CSE2003 Data Structures and Algorithms

**Assessment 2**

**Rishav Raj Jain**

**17BCE0042**

**L29-L30**

1. **Linked List**

**CODE**

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>

void newnode();

void display();

void insertion\_b();

void insertion\_e();

void insertion\_m();

void insertion\_d();

void delet\_b();

void delet\_e();

void delet\_m();

typedef struct node

{

int data;

struct node \*next;

}NODE;

int count = 0;

int choice = 1,n=0,x=0;

NODE \*head, \*first=0, \*temp = 0,\*temp2 = 0,\*temp3 = 0;

void main()

{

printf(" 17BCE0042\n");

char option='T';

while(option=='T')

{

printf("\n");

printf("Enter command\n");

printf("1. New node\n");

printf("2. Display\n");

printf("3. Insert Beginning\n");

printf("4. Insert End\n");

printf("5. Insert Middle\n");

printf("6. Insert By Data\n");

printf("7. Delete Beginning\n");

printf("8. Delete End\n");

printf("9. Delete Middle\n");

printf("10. Exit\n\n");

scanf("%d",&n);

switch(n)

{

case 1:

newnode();

break;

case 2:

display();

break;

case 3:

insertion\_b();

break;

case 4:

insertion\_e();

break;

case 5:

insertion\_m();

break;

case 6:

insertion\_d();

break;

case 7:

delet\_b();

break;

case 8:

delet\_e();

break;

case 9:

delet\_m();

break;

case 10:

printf(" Thankyou!!\n");

exit(0);

}

}

}

void newnode()

{

while (choice)

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

if (first != 0)

{

head->next = temp;

head = temp;

}

else

{

first = head = temp;

}

fflush(stdin);

printf("Do you want to continue(Type 0 or 1)?\n");

scanf("%d", &choice);

}

head->next = 0;

display();

}

void display()

{

count=0;

temp = first;

printf("\n status of the linked list is\n");

while (temp != 0)

{

printf("%d=>", temp->data);

count++;

temp = temp -> next;

}

printf("NULL\n");

printf("No. of nodes in the list = %d\n", count);

}

void insertion\_b()

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

temp->next=first;

first = temp;

fflush(stdin);

printf("\nDone\n");

display();

}

void insertion\_e()

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

head->next=temp;

head = temp;

head->next = 0;

fflush(stdin);

printf("\nDone\n");

display();

}

void insertion\_m()

{

count=0;

temp = first;

while (temp != 0)

{

count++;

temp = temp -> next;

}

if(count%2==0)

{

x=count/2;

temp = first;

while(x-1>0)

{

temp = temp -> next;

x--;

}

}

else

{

x=(count+1)/2;

temp = first;

while(x-1>0)

{

temp = temp -> next;

x--;

}

}

temp3=temp->next;

temp2 = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp2-> data);

temp->next=temp2;

temp2->next=temp3;

fflush(stdin);

printf("\nDone\n");

display();

}

void insertion\_d()

{

temp = first;

temp2 = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp2-> data);

while (temp2->data > temp->next->data)

{

temp = temp -> next;

}

temp3=temp->next;

temp->next=temp2;

temp2->next=temp3;

fflush(stdin);

printf("\nDone\n");

display();

}

void delet\_b()

{

temp=first;

first = temp->next;

fflush(stdin);

printf("\nDone\n");

display();

}

void delet\_e()

{

temp=first;

while (temp->next->next!=0)

{

/\*printf("%d=>", temp->data);\*/

count++;

temp = temp -> next;

}

temp->next=0;

fflush(stdin);

printf("\nDone\n");

display();

}

void delet\_m()

{

count=0;

temp = first;

while (temp != 0)

{

count++;

temp = temp -> next;

}

if(count%2==0)

{

x=count/2;

temp = first;

while(x-1>0)

{

temp = temp -> next;

x--;

}

}

else

{

x=(count+1)/2;

temp = first;

while(x-2>0)

{

temp = temp -> next;

x--;

}

}

temp3=temp->next;

temp->next=temp3->next;

