start;
        while(ptr->next!=NULL)
        {
                preptr=ptr;
                ptr=ptr->next;
                if(preptr->data<ptr->data)
                        flag=1;
                else
                        flag=0;

        }
        printf("%d",flag);

}
```


C:₩WINDOWS₩system32₩cmd.exe

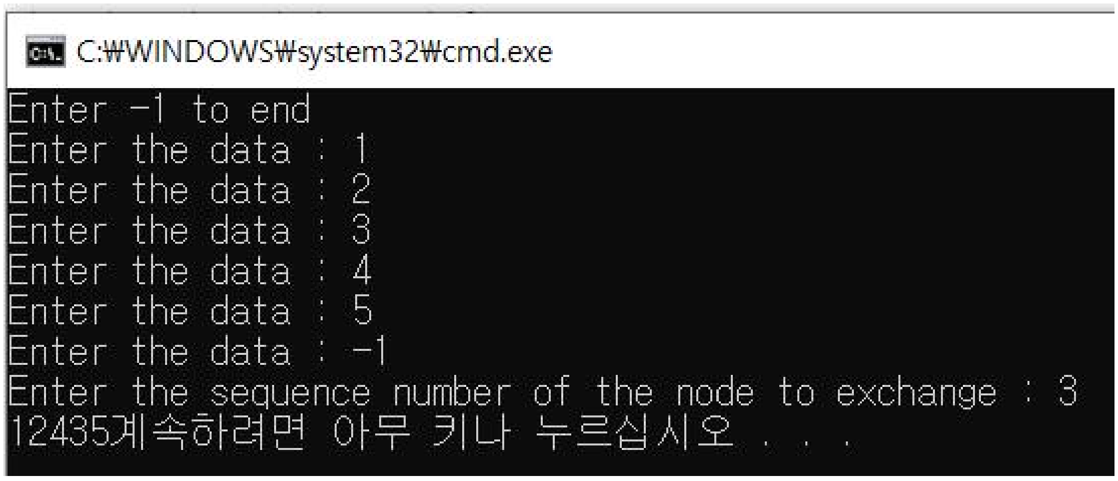
```
Enter -1 to end
Enter the data : 4
Enter the data : 2
Enter the data : 3
Enter the data : 1
Enter the data : -1
0계속하려면 아무 키나 누르십시오 . . .
```

36.

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



struct node
{
	int data;
	struct node *next;
	struct node *prev;
	int pos;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *change(struct node *start);
int main()
{
	start=create(start);
	start=change(start);
	start=display(start);
}

struct node *create(struct node *start)
{
	struct node *new_node, *ptr;
	int num;
	int pos=1;
	printf("Enter -1 to end\n");
	printf("Enter the data : ");
	scanf("%d",&num);
	while(num!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=num;
		new_node->pos=pos;
		pos++;
		if(start==NULL)
		{
			new_node->prev=NULL;
			new_node->next=new_node;
			start=new_node;
```

```
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=start)
                                ptr=ptr->next;
                        new_node->prev=ptr;
                        ptr->next=new_node;
                        new_node->next=start;
                        start->prev=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr->next!=start)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        printf("%d",ptr->data);
        return start;
}


struct node *change(struct node *start)
{
        struct node *ptr1 , *ptr2;
        int pos1,pos2,temp;
        printf("Enter the sequence number of the node to exchange : ");
        scanf("%d",&pos1);
        pos2=pos1+1;
        ptr1=start;
        ptr2=start;
        while(ptr1->pos!=pos1)
        {
```

```
		ptr1=ptr1->next;
	}
	while(ptr2->pos!=pos2)
	{
		ptr2=ptr2->next;
	}
	temp=ptr1->data;
	ptr1->data=ptr2->data;
	ptr2->data=temp;
	return start;
}
```

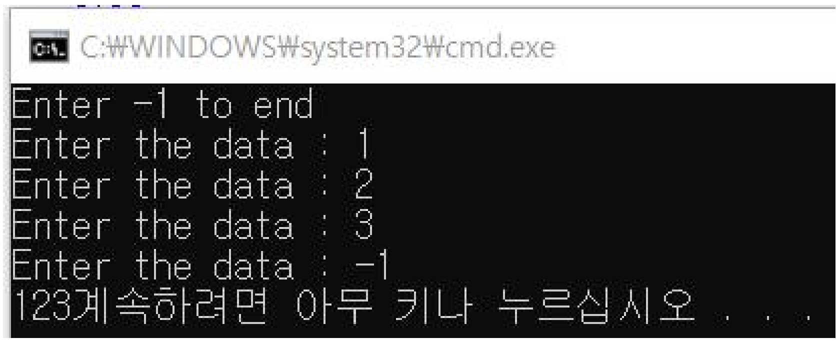


C:₩WINDOWS₩system32₩cmd.exe

```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
Enter the sequence number of the node to exchange : 3
12435계속하려면 아무 키나 누르십시오 . . .
```

37.

