Name: Oaish Qazi Roll no: 210455

Practical No 10: Perform Operation On Singly Linked List

* Insertion at Beginning

CODE:

#include <stdio.h>

#include <stdlib.h>

typedef struct linked\_list

{ int data;

struct linked\_list \*next; } node;

int printList(node \*ptr)

{ printf("[ %d ]", ptr->data);

ptr = ptr->next;

while (ptr != NULL)

{

printf("--->[ %d ]", ptr->data);

ptr = ptr->next;

}

printf("\n");

return 0;

}

int insertBegin(node \*\*ptr, int num)

{ node \*temp;

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

temp->data = num;

if (\*ptr == NULL)

{

temp->next = NULL;

} else {

temp->next = \*ptr;

} \*ptr = temp;

}

int main()

{

node \*start, \*ptr, \*temp;

int num, n, index, i = 1;

start = (node \*)malloc(sizeof(node));

printf("Oaish Qazi - 210455\n");

printf("Enter the Size of the List: ");

scanf("%d", &n);

printf("Node %d: ", i);

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

start->next = NULL;

temp = start;

for (i = 2; i <= n; i++)

{

ptr = (node \*)malloc(sizeof(node));

printf("Node %d: ", i); ("%d", &ptr->data);

ptr->next = NULL; temp->next = ptr; temp = ptr;

}

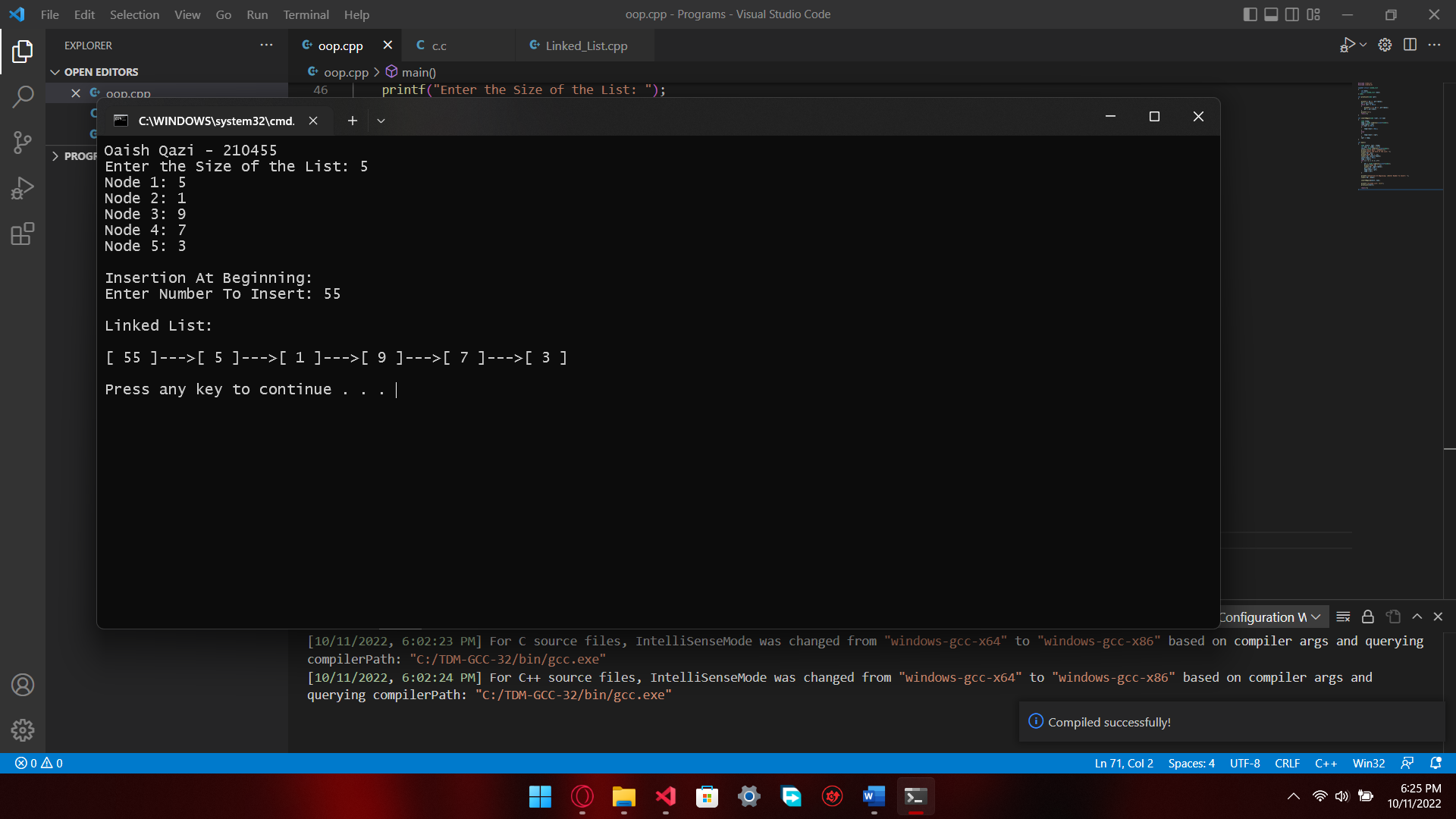
printf("\nInsertion At Beginning: \nEnter Number To Insert: "); scanf("%d", &num);

