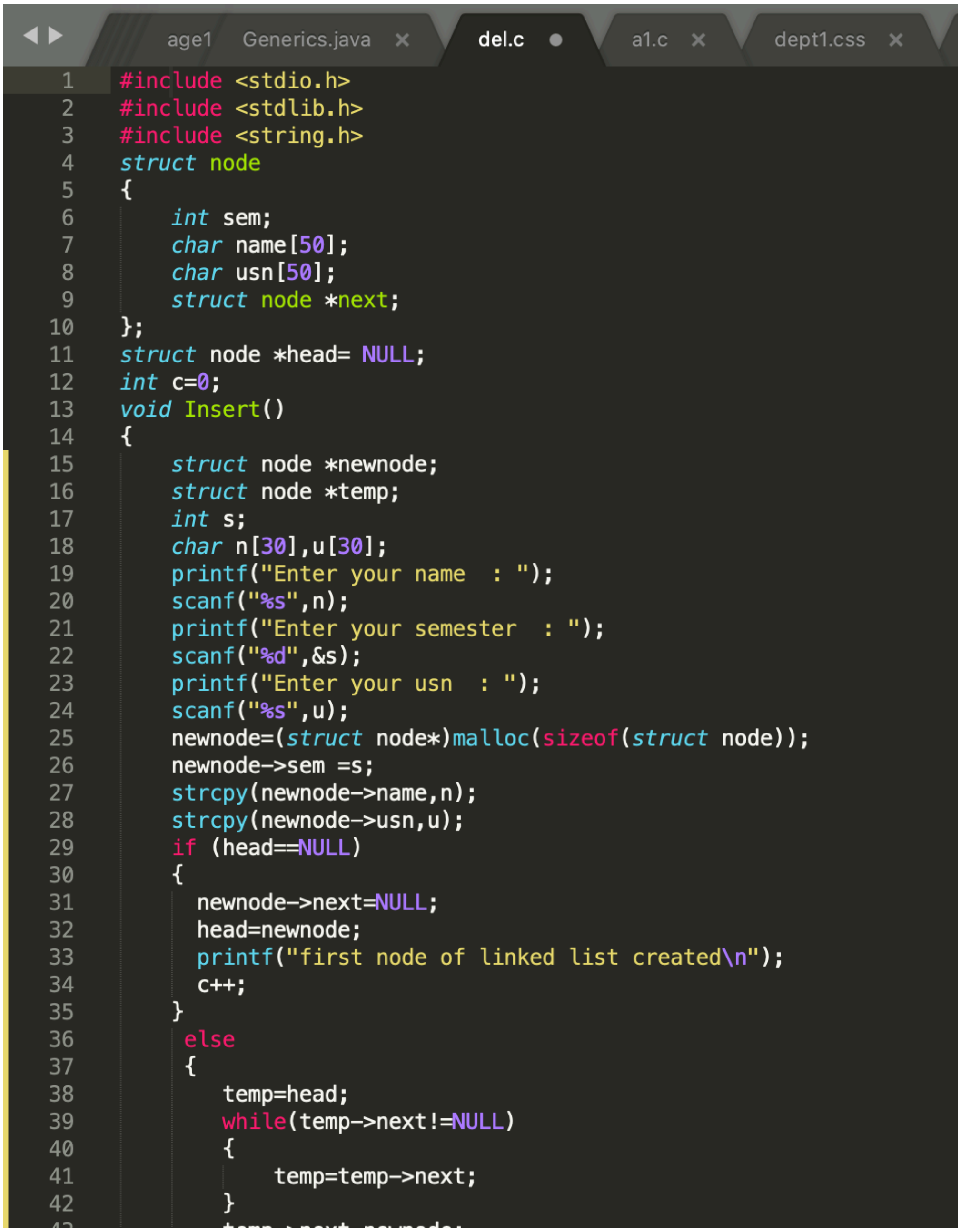
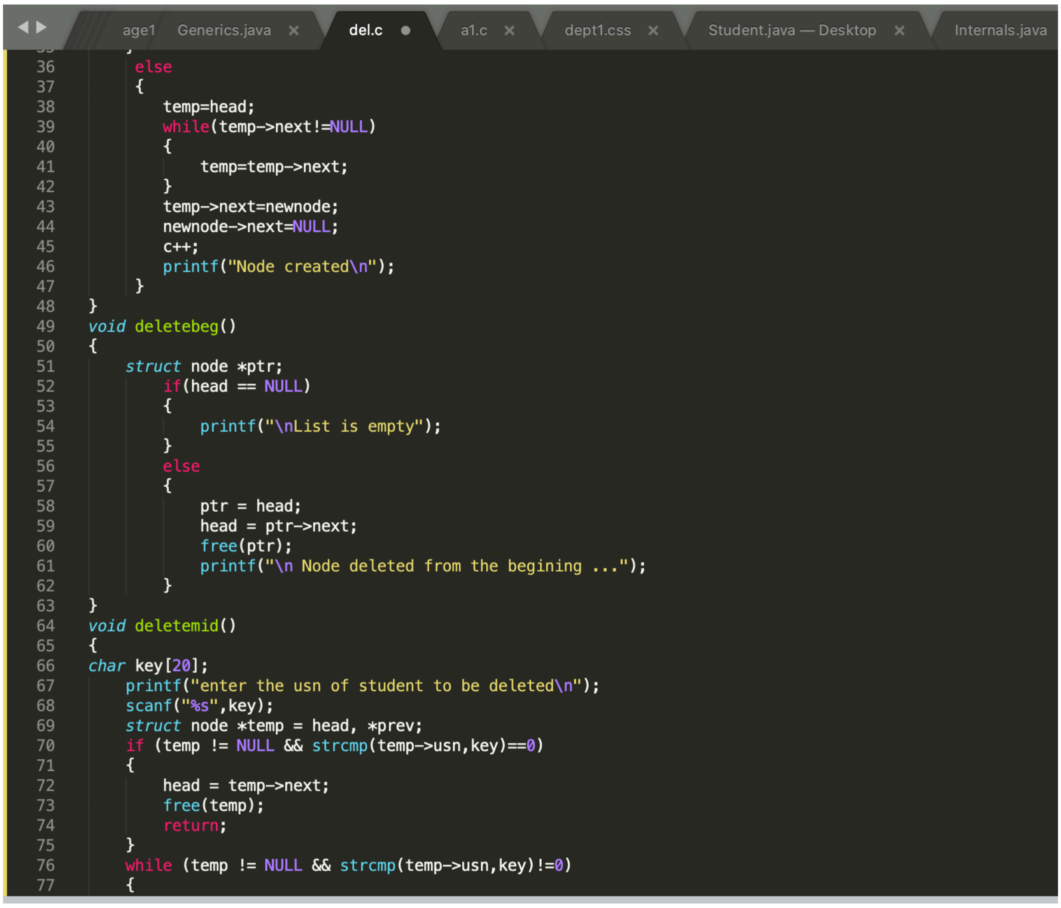
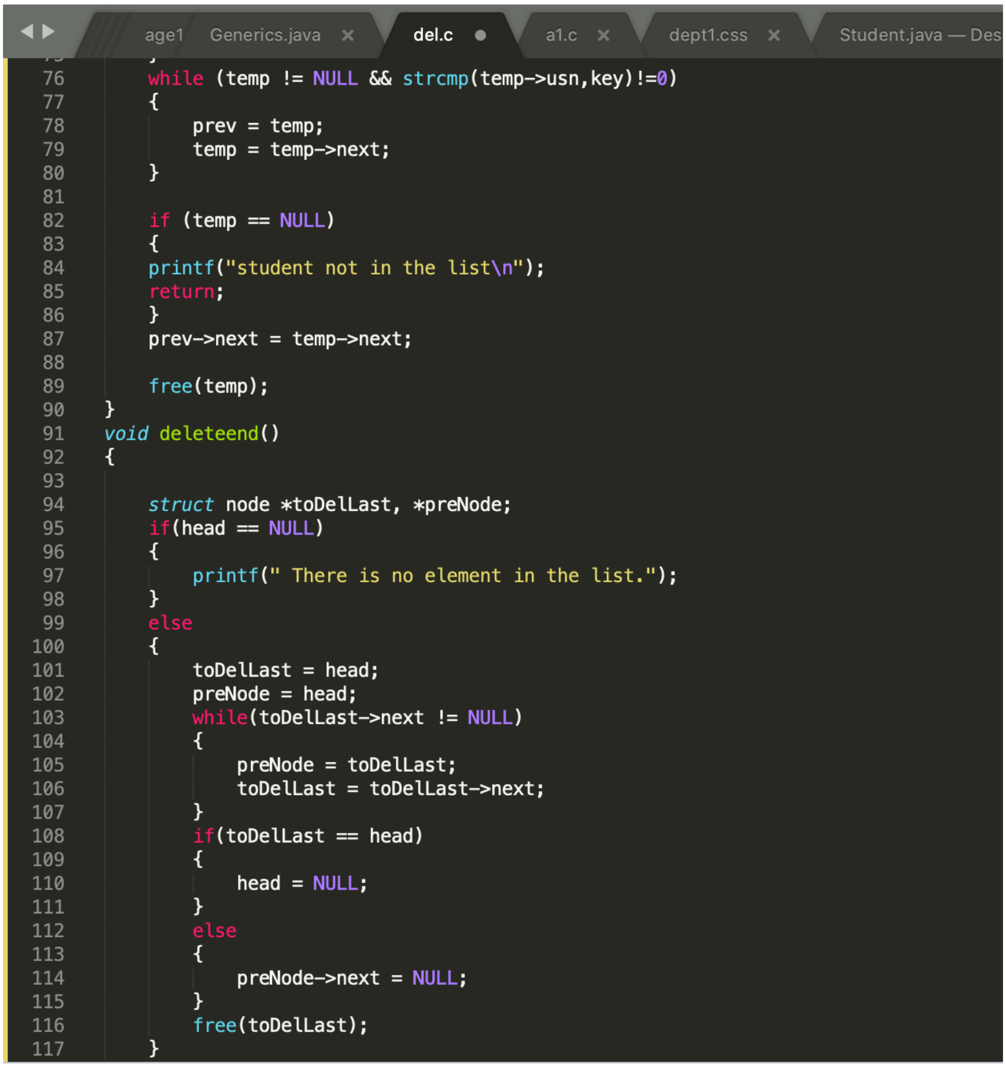