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

struct node
{
	int data;
	struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
int main()
{
	start=create(start);
	start=display(start);
}

struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                new_node->next=NULL;
                if(start==NULL)
                {
                        start=(struct node*)malloc(sizeof(struct node));
                        start->next=new_node;

                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;

                }
                printf("Enter the data : ");
                scanf("%d", &num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start->next;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }

}
```



```
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : -1
123계속하려면 아무 키나 누르십시오 . . .
```

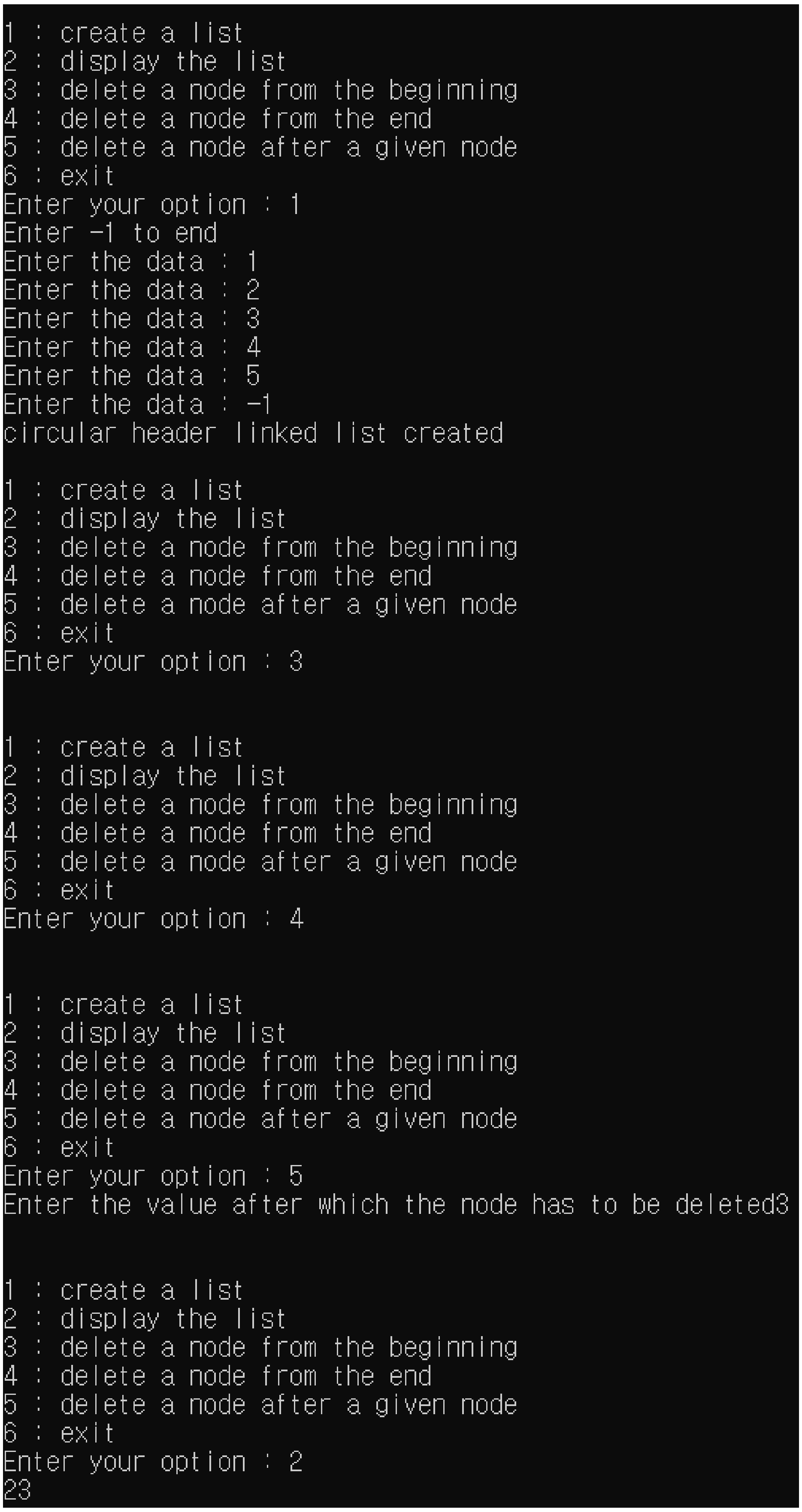
38.