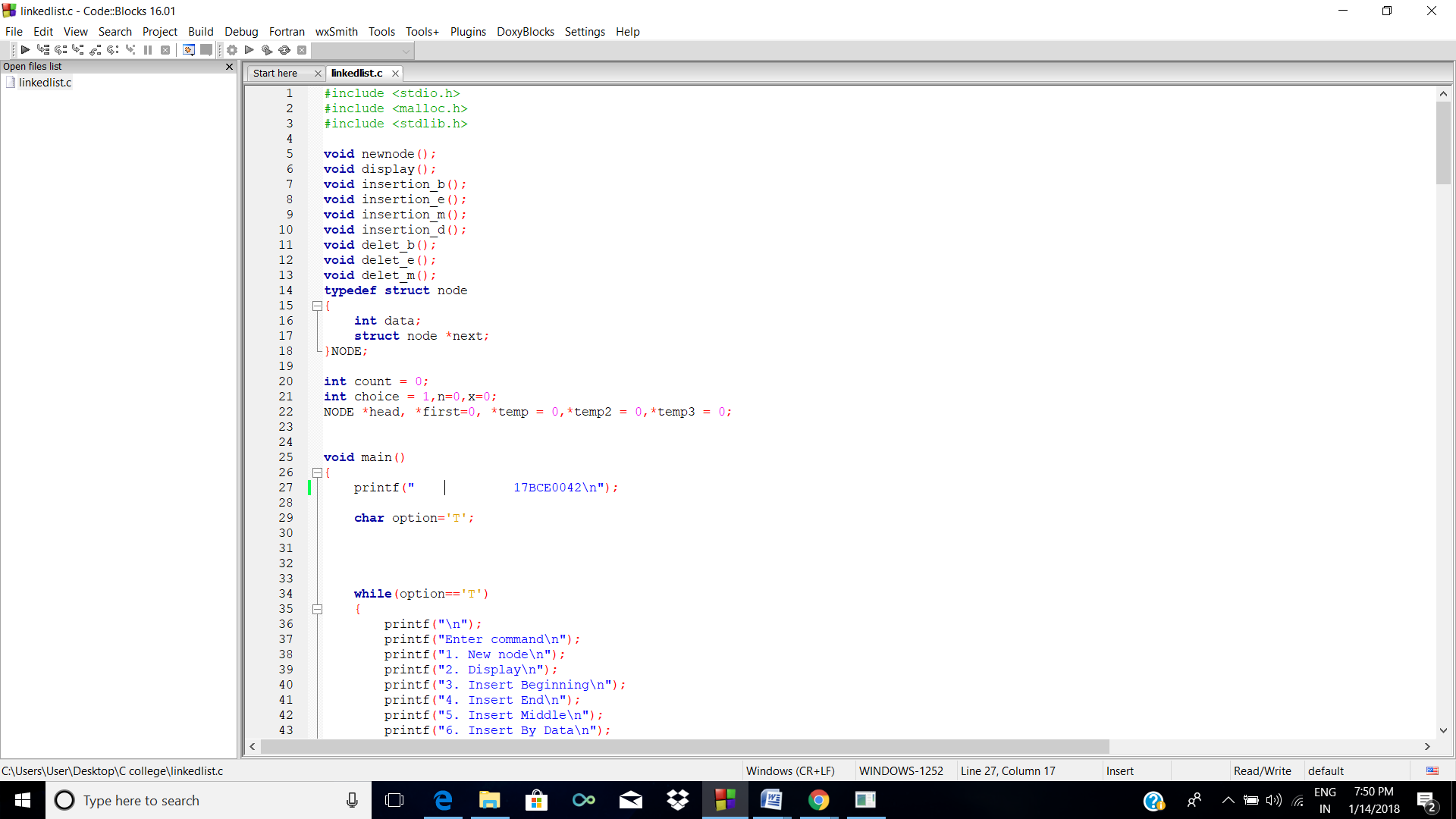
fflush(stdin);

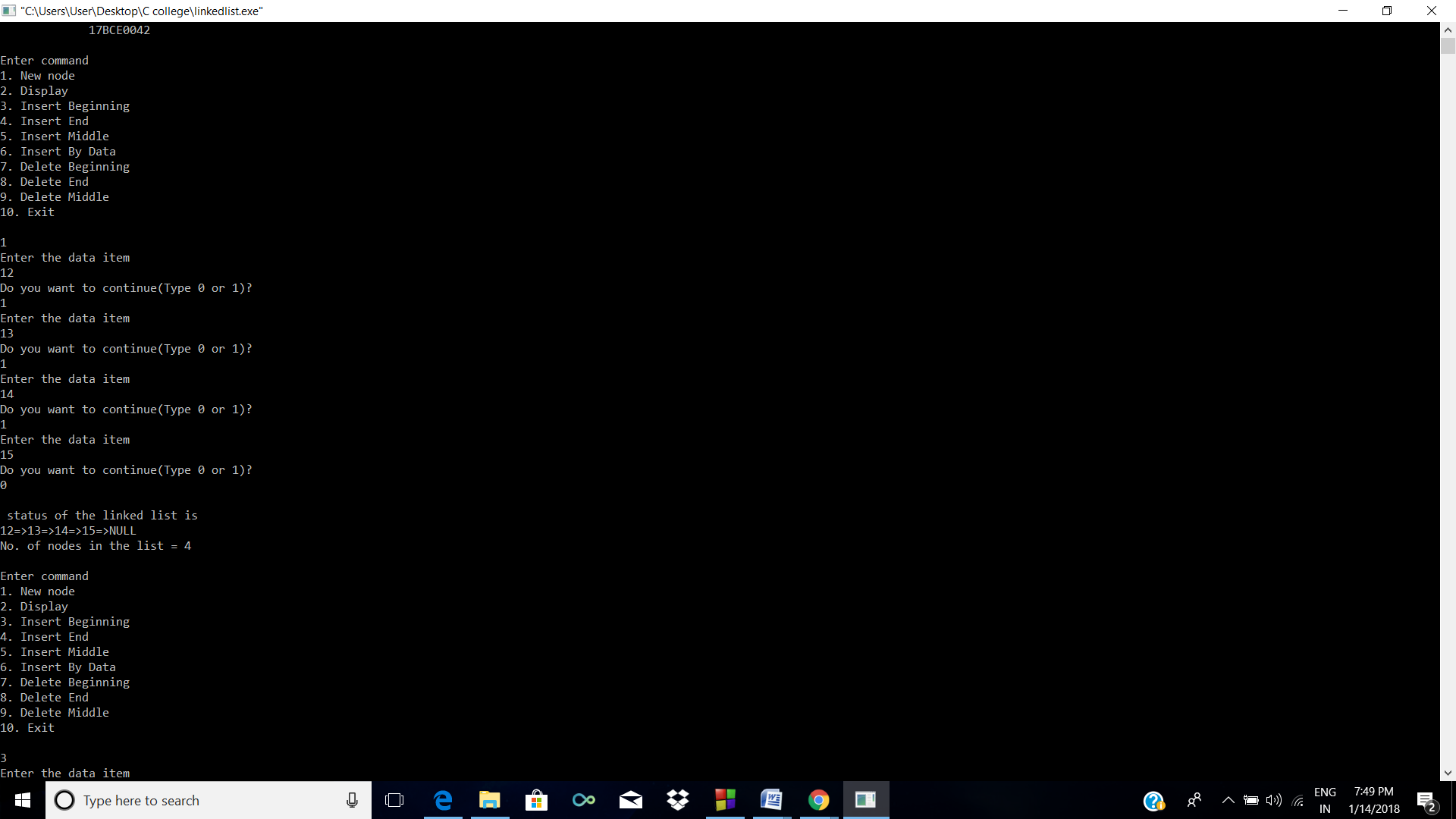
printf("\nDone\n");