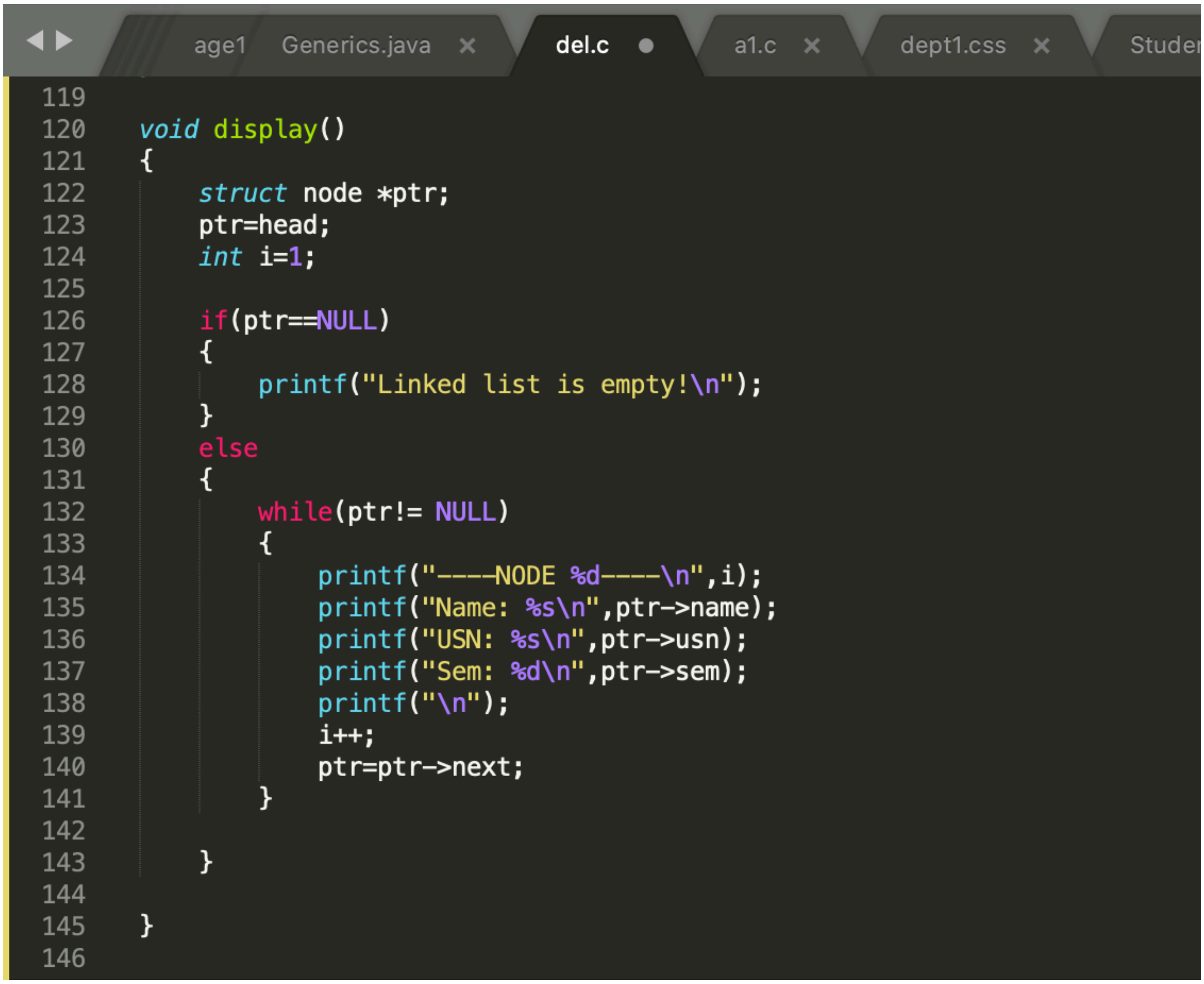
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct node
{
    int sem;
    char name[50];
    char usn[50];
    struct node *next;
};
struct node *head= NULL;
int c=0;
void Insert()
{
    struct node *newnode;
    struct node *temp;
    int s;
    char n[30],u[30];
    printf("Enter your name  : ");
    scanf("%s",n);
    printf("Enter your semester  : ");
    scanf("%d",&s);
    printf("Enter your usn  : ");
    scanf("%s",u);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->sem =s;
    strcpy(newnode->name,n);
    strcpy(newnode->usn,u);
    if (head==NULL)
    {
      newnode->next=NULL;
      head=newnode;
      printf("first node of linked list created\n");
      c++;
    }
     else
     {
        temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
```



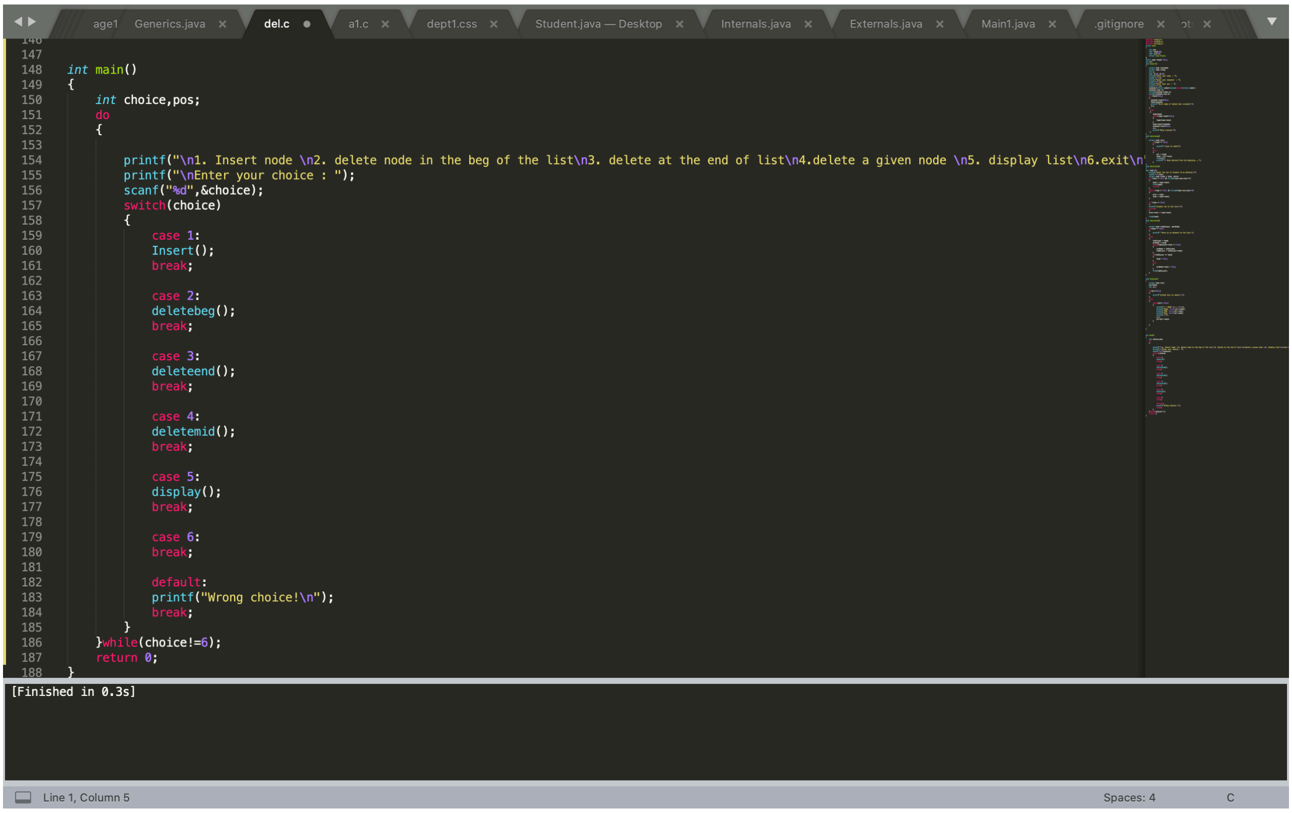
```
     else
     {
        temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->next=NULL;
        c++;
        printf("Node created\n");
     }
}
void deletebeg()
{
    struct node *ptr;
        if(head == NULL)
        {
            printf("\nList is empty");
        }
        else
        {
            ptr = head;
            head = ptr->next;
            free(ptr);
            printf("\n Node deleted from the begining ...");
        }
}
void deletemid()
{
char key[20];
    printf("enter the usn of student to be deleted\n");
    scanf("%s",key);
    struct node *temp = head, *prev;
    if (temp != NULL && strcmp(temp->usn,key)==0)
    {
        head = temp->next;
        free(temp);
        return;
    }
    while (temp != NULL && strcmp(temp->usn,key)!=0)
    {
```



```
    while (temp != NULL && strcmp(temp->usn,key)!=0)
    {
        prev = temp;
        temp = temp->next;
    }

    if (temp == NULL)
    {
    printf("student not in the list\n");
    return;
    }
    prev->next = temp->next;

    free(temp);
}
void deleteend()
{

    struct node *toDelLast, *preNode;
    if(head == NULL)
    {
        printf(" There is no element in the list.");
    }
    else
    {
        toDelLast = head;
        preNode = head;
        while(toDelLast->next != NULL)
        {
            preNode = toDelLast;
            toDelLast = toDelLast->next;
        }
        if(toDelLast == head)
        {
            head = NULL;
        }
        else
        {
            preNode->next = NULL;
        }
        free(toDelLast);
    }
```

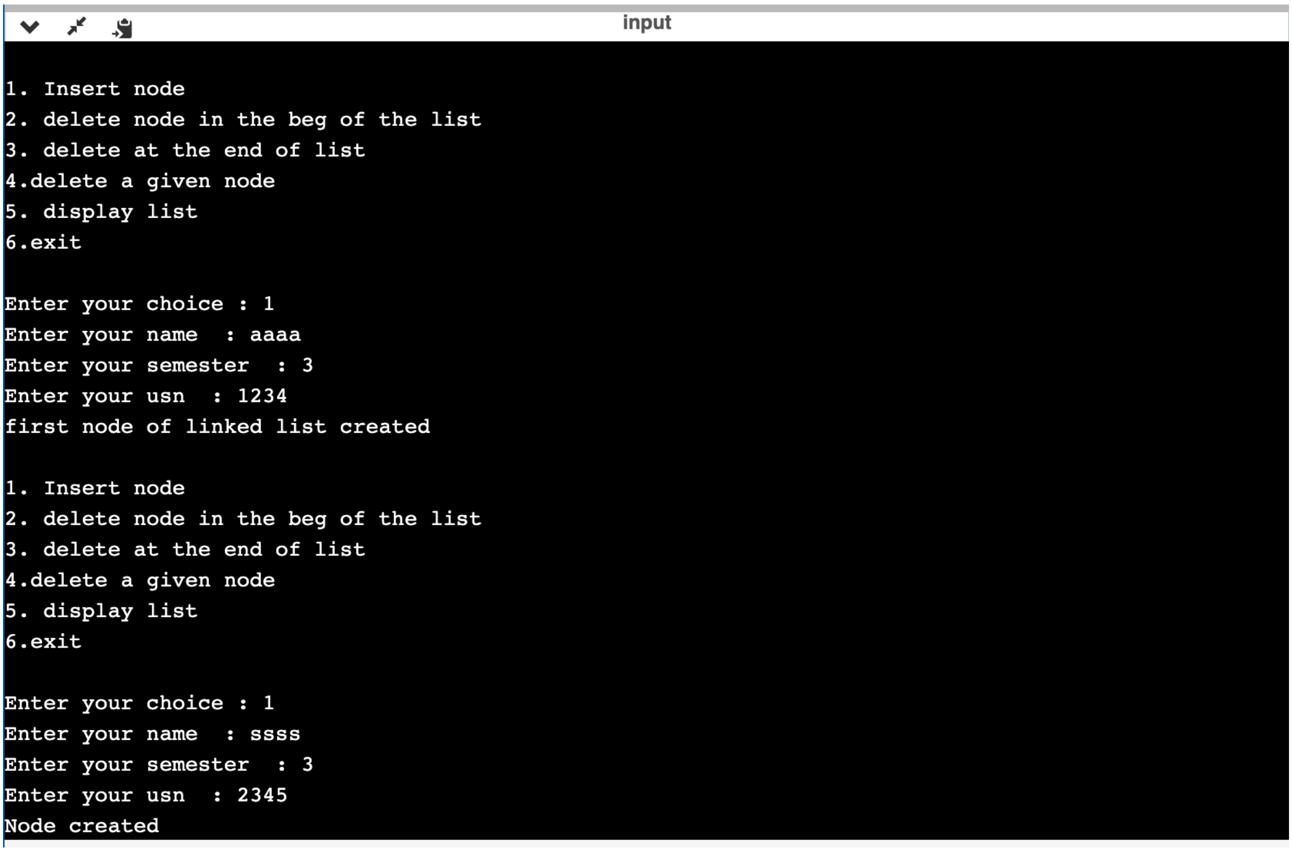