```
#include <stdio.h>
#include <malloc.h>
struct node
{
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node *delete_beg(struct node *start);
struct node *delete_end(struct node *start);
struct node *delete_after(struct node *start);

int main()
{
        int option;
        do{
        printf("\n\n");
        printf("1 : create a list\n");
        printf("2 : display the list\n");
        printf("3 : delete a node from the beginning\n");
        printf("4 : delete a node from the end\n");
        printf("5 : delete a node after a given node\n");
        printf("6 : exit\n");
        printf("Enter your option : ");
        scanf("%d",&option);
                switch(option)
                {
                        case 1: start=create(start);
                        printf("circular header linked list created");
                        break;
                        case 2: start=display(start);
                        break;
```

```
                    case 3: start=delete_beg(start);
                    break;
                    case 4: start=delete_end(start);
                    break;
                    case 5: start=delete_after(start);
                    break;
        }

        }while(option!=6);
}
struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;

                if(start==NULL)
                {
                        start=(struct node*)malloc(sizeof(struct node));
                        start->next=new_node;
                        new_node->next=start;

                }
                else
                {
                        ptr=start;
                        while(ptr->next!=start)
                                ptr=ptr->next;
                        ptr->next=new_node;
                        new_node->next=start;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}
```

```
struct node *display(struct node *start)
{
	struct node *ptr;
	ptr=start->next;
	while(ptr->next!=start)
	{
		printf("%d",ptr->data);
		ptr=ptr->next;
	}
	printf("%d",ptr->data);
	return start;
}

struct node *delete_beg(struct node *start)
{
	struct node *ptr, *temp;
	ptr=start->next;
	while(ptr->next!=start)
		ptr=ptr->next;
	temp=start->next;
	start->next=start->next->next;
	free(temp);
	return start;
}

struct node *delete_end(struct node *start)
{
	struct node *ptr, *preptr;
	ptr=start;
	while(ptr->next!=start)
	{
		preptr=ptr;
		ptr=ptr->next;
	}
	preptr->next=ptr->next;
	free(ptr);
	return start;
}

struct node *delete_after(struct node *start)
{
	struct node *ptr, *preptr;
	int val;
```

```
        printf("Enter the value after which the node has to be deleted");
        scanf("%d",&val);
        ptr=start;
        preptr=ptr;
        while(preptr->data!=val)
        {
                preptr=ptr;
                ptr=ptr->next;
        }
        preptr->next=ptr->next;
        if(ptr==start)
                start=preptr->next;
        free(ptr);
        return start;
}
```

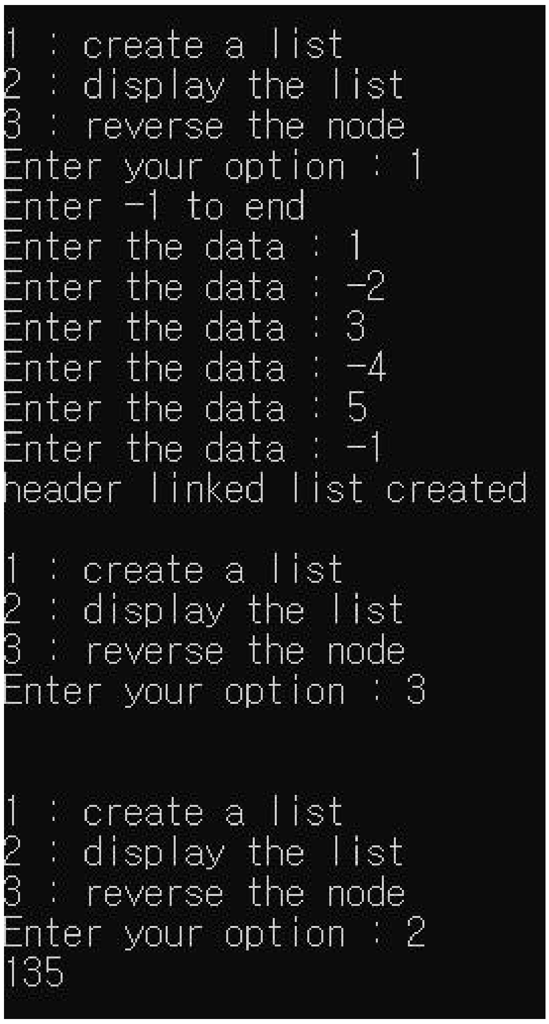