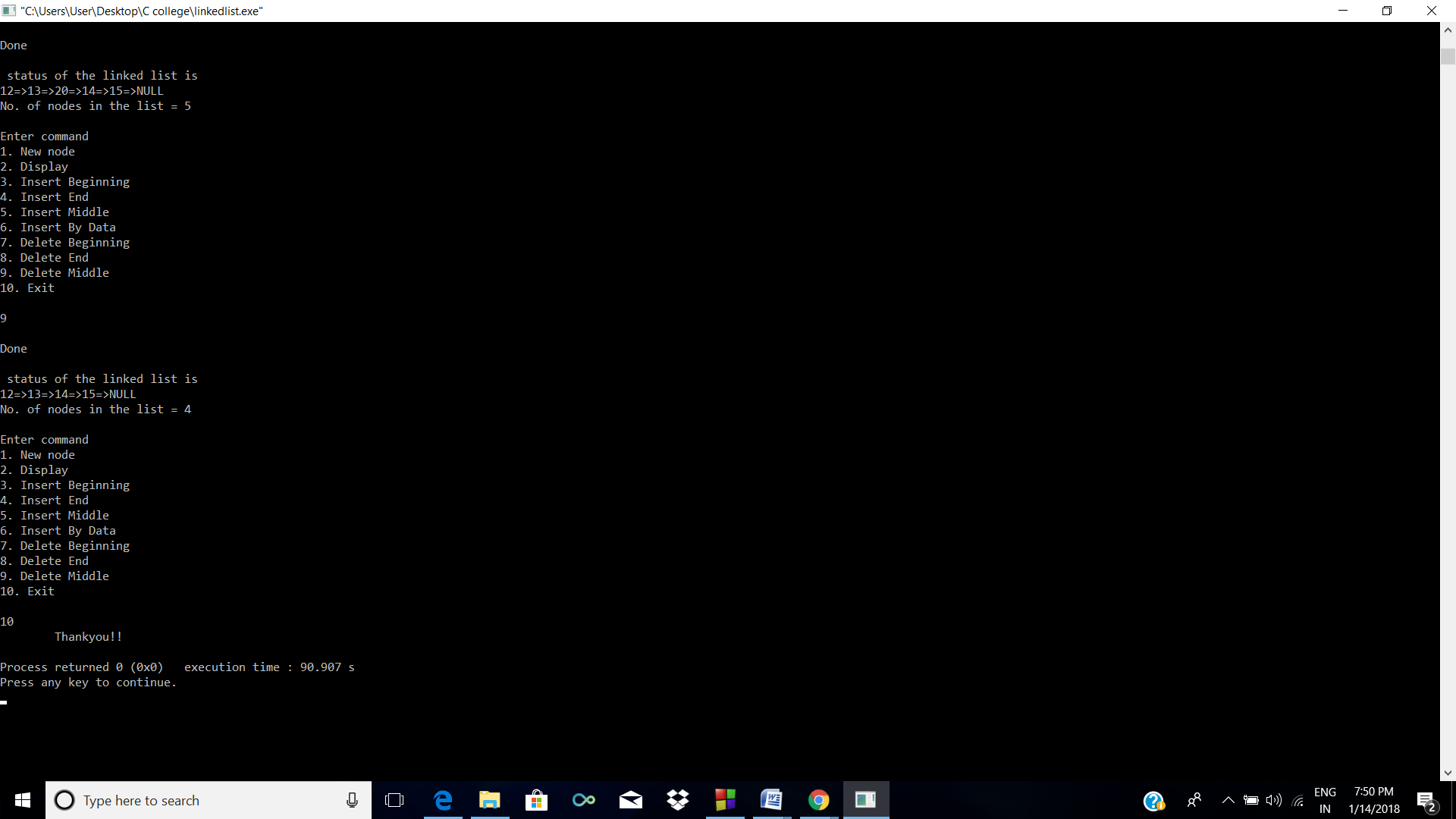
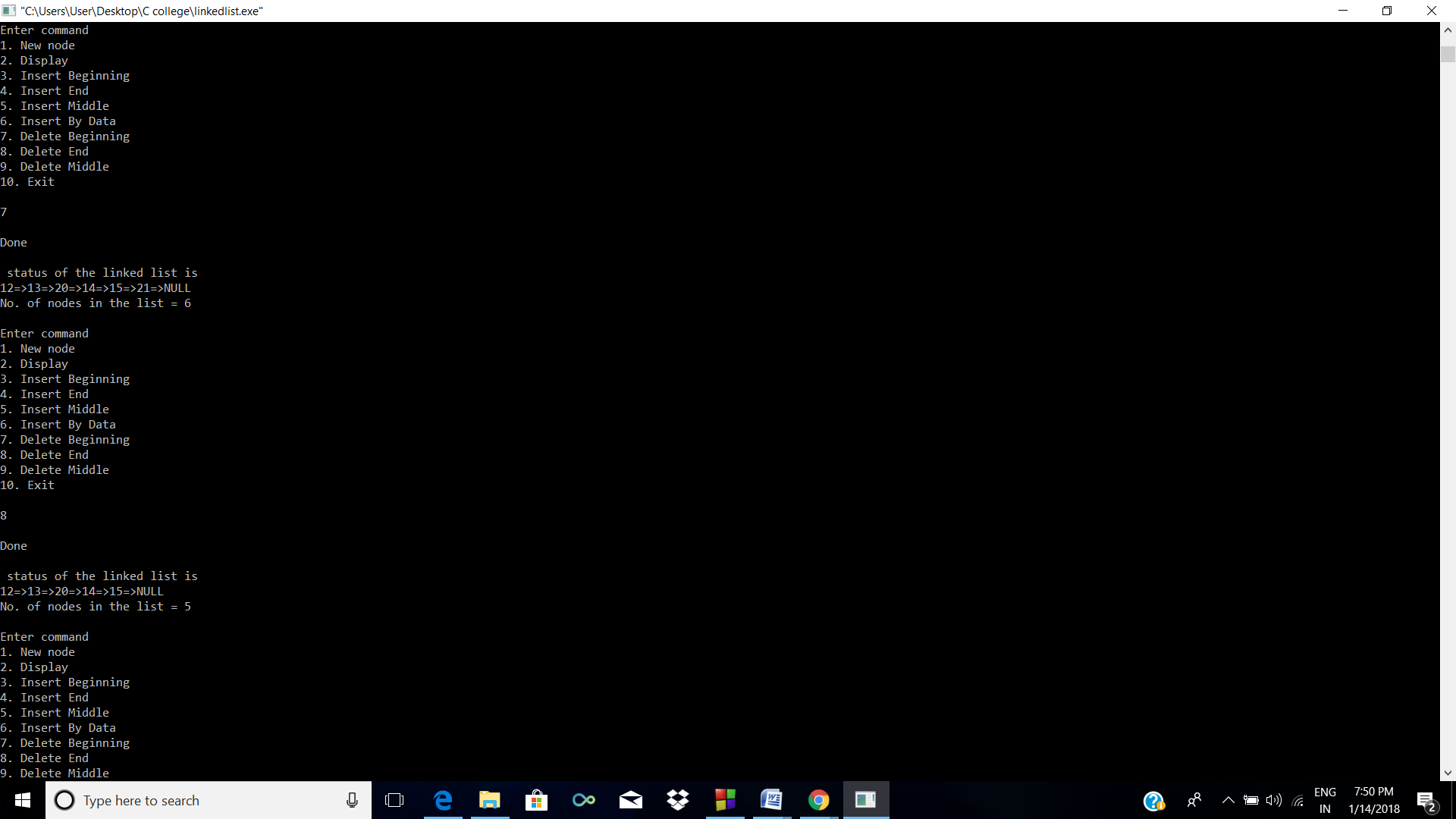