```c

void display()
{
    struct node *ptr;
    ptr=head;
    int i=1;

    if(ptr==NULL)
    {
        printf("Linked list is empty!\n");
    }
    else
    {
        while(ptr!= NULL)
        {
            printf("----NODE %d----\n",i);
            printf("Name: %s\n",ptr->name);
            printf("USN: %s\n",ptr->usn);
            printf("Sem: %d\n",ptr->sem);
            printf("\n");
            i++;
            ptr=ptr->next;
        }

    }

}

```

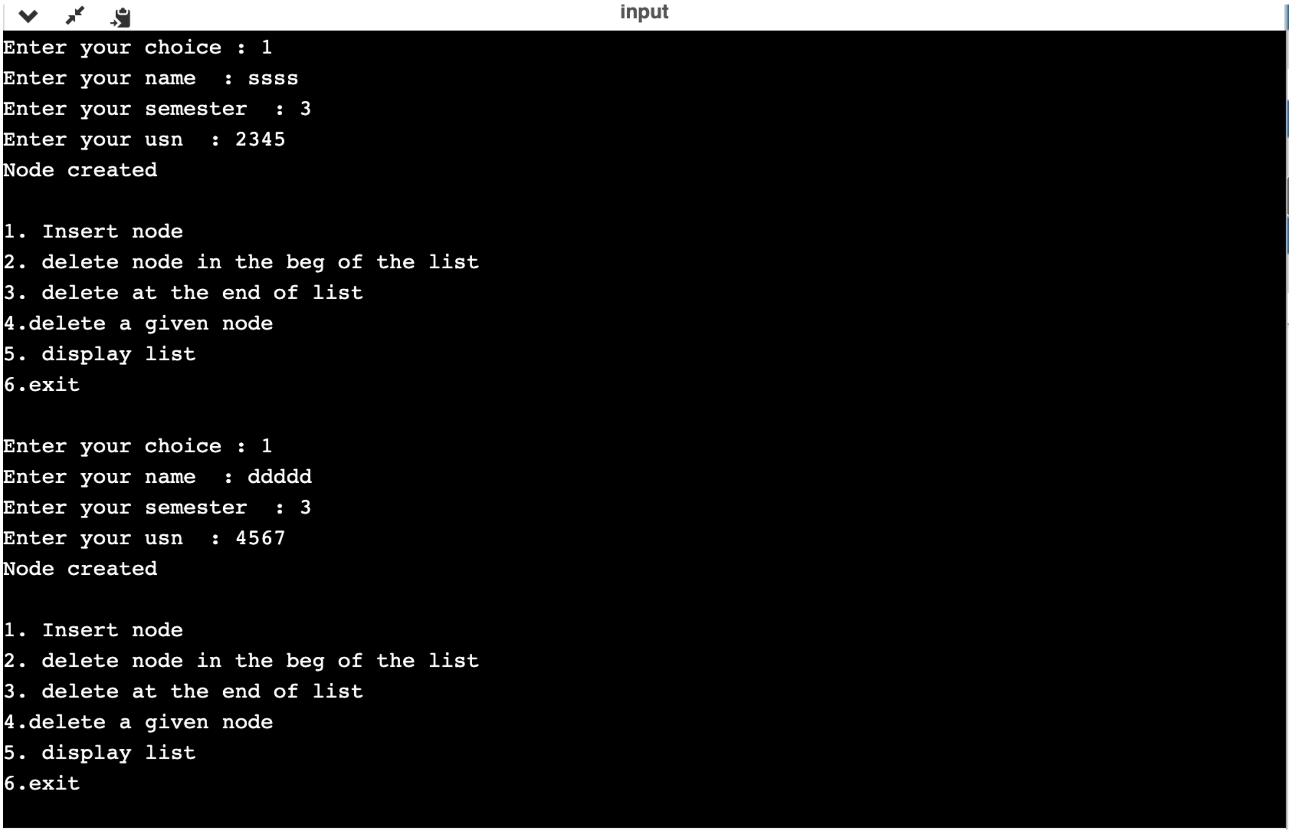


```c

int main()
{
    int choice,pos;
    do
    {

        printf("\n1. Insert node \n2. delete node in the beg of the list\n3. delete at the end of list\n4.delete a given node \n5. display list\n6.exit\n
        printf("\nEnter your choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            Insert();
            break;

            case 2:
            deletebeg();
            break;

            case 3:
            deleteend();
            break;

            case 4:
            deletemid();
            break;

            case 5:
            display();
            break;

            case 6:
            break;

            default:
            printf("Wrong choice!\n");
            break;
        }
    }while(choice!=6);
    return 0;
}
```

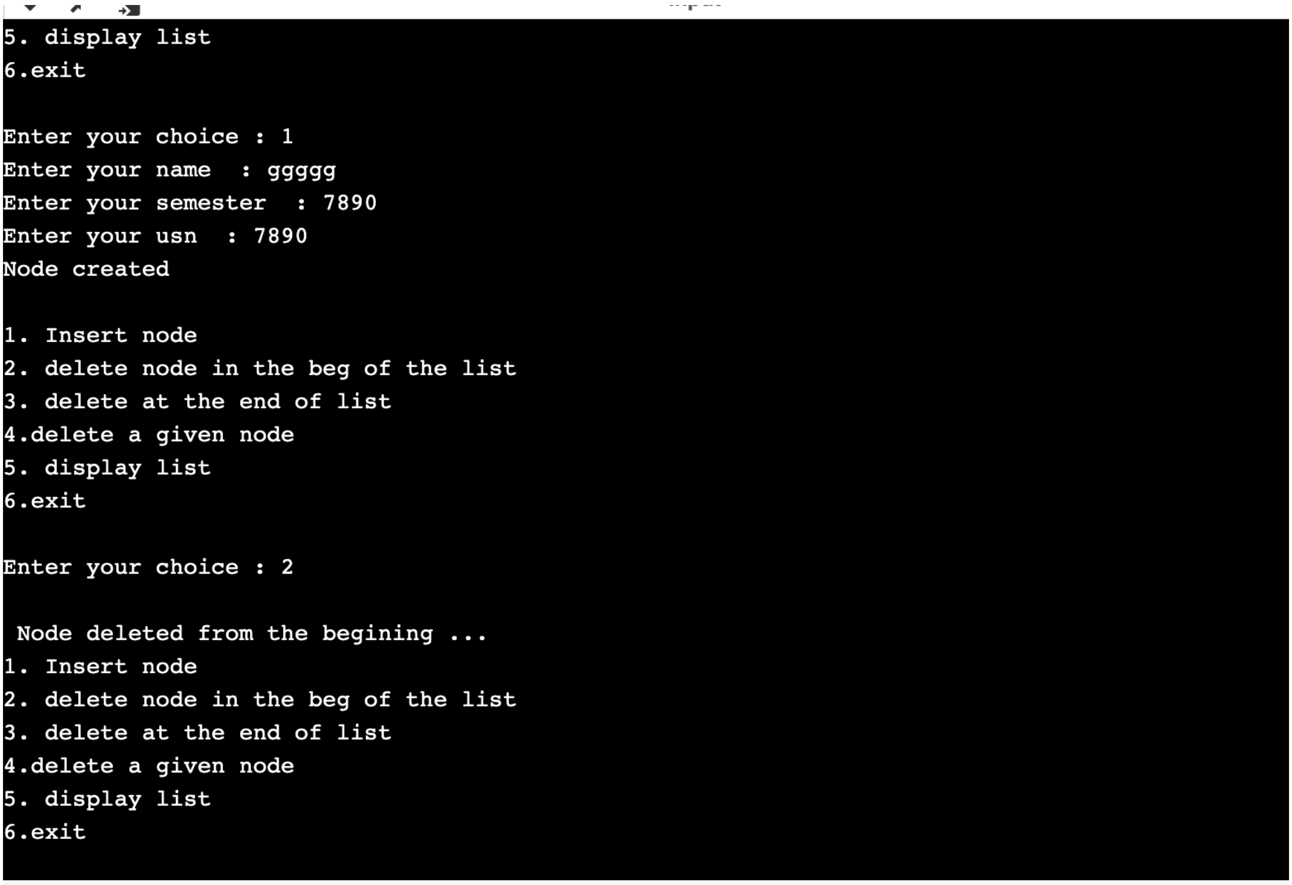
```
[Finished in 0.3s]
```



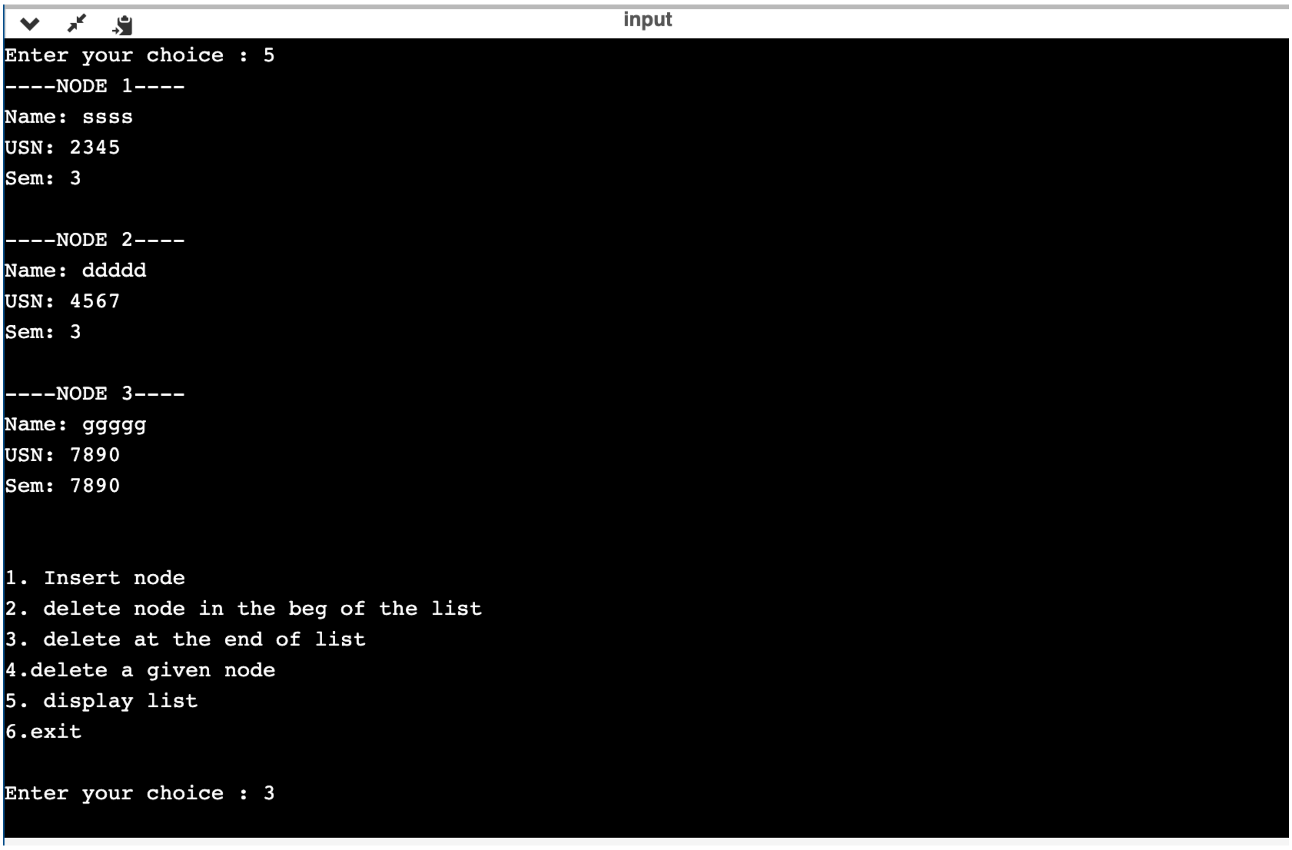
```
1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 1
Enter your name  : aaaa
Enter your semester  : 3
Enter your usn  : 1234
first node of linked list created