insertBegin(&start, num);

printf("\nLinked List: \n\n");

printList(start); }

OUTPUT:



* Insertion At Middle

CODE:

#include <stdio.h>

#include <stdlib.h>

typedef struct linked\_list

{

int data;

struct linked\_list \*next;

}node;

int printList(node \*ptr)

{

printf("[ %d ]", ptr->data);

ptr = ptr->next;

while (ptr != NULL)

{

printf("--->[ %d ]", ptr->data);

ptr = ptr->next;

}

printf("\n");

return 0;

}

void insertMid(node \*\*temp,int num,int index)

{

node \*ptr, \*loc;

int i=1;

ptr = (node \*)malloc(sizeof(node));

ptr->data = num;

if (\*temp == NULL)

{

ptr->next = NULL;

\*temp = ptr;

return;

}

loc = \*temp;

while (i < index - 1)

{

loc = loc->next;

if (loc == NULL)

{

printf(" link list size exceed");

return;

}

i++;

}

ptr->next = loc->next;

loc->next = ptr;

}

int main()

{

node \*start, \*ptr, \*temp;

int num, n, index, i = 1;

start = (node \*)malloc(sizeof(node));

printf("Oaish Qazi - 210455\n");

printf("Enter the Size of the List: ");

scanf("%d", &n);

printf("Node %d: ", i);

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

start->next = NULL;

temp = start;

for (i = 2; i <= n; i++)

{

ptr = (node \*)malloc(sizeof(node));

printf("Node %d: ", i);

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

ptr->next = NULL;

temp->next = ptr;

temp = ptr;

}

printf("\nInsertion At Middle: \nEnter Number To Insert: ");

scanf("%d", &num);

printf("Enter Index: ");

scanf("%d", &index);

insertMid(&start, num, index);