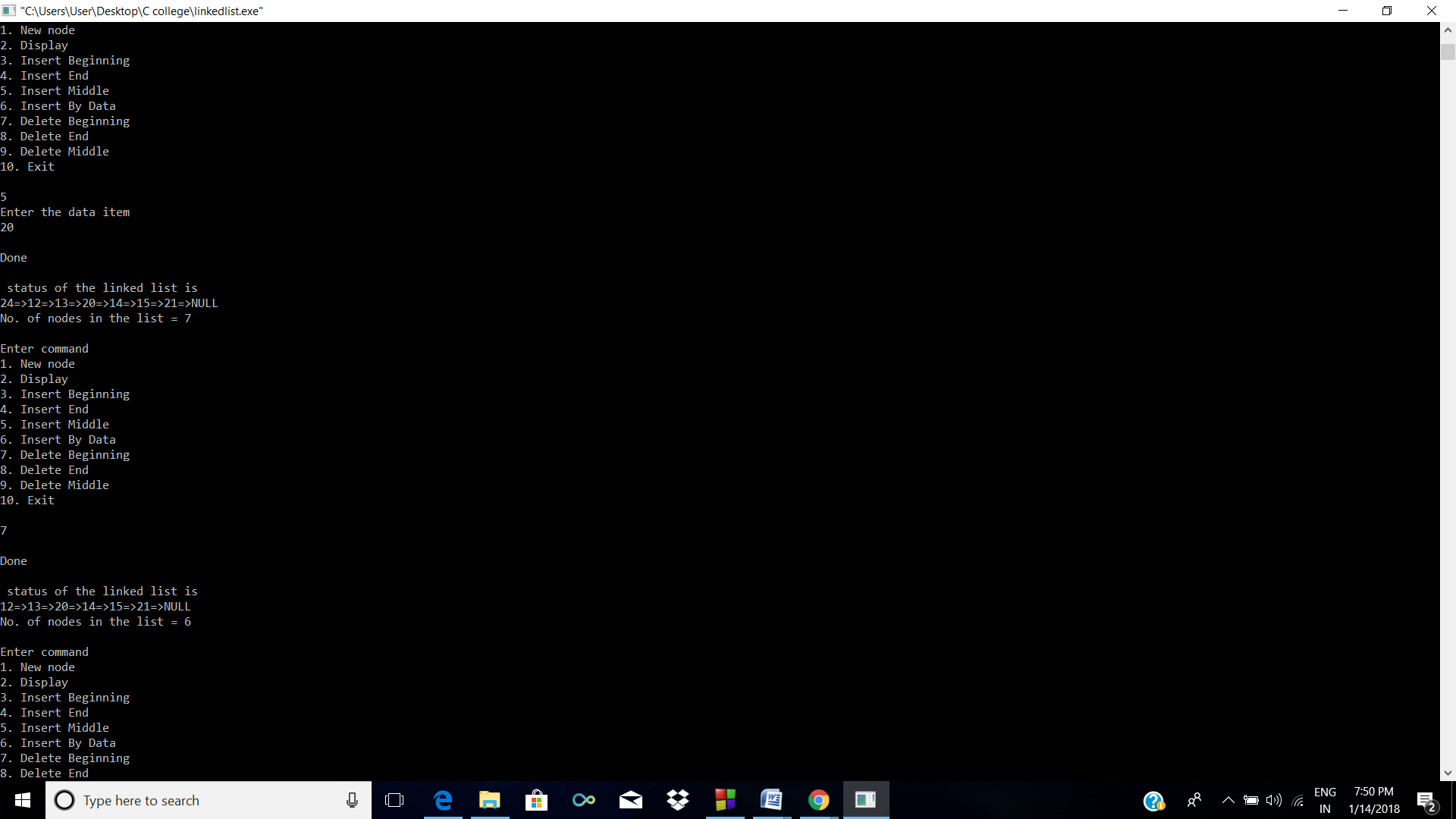
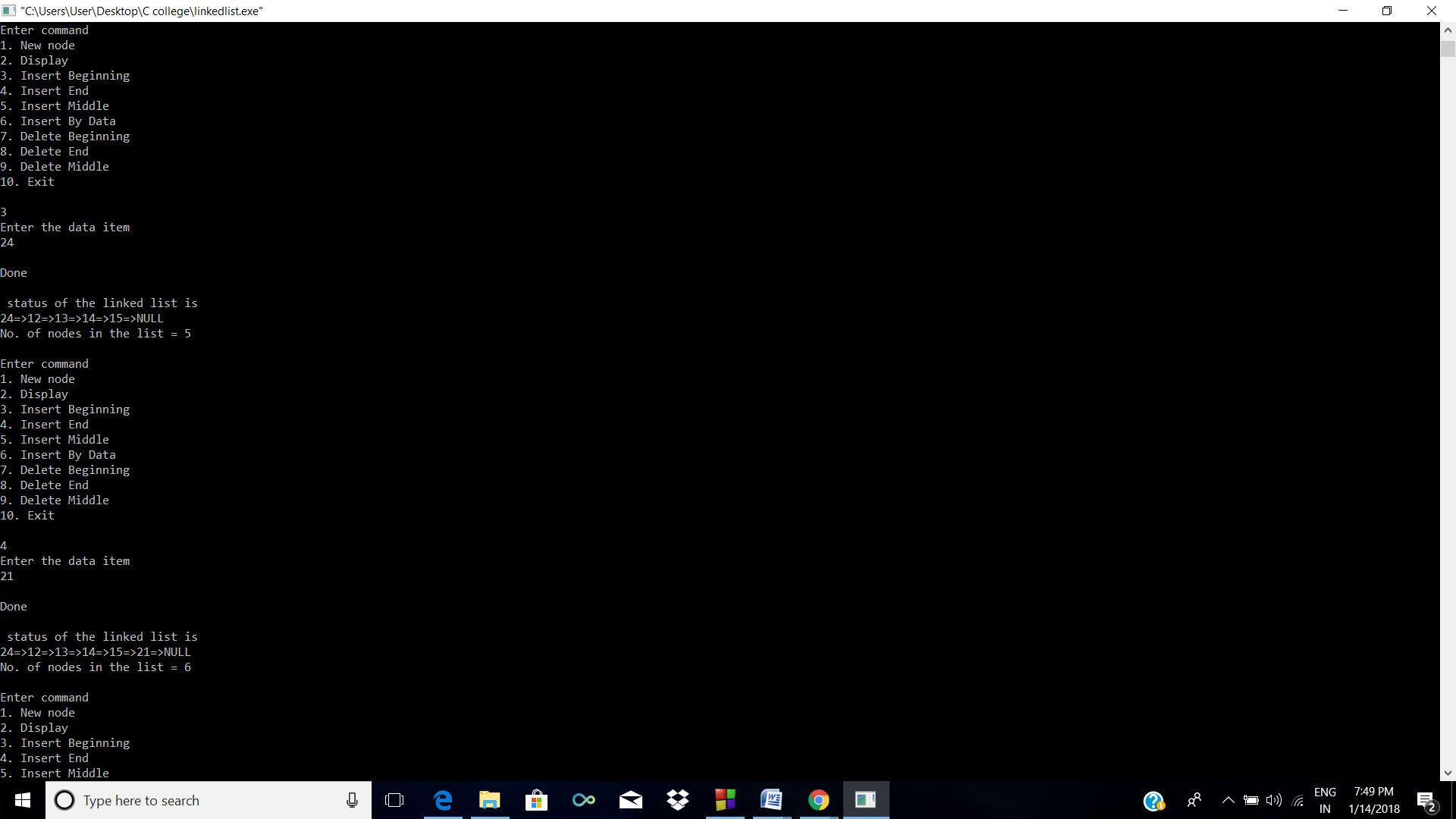
display();