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 1
Enter your name  : ssss
Enter your semester  : 3
Enter your usn  : 2345
Node created
```



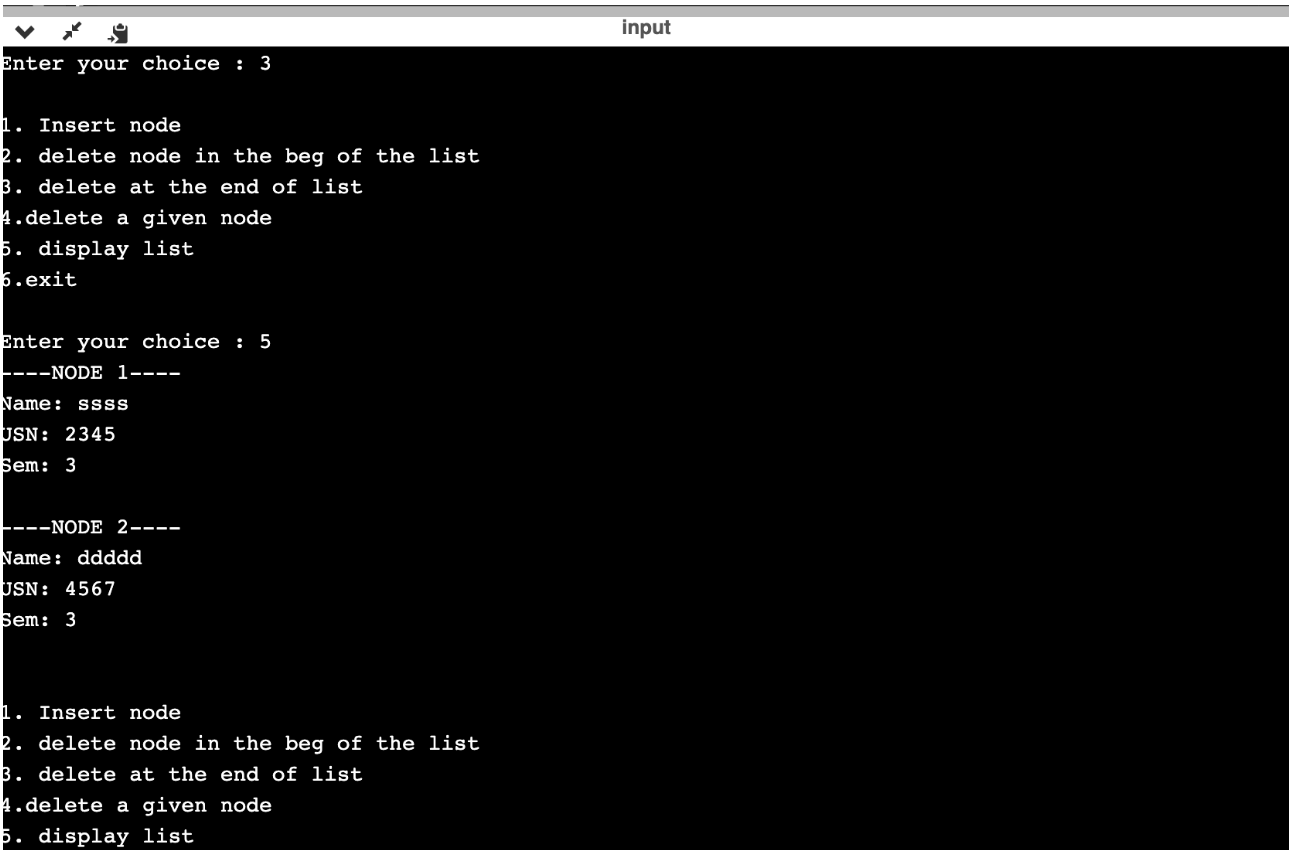
```
Enter your choice : 1
Enter your name  : ssss
Enter your semester  : 3
Enter your usn  : 2345
Node created

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 1
Enter your name  : ddddd
Enter your semester  : 3
Enter your usn  : 4567
Node created

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit
```



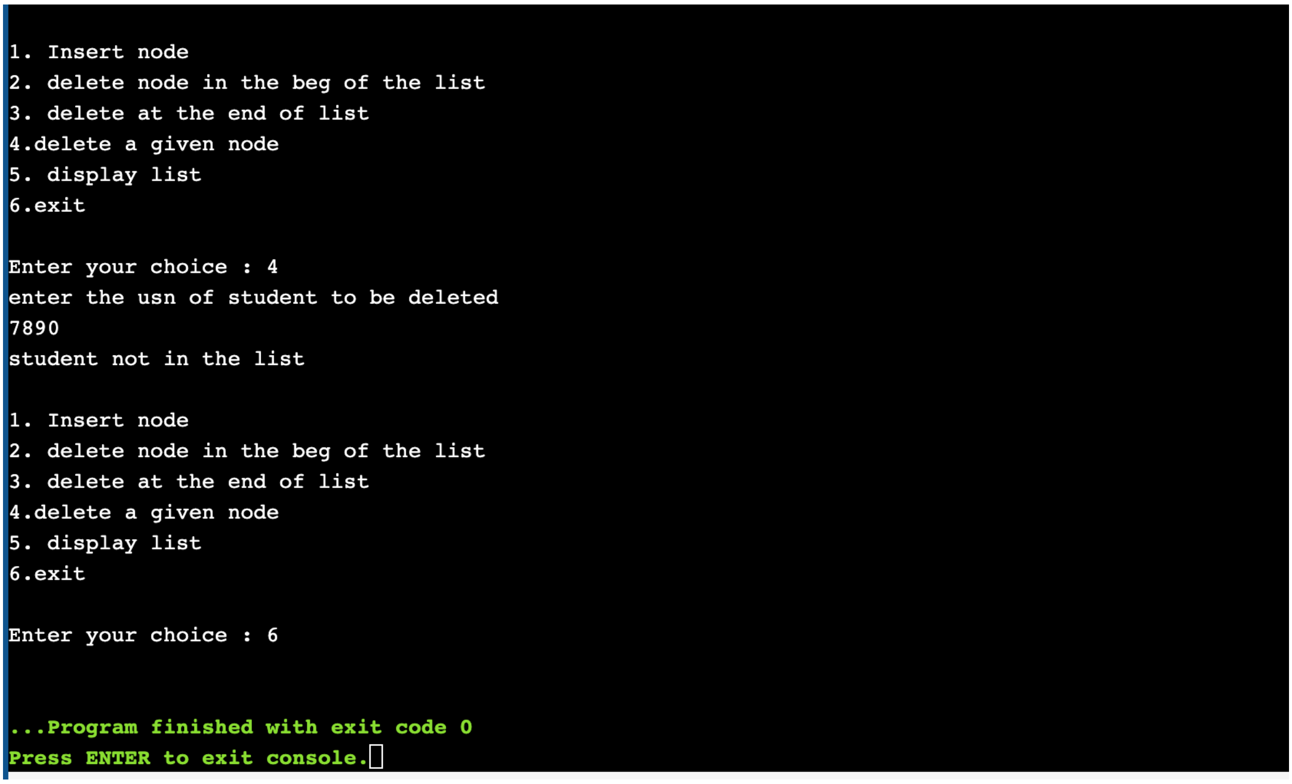
```
5. display list
6.exit

Enter your choice : 1
Enter your name  : ggggg
Enter your semester  : 7890
Enter your usn  : 7890
Node created