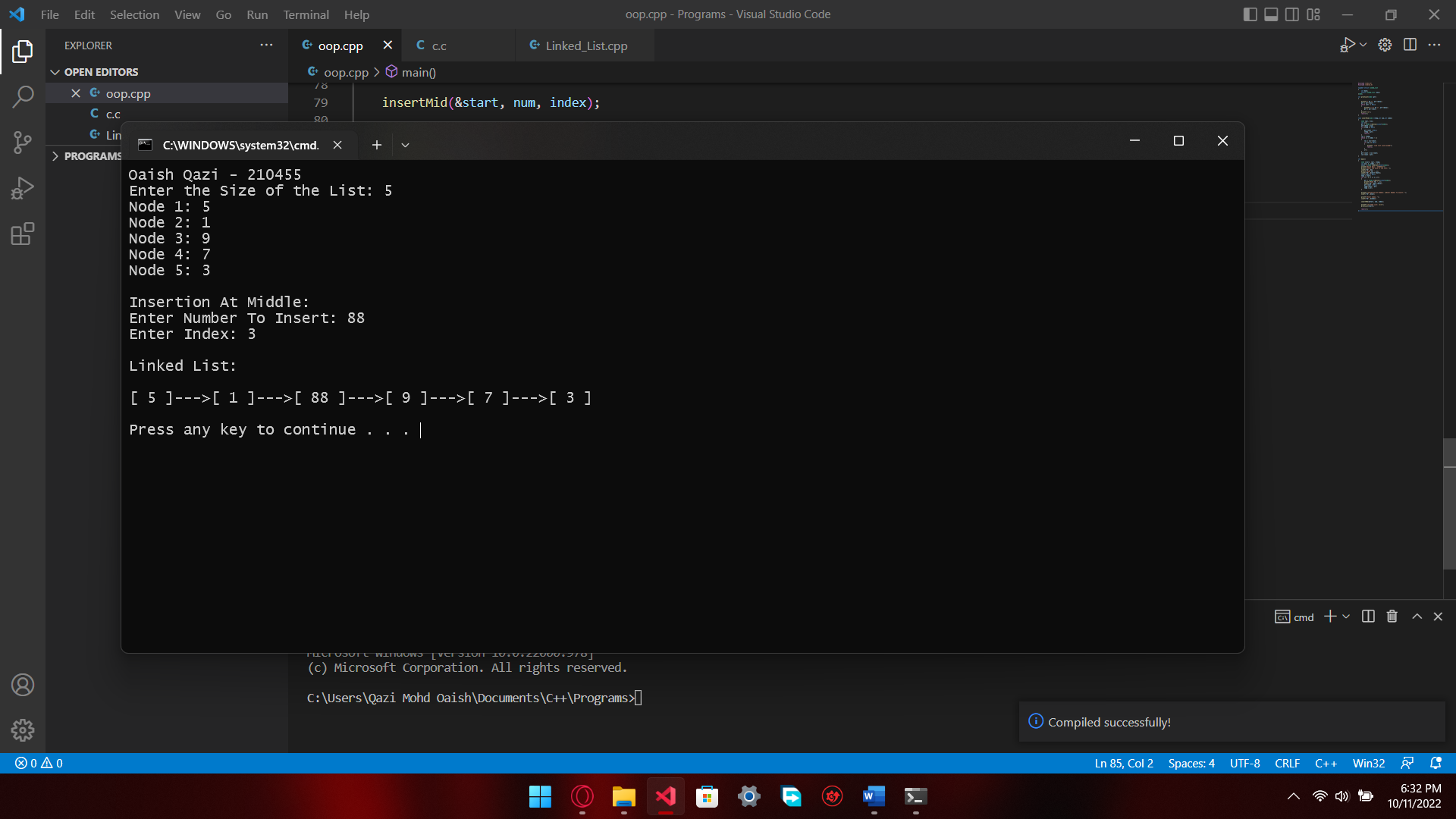
printf("\nLinked List: \n\n");

printList(start);

return 0;

}

OUTPUT:



* Insertion At End

CODE:

#include <stdio.h>

#include <stdlib.h>

typedef struct linked\_list

{

int data;

struct linked\_list \*next;

}node;

int printList(node \*ptr)

{

printf("[ %d ]", ptr->data);

ptr = ptr->next;

while (ptr != NULL)

{

printf("--->[ %d ]", ptr->data);

ptr = ptr->next;

}

printf("\n");

return 0;

}

int insertEnd(node \*\*temp, int num)

{

node \*ptr, \*loc;

ptr = (node \*)malloc(sizeof(node));

ptr->data = num;

ptr->next = NULL;

if (\*temp == NULL)

{

\*temp = ptr;

return 0;

}

loc = \*temp;

while (loc->next != NULL)

loc = loc->next;

loc->next = ptr;

return 0;

}

int main()

{

node \*start, \*ptr, \*temp;

int num, n, index, i = 1;

start = (node \*)malloc(sizeof(node));

printf("Oaish Qazi - 210455\n");

printf("Enter the Size of the List: ");

scanf("%d", &n);

printf("Node %d: ", i);

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

start->next = NULL;

temp = start;

for (i = 2; i <= n; i++)

{

ptr = (node \*)malloc(sizeof(node));

printf("Node %d: ", i);

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

ptr->next = NULL;

temp->next = ptr;

temp = ptr;

}

printf("\nInsertion At The End: \nEnter Number To Insert: ");

scanf("%d", &num);

insertEnd(&start,num);