}

**SCREENSHOT**







**2. Stack using linked list**

**CODE**

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>

void creation();

void display();

void peek();

void push();

void pop();

void isEmpty();

typedef struct node

{

int data;

struct node \*next;

}NODE;

int count = 0;

int choice = 1,n=0,x=0;

NODE \*head, \*first=0, \*temp = 0,\*temp2 = 0,\*temp3 = 0;

void main()

{

printf(" Welcome\n");

char option='T';

while(option=='T')

{

printf("\n");

printf("\n");

printf("Enter command\n");

printf("1. Creation\n");

printf("2. Push\n");

printf("3. Pop\n");

printf("4. Display\n");

printf("5. isEmpty\n");

printf("6. Peek\n");

printf("7. Exit\n\n");

scanf("%d",&n);

switch(n)

{

case 1:

creation();

break;

case 2:

push();

break;

case 3:

pop();

break;

case 4:

display();

break;

case 5:

isEmpty();

break;

case 6:

peek();

break;

case 7:

printf(" Thankyou!!\n");

exit(0);

}

}

}

void creation()

{

while (choice)

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

if (first != 0)

{

head->next = temp;

head = temp;

}

else

{

first = head = temp;

}

fflush(stdin);

printf("Do you want to continue(Type 0 or 1)?\n");

scanf("%d", &choice);

}

head->next = 0;

display();

}

void display()

{

count=0;

temp = first;

printf("\n status of the linked list is\n");

while (temp != 0)

{

printf("%d=>", temp->data);

count++;

temp = temp -> next;

}

printf("NULL\n");

printf("No. of nodes in the list = %d\n", count);

}

void peek()

{

temp = first;

while (temp->next!=0)

{

temp = temp -> next;

}

printf("Peek is %d",temp->data);

}

void push()

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

head->next=temp;

head = temp;

head->next = 0;

fflush(stdin);

printf("\nDone\n");

display();

}

void pop()

{

temp=first;

while (temp->next->next!=0)

{

/\*printf("%d=>", temp->data);\*/

count++;

temp = temp -> next;

}

temp->next=0;

fflush(stdin);

printf("\nDone\n");

display();

}

void isEmpty()

{

count=0;

temp = first;

printf("\n Status \n");

while (temp != 0)

{

count++;

temp = temp -> next;

}

if(count>0)