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 2

 Node deleted from the begining ...
1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit
```



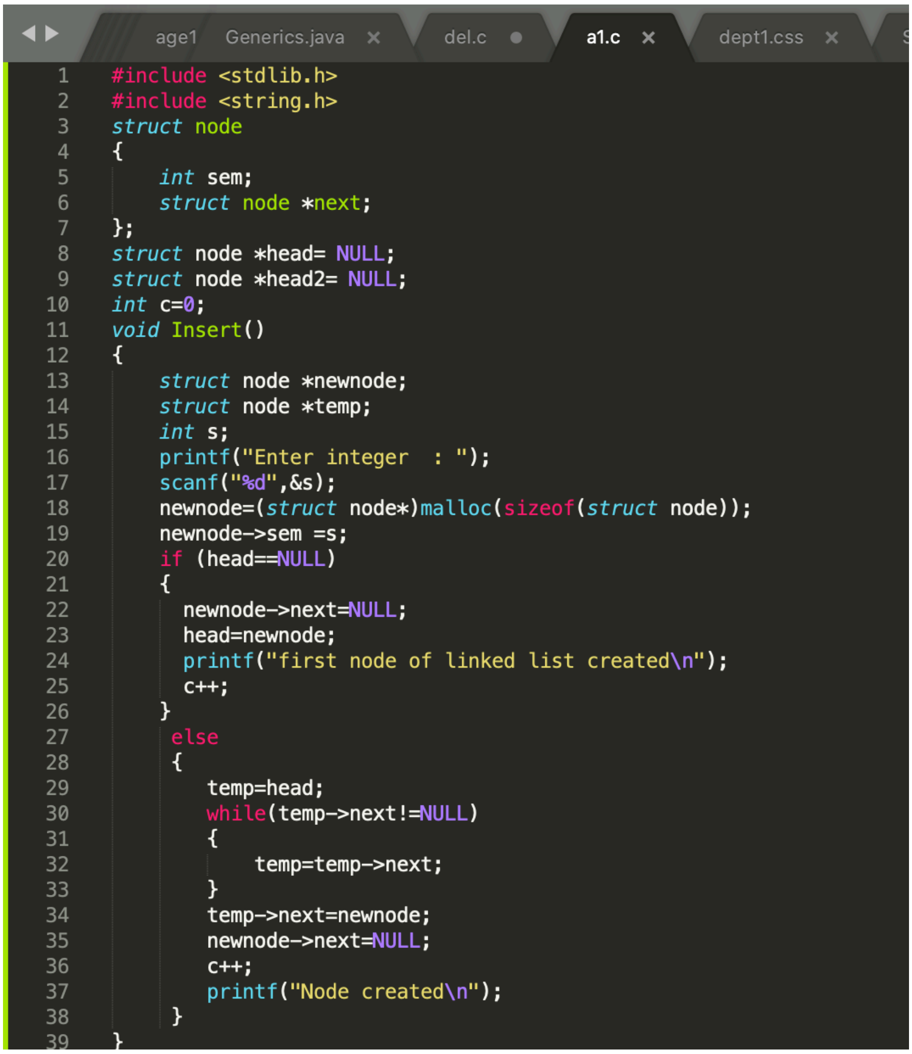
```
Enter your choice : 5
----NODE 1----
Name: ssss
USN: 2345
Sem: 3

----NODE 2----
Name: ddddd
USN: 4567
Sem: 3

----NODE 3----
Name: ggggg
USN: 7890
Sem: 7890


1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 3
```



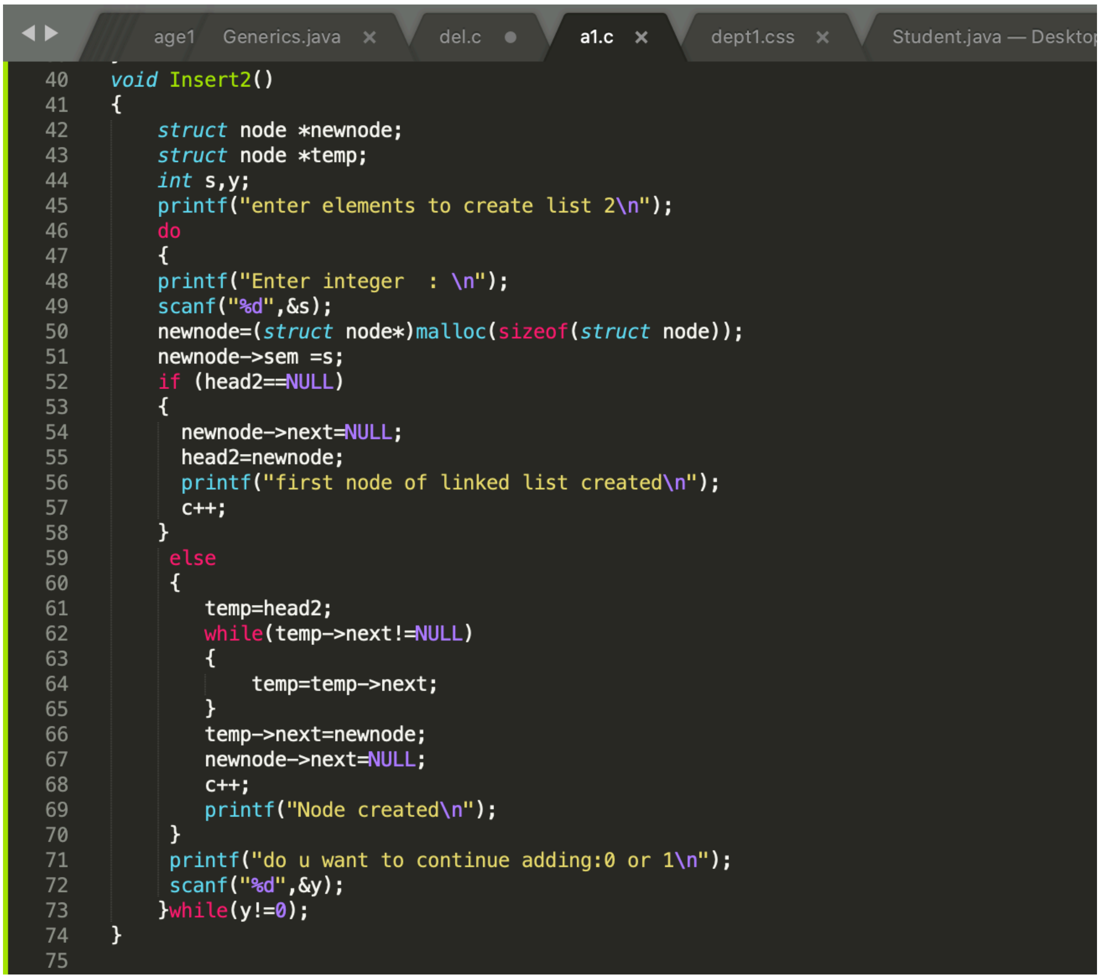
```
Enter your choice : 3

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 5
----NODE 1----
Name: ssss
USN: 2345
Sem: 3

----NODE 2----
Name: ddddd
USN: 4567
Sem: 3


1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
```



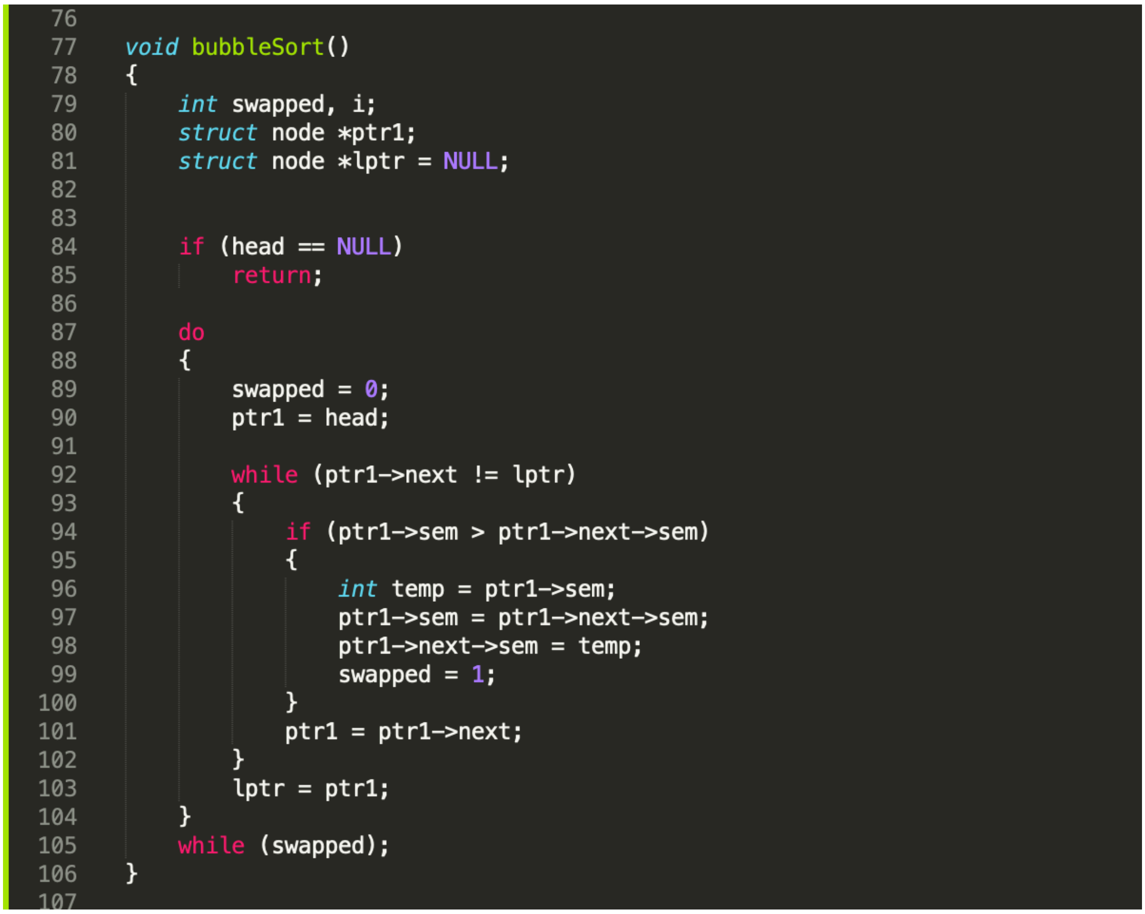
```

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 4
enter the usn of student to be deleted
7890
student not in the list

1. Insert node
2. delete node in the beg of the list
3. delete at the end of list
4.delete a given node
5. display list
6.exit