```
1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : 2
Enter the data : 3
Enter the data : 4
Enter the data : 5
Enter the data : -1
circular header linked list created

1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 3


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 4


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 5
Enter the value after which the node has to be deleted3


1 : create a list
2 : display the list
3 : delete a node from the beginning
4 : delete a node from the end
5 : delete a node after a given node
6 : exit
Enter your option : 2
23
```

39.
```
#include <stdio.h>
#include <malloc.h>
struct node
{
        int data;
        struct node *next;
};
struct node *start=NULL;
struct node *create(struct node *start);
struct node *display(struct node *start);
struct node* find_delete(struct node *start);
struct node *delete_beg(struct node *start);
int main()
{
        int option;
        do{
        printf("\n\n");
        printf("1 : create a list\n");
        printf("2 : display the list\n");
        printf("3 : delete the negative element\n");
        printf("Enter your option : ");
        scanf("%d",&option);
                switch(option)
                {
                        case 1: start=create(start);
                        printf("header linked list created");
                        break;

                        case 2: start=display(start);
                        break;

                        case 3: start=find_delete(start);
                        break;

        }

        }while(option!=6);
}
struct node *create(struct node *start)
{
        struct node *new_node, *ptr;
        int num;
        printf("Enter -1 to end\n");
        printf("Enter the data : ");
        scanf("%d",&num);
        while(num!=-1)
        {
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=num;
                new_node->next=NULL;
                if(start==NULL)
                {
                        start=(struct node*)malloc(sizeof(struct node));
                        start->next=new_node;
                }
                else
                {
                        ptr=start;
                        while(ptr->next!=NULL)
                                ptr=ptr->next;
                        ptr->next=new_node;
                }
                printf("Enter the data : ");
                scanf("%d",&num);
        }
        return start;
}

struct node *display(struct node *start)
{
        struct node *ptr;
        ptr=start->next;
        while(ptr!=NULL)
        {
                printf("%d",ptr->data);
                ptr=ptr->next;
        }
        return start;
}
struct node* find_delete(struct node *start)
{
        struct node *ptr,*preptr,*temp, *temp2;
        ptr=start->next;
        if(ptr->data<0){
```

```
                                        if(ptr==start->next)
                                        start=delete_beg(start);
                        }
            ptr=start->next;
            while(ptr!=NULL)
            {

                        if(ptr->data<0)
                                    {
                                                if(ptr->next!=NULL)
                                                            preptr=ptr;
                                    }
                        else
                        {
                                    if(ptr->next==NULL)
                                                break;
                          preptr=ptr;
                          ptr=ptr->next;
                        }

                        if(ptr->data<0)
                        {
                                    if(ptr->next==NULL)
                                    {
                                                          temp2=ptr;
                                                          preptr->next=NULL;
                                                          free(temp2);
                                                          break;
                                    }
                                    temp=ptr;
                                    preptr->next=ptr->next;
                                    ptr=ptr->next;
                                    free(temp);

                        }
            }
            return start;
}
struct node *delete_beg(struct node *start)
{
            struct node *ptr, *temp;
            ptr=start->next;
            start->next=ptr->next;
```

```
		free(ptr);
		return start;
}
```



```
1 : create a list
2 : display the list
3 : reverse the node
Enter your option : 1
Enter -1 to end
Enter the data : 1
Enter the data : -2
Enter the data : 3
Enter the data : -4
Enter the data : 5
Enter the data : -1
header linked list created

1 : create a list
2 : display the list
3 : reverse the node
Enter your option : 3


1 : create a list
2 : display the list
3 : reverse the node
Enter your option : 2
135
```