{

printf("It is not Empty");

}

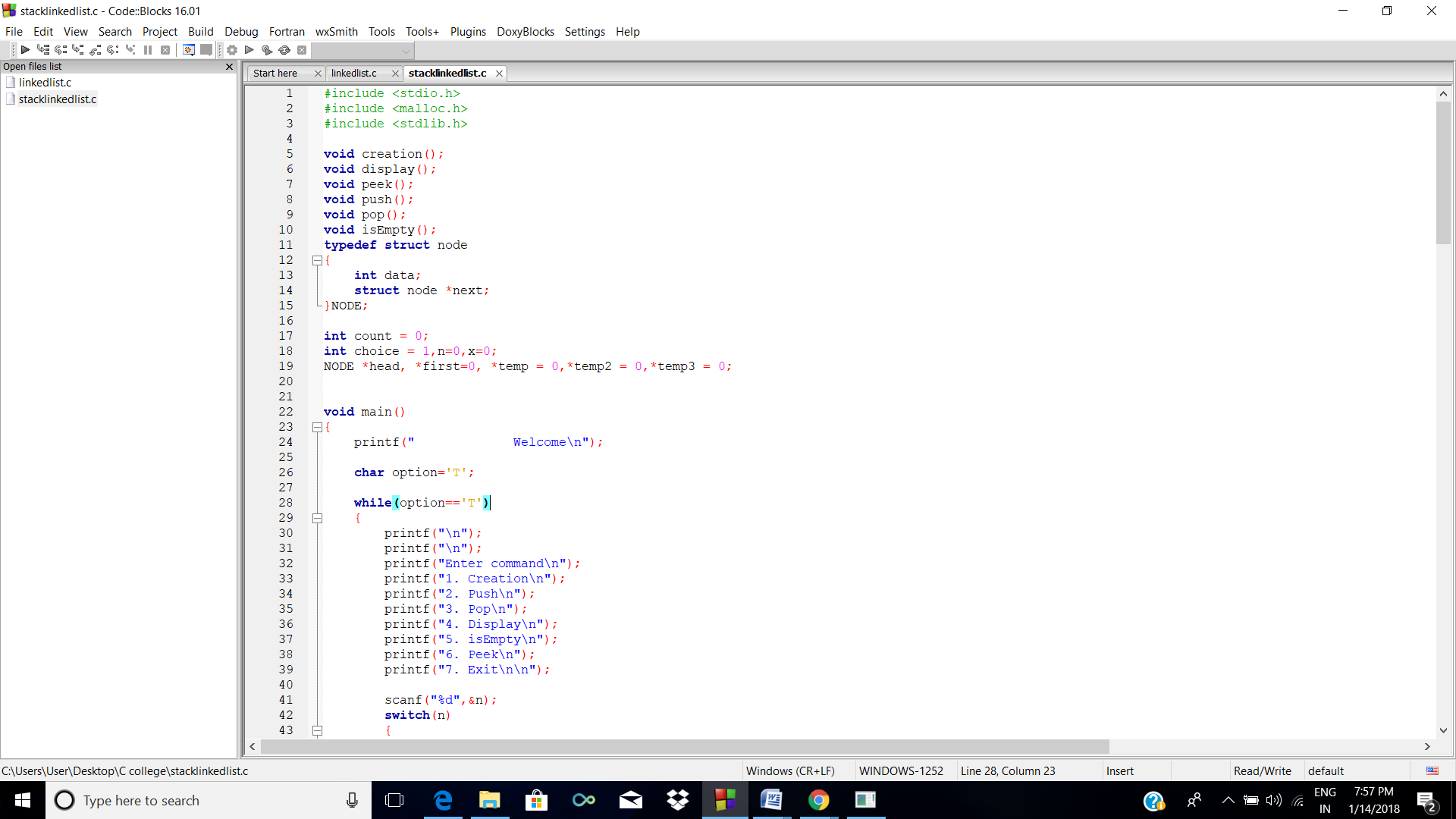
else{

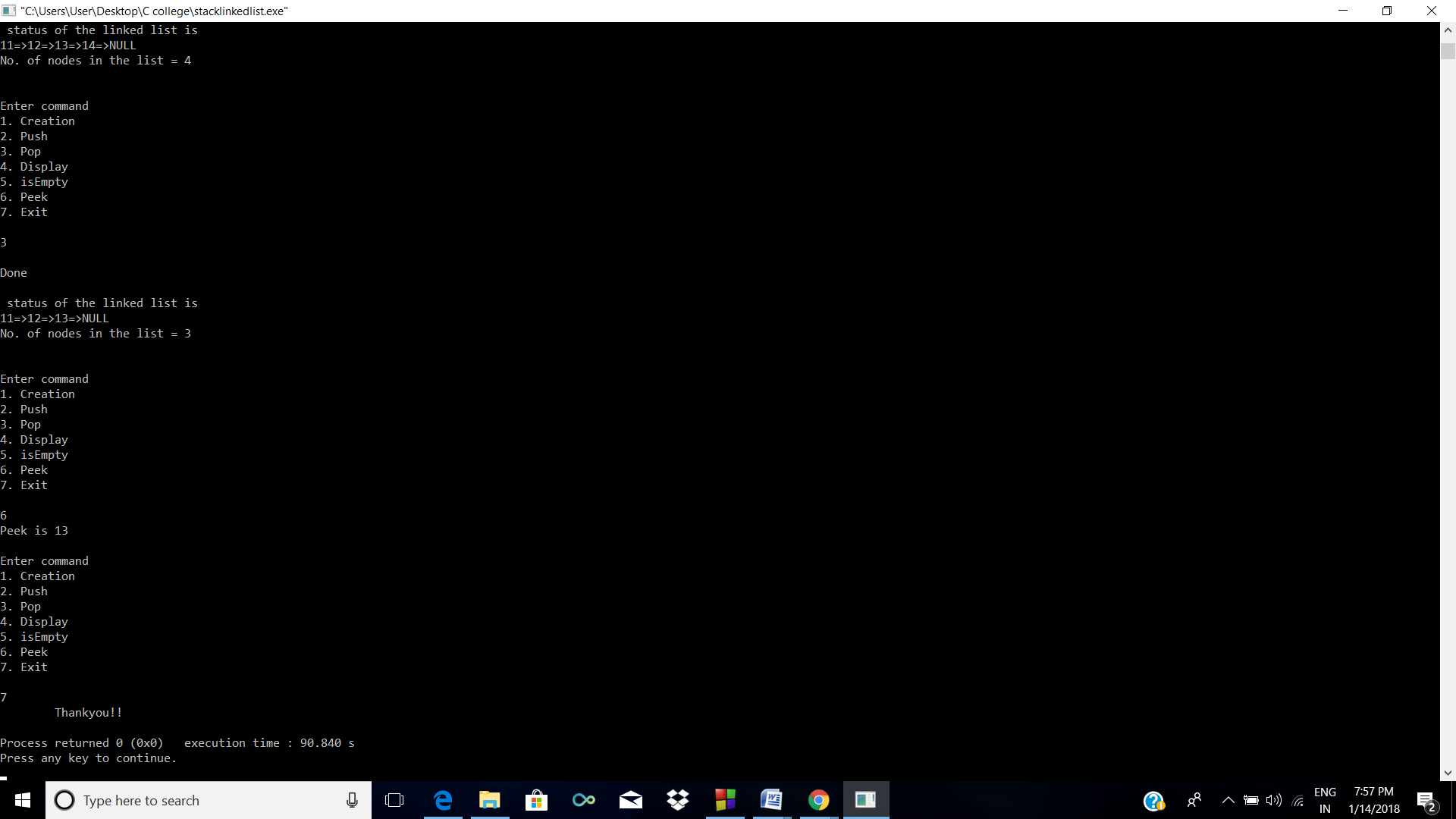