Enter your choice : 6


...Program finished with exit code 0
Press ENTER to exit console.
```



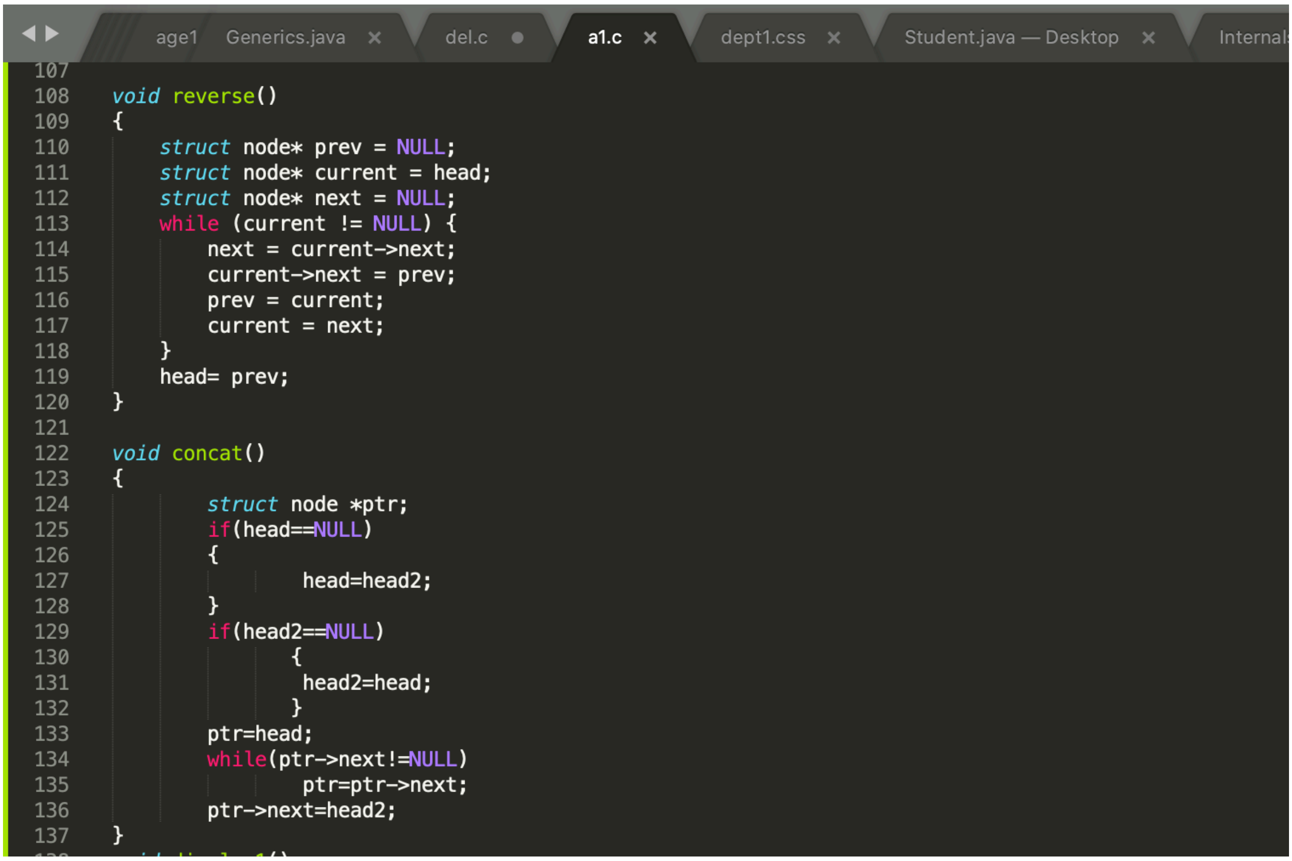
```
#include <stdlib.h>
#include <string.h>
struct node
{
    int sem;
    struct node *next;
};
struct node *head= NULL;
struct node *head2= NULL;
int c=0;
void Insert()
{
    struct node *newnode;
    struct node *temp;
    int s;
    printf("Enter integer  : ");
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->sem =s;
    if (head==NULL)
    {
      newnode->next=NULL;
      head=newnode;
      printf("first node of linked list created\n");
      c++;
    }
     else
     {
        temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->next=NULL;
        c++;
        printf("Node created\n");
     }
}
```



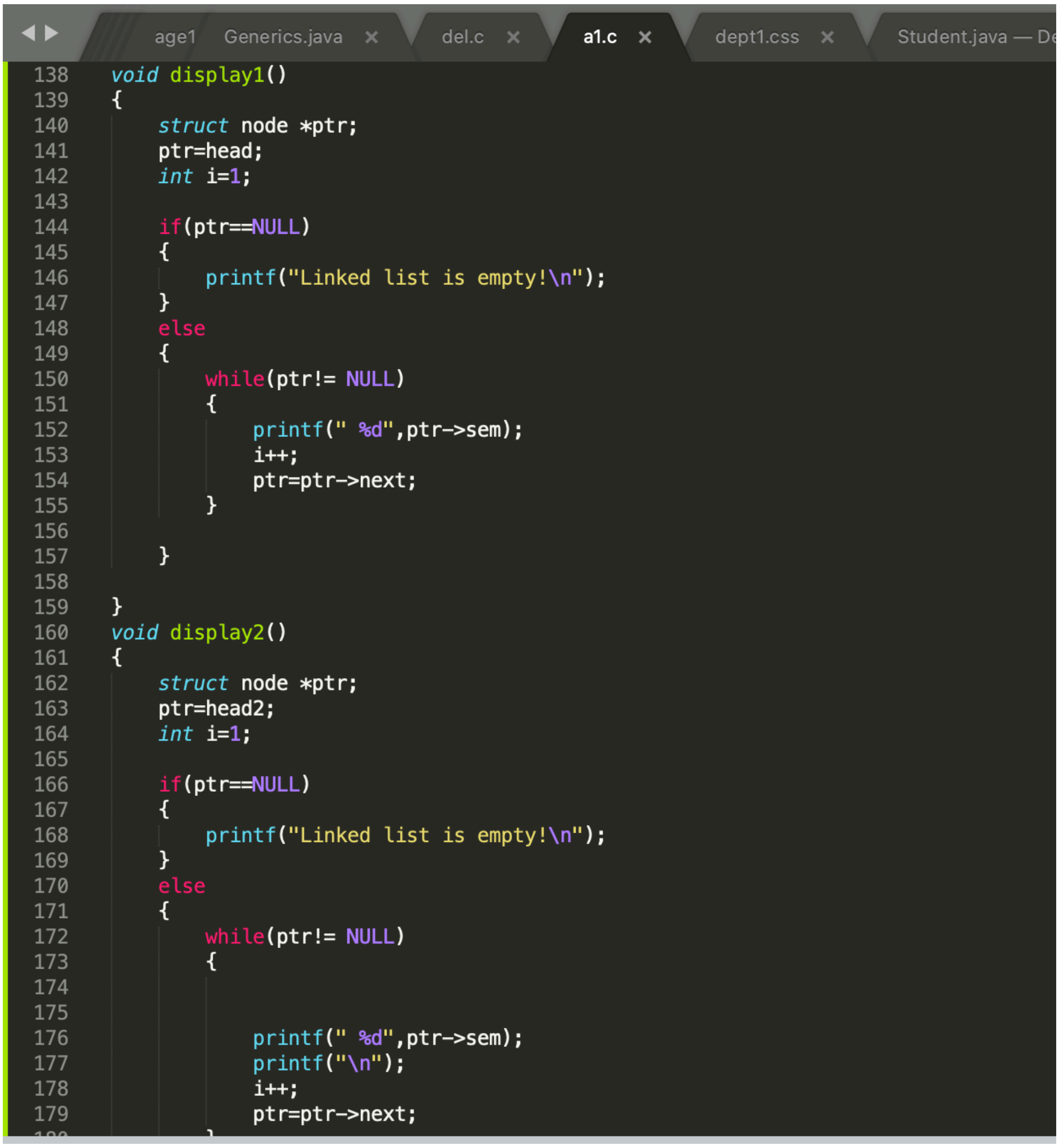
```
void Insert2()
{
    struct node *newnode;
    struct node *temp;
    int s,y;
    printf("enter elements to create list 2\n");
    do
    {
    printf("Enter integer  : \n");
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->sem =s;
    if (head2==NULL)
    {
      newnode->next=NULL;
      head2=newnode;
      printf("first node of linked list created\n");
      c++;
    }
     else
     {
        temp=head2;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->next=NULL;
        c++;
        printf("Node created\n");
     }
     printf("do u want to continue adding:0 or 1\n");
     scanf("%d",&y);
    }while(y!=0);
}
```



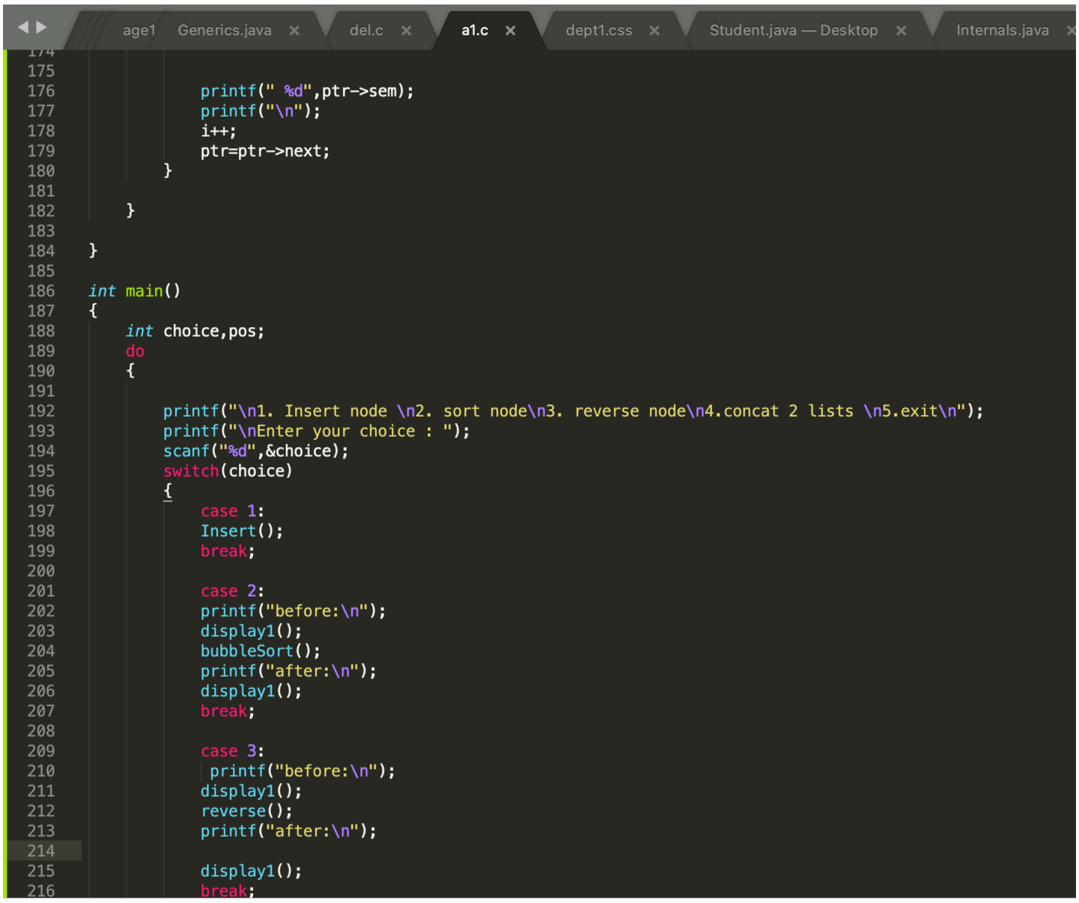
```c
void bubbleSort()
{
    int swapped, i;
    struct node *ptr1;
    struct node *lptr = NULL;


    if (head == NULL)
        return;

    do
    {
        swapped = 0;
        ptr1 = head;

        while (ptr1->next != lptr)
        {
            if (ptr1->sem > ptr1->next->sem)
            {
                int temp = ptr1->sem;
                ptr1->sem = ptr1->next->sem;
                ptr1->next->sem = temp;
                swapped = 1;
            }
            ptr1 = ptr1->next;
        }
        lptr = ptr1;
    }
    while (swapped);
}
```



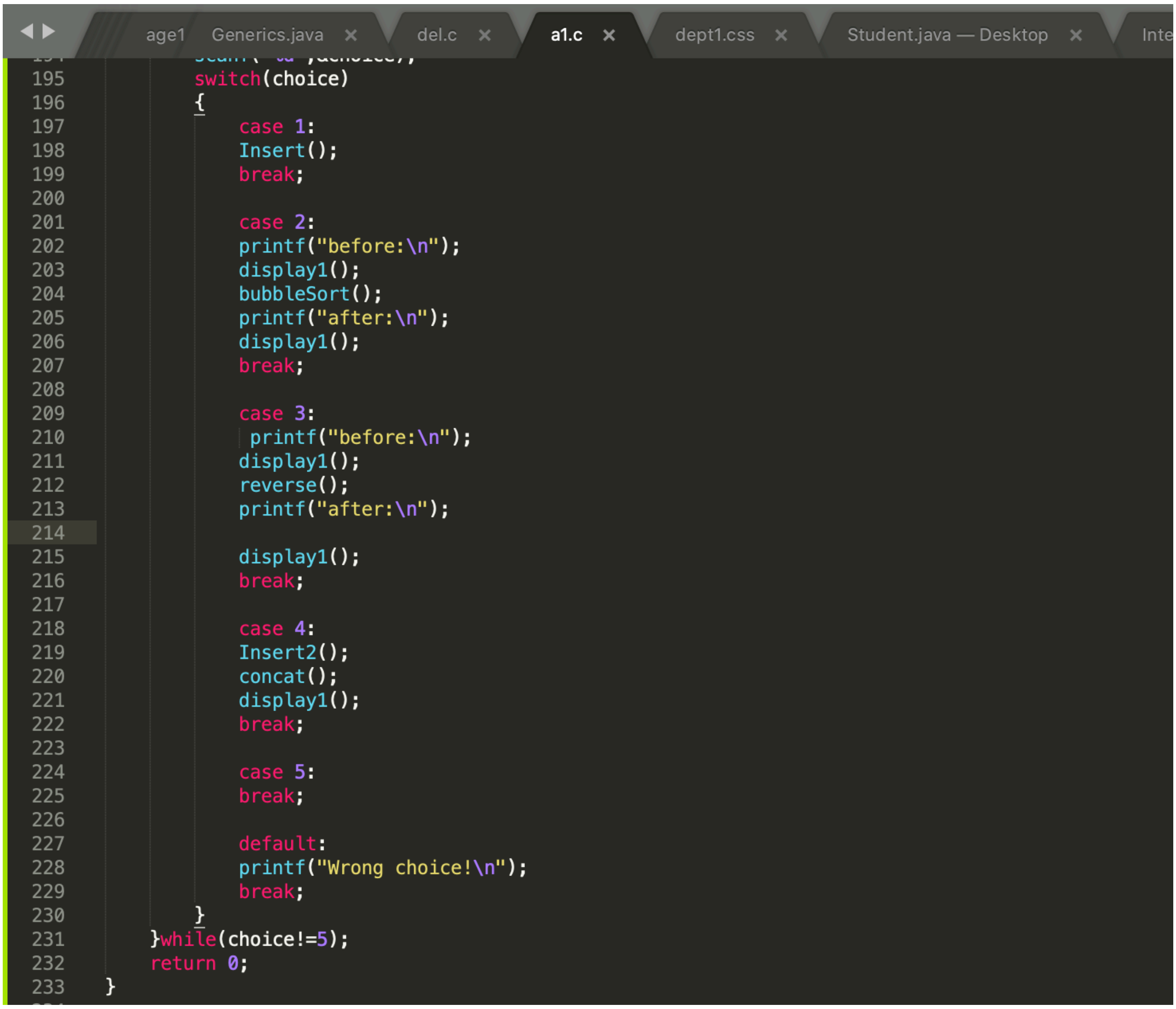
```c
void reverse()
{
    struct node* prev = NULL;
    struct node* current = head;
    struct node* next = NULL;
    while (current != NULL) {
        next = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    head= prev;
}

void concat()
{
        struct node *ptr;
        if(head==NULL)
        {
                head=head2;
        }
        if(head2==NULL)
               {
                head2=head;
               }
        ptr=head;
        while(ptr->next!=NULL)
                ptr=ptr->next;
        ptr->next=head2;
}
```



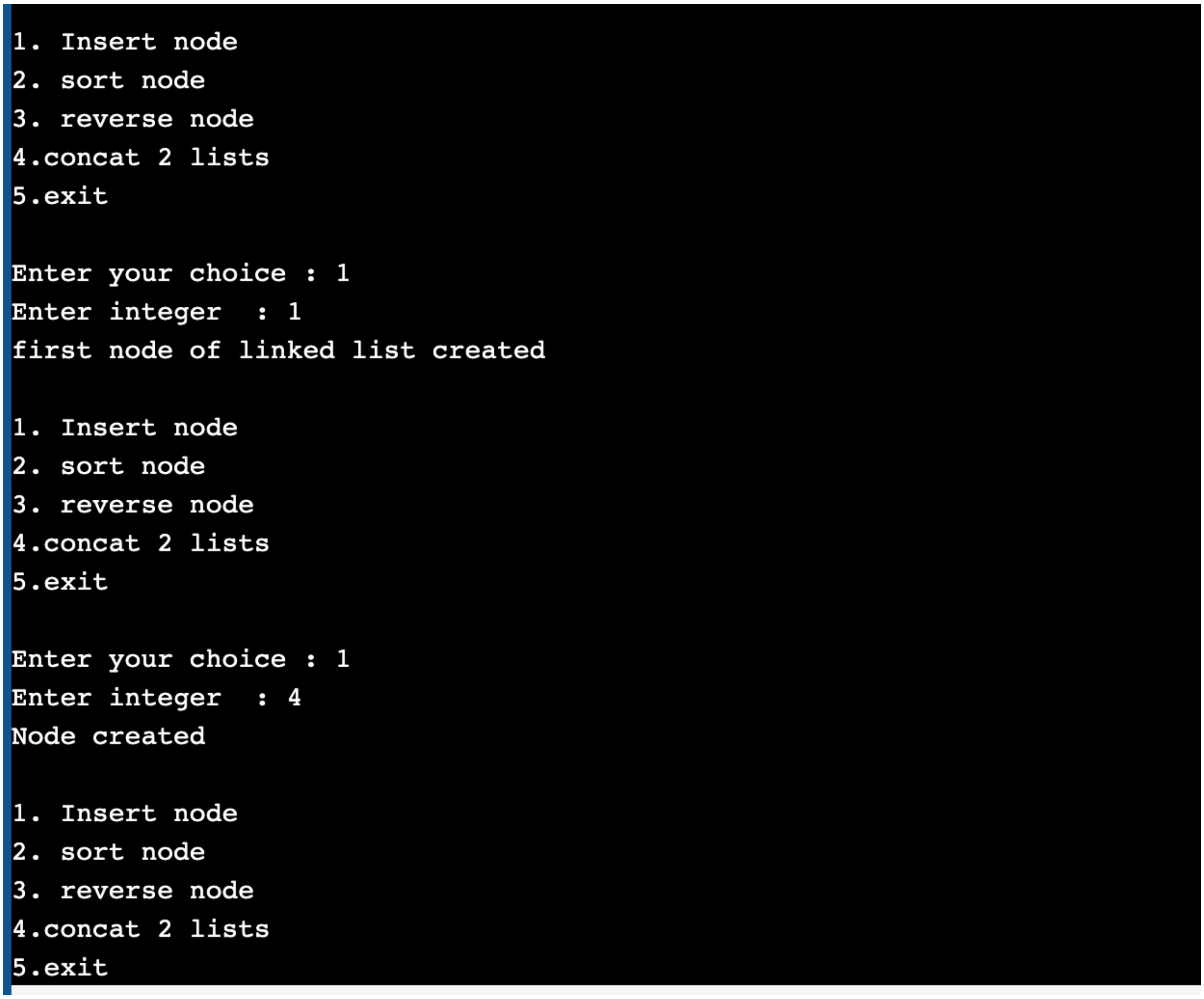
```c
void display1()
{
    struct node *ptr;
    ptr=head;
    int i=1;

    if(ptr==NULL)
    {
        printf("Linked list is empty!\n");
    }
    else
    {
        while(ptr!= NULL)
        {
            printf(" %d",ptr->sem);
            i++;
            ptr=ptr->next;
        }

    }

}
void display2()
{
    struct node *ptr;
    ptr=head2;
    int i=1;

    if(ptr==NULL)
    {
        printf("Linked list is empty!\n");
    }
    else
    {
        while(ptr!= NULL)
        {


            printf(" %d",ptr->sem);
            printf("\n");
            i++;
            ptr=ptr->next;
        }
```



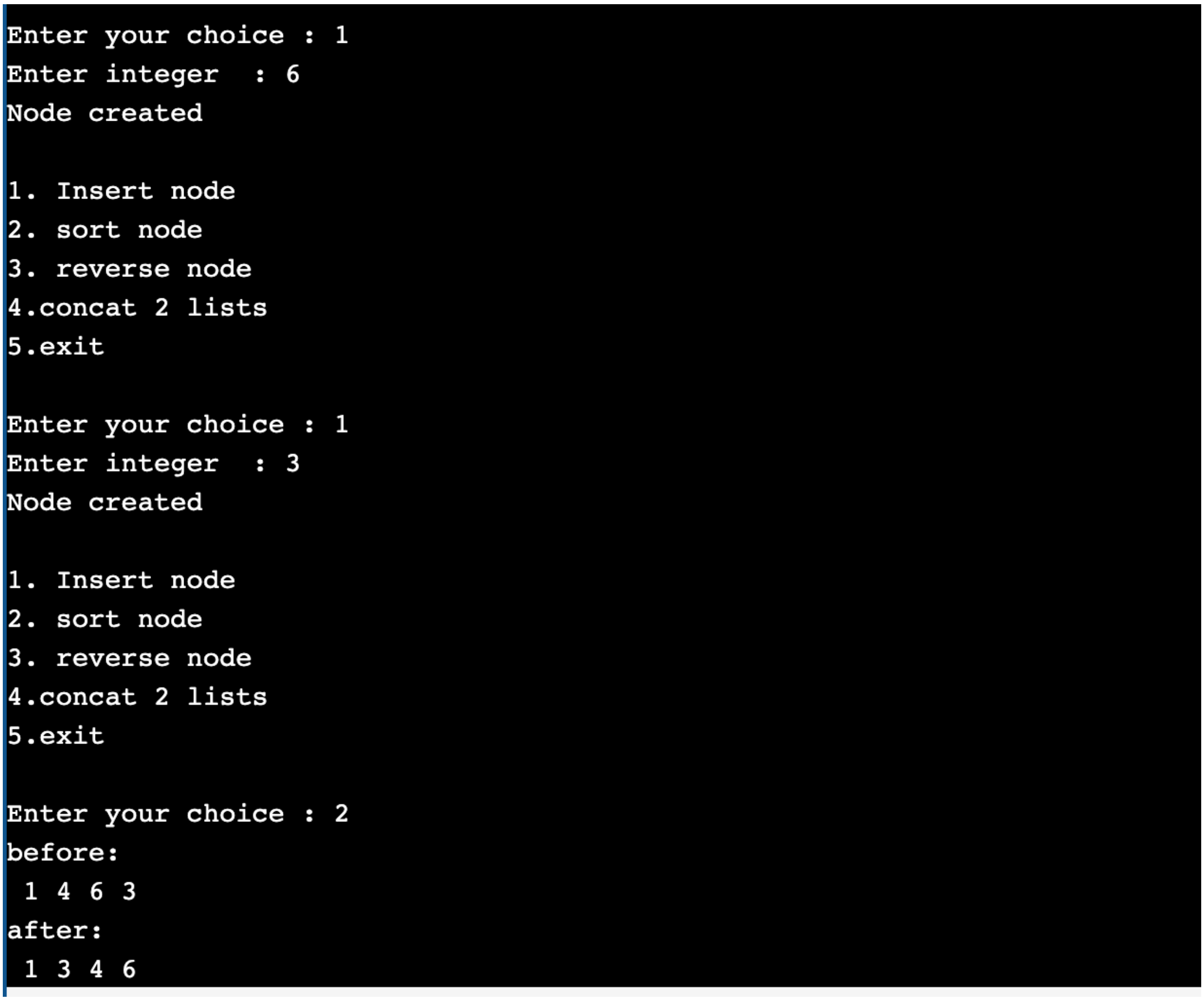
```c