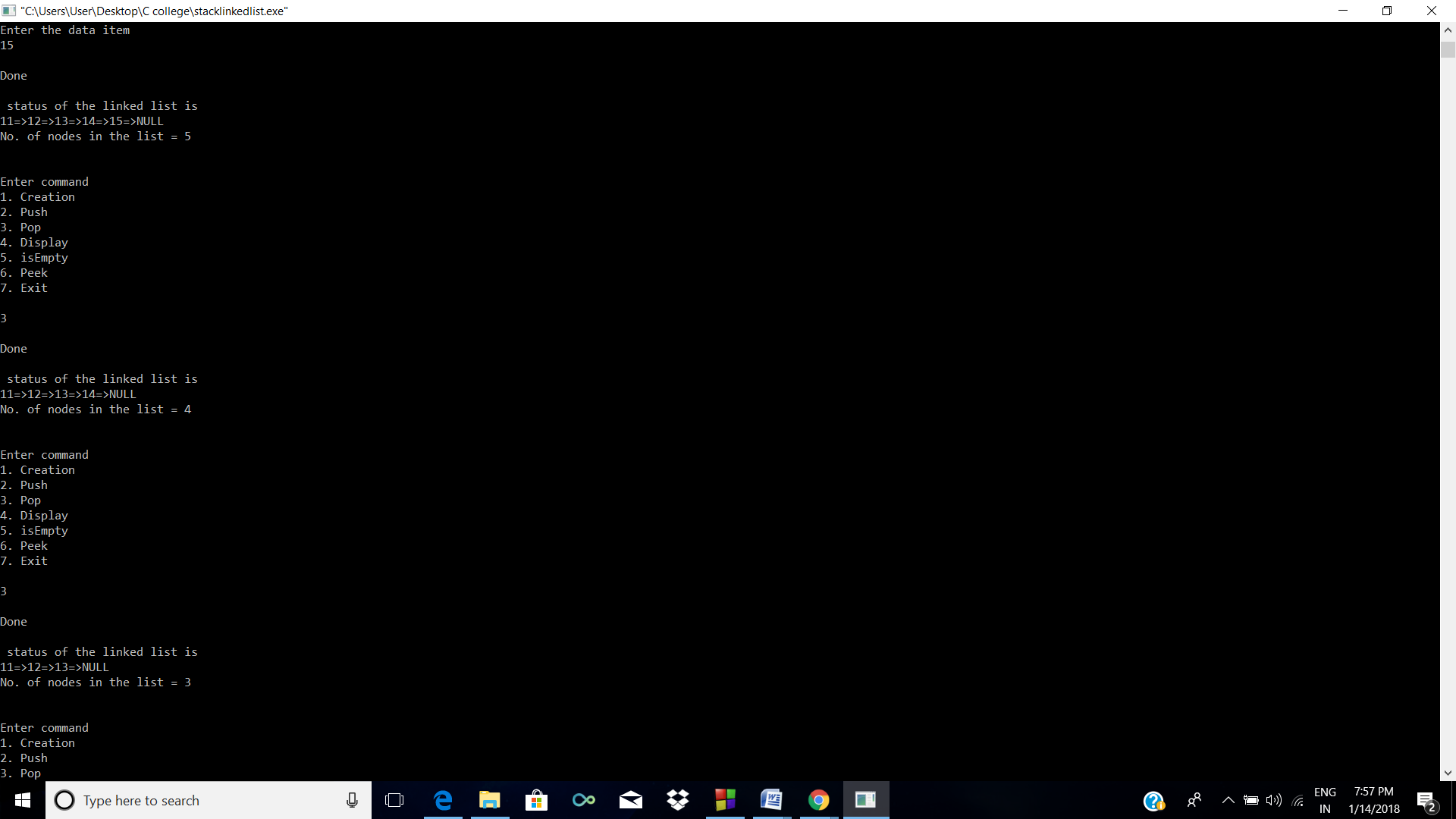
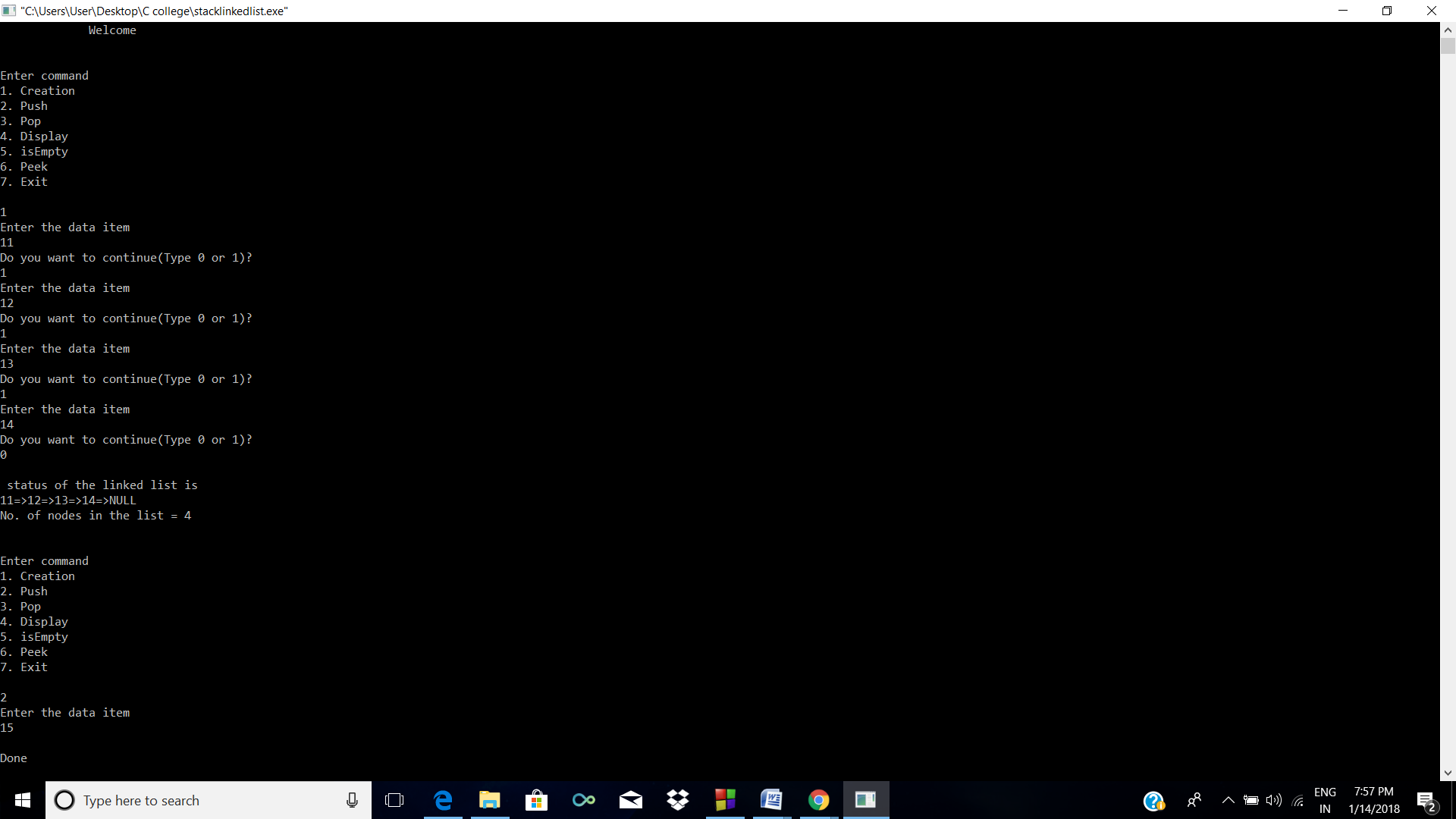
printf("It is empty\n");

}

}

**SCREENSHOTS**





**2. Queue using linked list**

**CODE**

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>

void newnode();

void display();

void enqueue();

void dequeue();

void peek();

void isEmpty();

typedef struct node

{

int data;

struct node \*next;

}NODE;

int count = 0;

int choice = 1,n=0,x=0;

NODE \*head, \*first=0, \*temp = 0,\*temp2 = 0,\*temp3 = 0;

void main()

{

printf(" 17BCE0042\n");

char option='T';

while(option=='T')

{

printf("\n");

printf("Enter command (17BCE0042)\n");

printf("1. Enqueue\n");

printf("2. Dequeue\n");

printf("3. Display\n");

printf("4. isEmpty\n");

printf("5. Peek\n");

printf("6. Exit\n\n");

scanf("%d",&n);

switch(n)

{

case 1:

enqueue();

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

isEmpty();

break;

case 5:

peek();

break;

case 6:

printf(" Thankyou!!\n");

exit(0);

}

}

}

void display()

{

count=0;

temp = first;

printf("\n status of the linked list is\n");

while (temp != 0)

{

printf("%d=>", temp->data);

count++;

temp = temp -> next;

}

printf("NULL\n");

printf("No. of nodes in the list = %d\n", count);

}

void enqueue()

{

temp = (NODE \*)malloc(sizeof(NODE));

printf("Enter the data item\n");

scanf("%d", &temp-> data);

if (first != 0)

{

head->next=temp;

head = temp;

head->next = 0;

}

else

{

first = head = temp;

head->next = 0;

}

fflush(stdin);

printf("\nDone\n");

display();

}

void peek()

{

if (first != 0)

{

temp = first;

while (temp->next!=0)

{

temp = temp -> next;

}

printf("Peek is %d",temp->data);

}

else

{

printf("List is empty");

}

}

void dequeue()

{

temp=first;

first = temp->next;

fflush(stdin);

printf("\nDone\n");

display();

}

void isEmpty()

{

count=0;

temp = first;

printf("\n Status \n");

while (temp != 0)

{

count++;

temp = temp -> next;

}

if(count>0)

{

printf("It is not Empty");

}

else{

printf("It is empty\n");

}

}

**SCREENSHOT**