            printf(" %d",ptr->sem);
            printf("\n");
            i++;
            ptr=ptr->next;
        }

    }

}

int main()
{
    int choice,pos;
    do
    {

        printf("\n1. Insert node \n2. sort node\n3. reverse node\n4.concat 2 lists \n5.exit\n");
        printf("\nEnter your choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            Insert();
            break;

            case 2:
            printf("before:\n");
            display1();
            bubbleSort();
            printf("after:\n");
            display1();
            break;

            case 3:
             printf("before:\n");
            display1();
            reverse();
            printf("after:\n");

            display1();
            break;
```



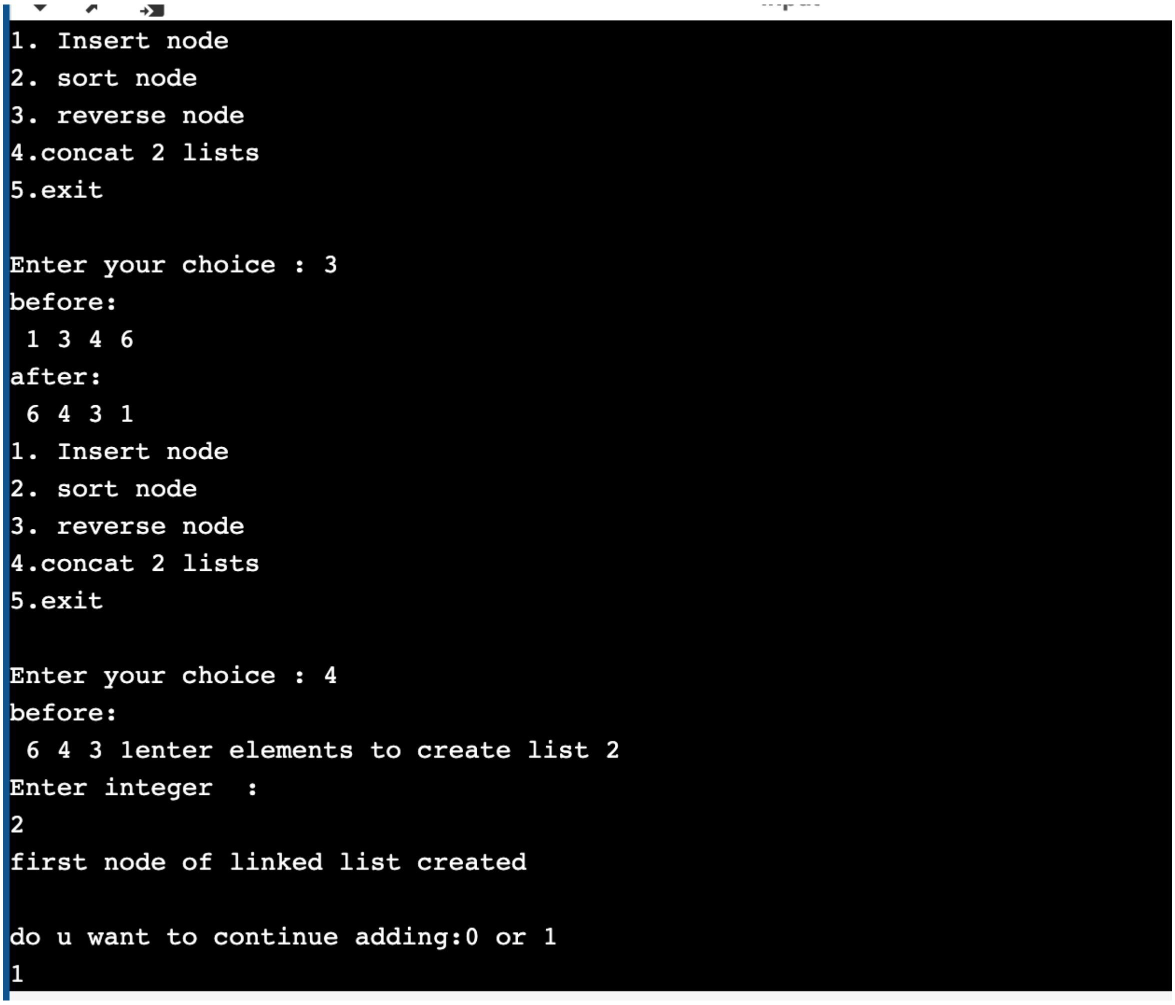
```c
        switch(choice)
        {
            case 1:
            Insert();
            break;

            case 2:
            printf("before:\n");
            display1();
            bubbleSort();
            printf("after:\n");
            display1();
            break;

            case 3:
             printf("before:\n");
            display1();
            reverse();
            printf("after:\n");

            display1();
            break;

            case 4:
            Insert2();
            concat();
            display1();
            break;

            case 5:
            break;

            default:
            printf("Wrong choice!\n");
            break;
        }
    }while(choice!=5);
    return 0;
}
```



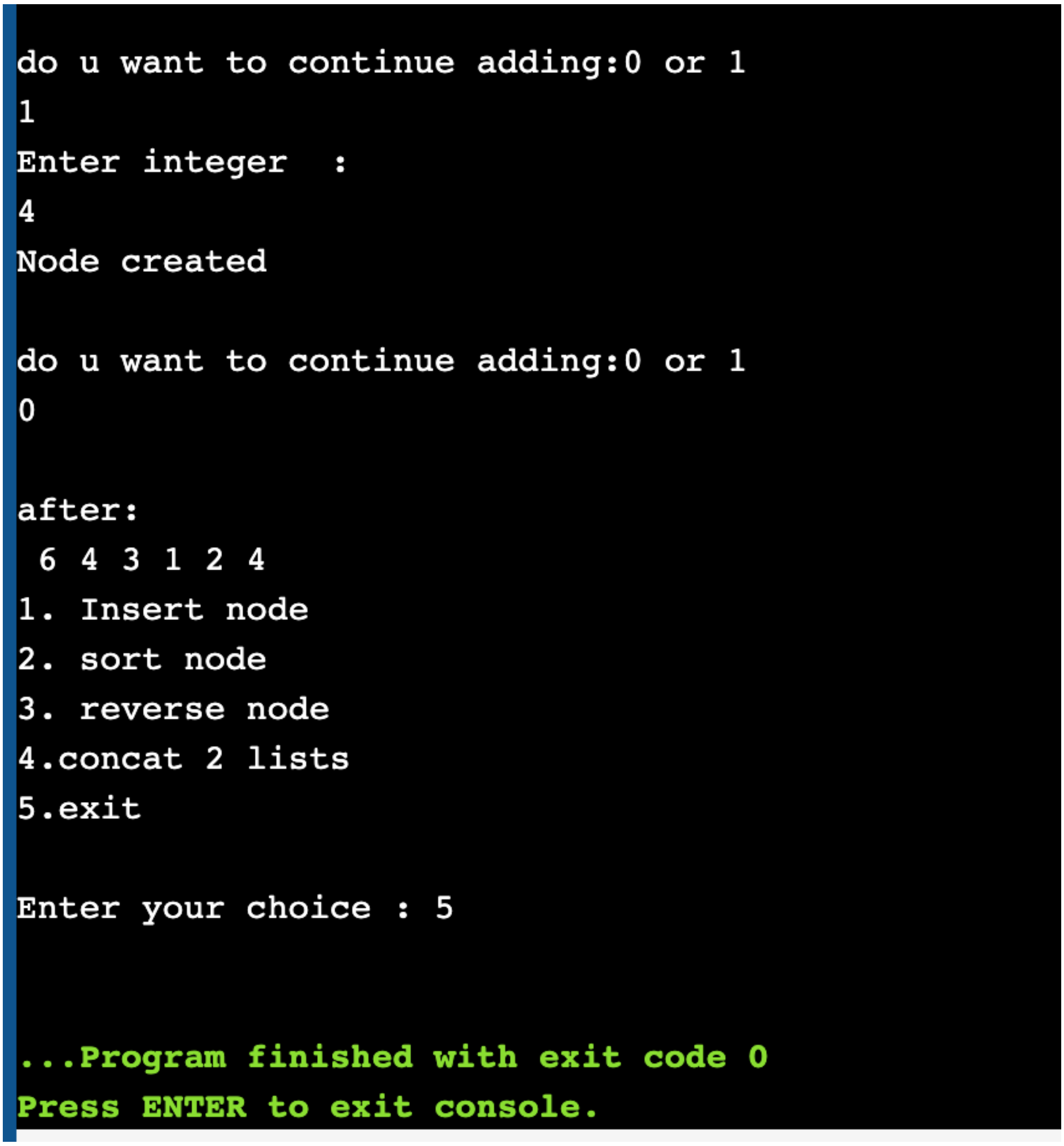
```
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 1
first node of linked list created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 4
Node created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit
```



```
Enter your choice : 1
Enter integer  : 6
Node created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 3
Node created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 2
before:
 1 4 6 3
after:
 1 3 4 6
```



```
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 3
before:
 1 3 4 6
after:
 6 4 3 1
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 4
before:
 6 4 3 1enter elements to create list 2
Enter integer  :
2
first node of linked list created

do u want to continue adding:0 or 1
1
```



```
do u want to continue adding:0 or 1
1
Enter integer  :
4
Node created

do u want to continue adding:0 or 1
0

after:
 6 4 3 1 2 4
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 5



...Program finished with exit code 0
Press ENTER to exit console.
```