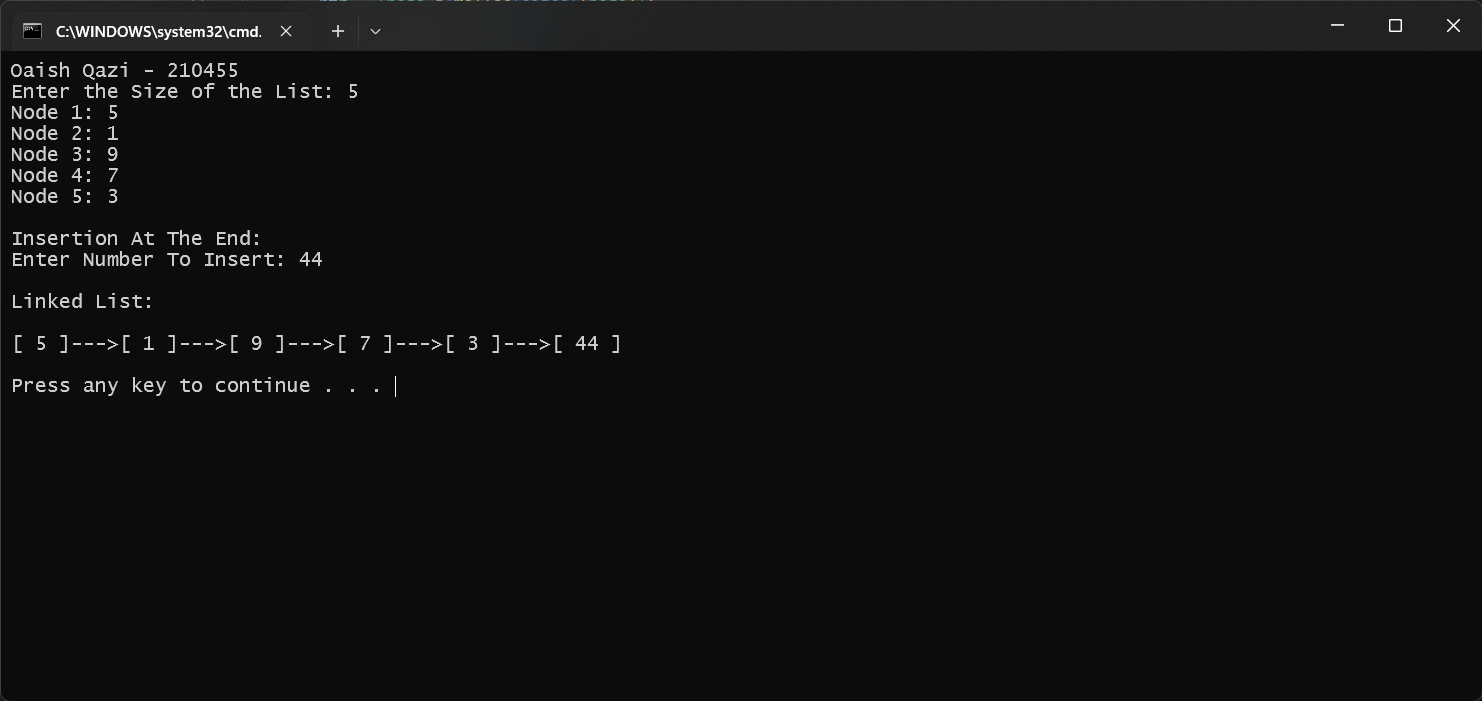
printf("\nLinked List: \n\n");

printList(start);

return 0;

}

OUTPUT:



* Deletion By Number

CODE:

#include <stdio.h>

#include <stdlib.h>

typedef struct linked\_list

{

int data;

struct linked\_list \*next;

} node;

int printList(node \*ptr)

{

printf("[ %d ]", ptr->data);

ptr = ptr->next;

while (ptr != NULL)

{

printf("--->[ %d ]", ptr->data);

ptr = ptr->next;

}

printf("\n");

return 0;

}

void deleteNum(node \*\*q, int num)

{

node \*old, \*temp;

temp = \*q;

old = temp;

while (temp != NULL)

{

if (temp->data == num)

{

if (temp == \*q) {

\*q = temp->next;

} else {

old->next = temp->next;

}

free(temp); return;

}

old = temp;

temp = temp->next;

}

printf(“Number Not Present\n”);

}

int main()

{

node \*start, \*ptr, \*temp;

int num, n, index, i = 1;

start = (node \*)malloc(sizeof(node));

printf("Oaish Qazi - 210455\n");

printf("Enter the Size of the List: ");

scanf("%d", &n);

printf("Node %d: ", i);

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

start->next = NULL;

temp = start;

for (i = 2; i <= n; i++)

{

ptr = (node \*)malloc(sizeof(node));

printf("Node %d: ", i);

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

ptr->next = NULL;

temp->next = ptr;

temp = ptr;

}

printf("\nDeletion By Number: \nEnter Number To Delete: ");

scanf("%d", &num);

deleteNum(&start, num);

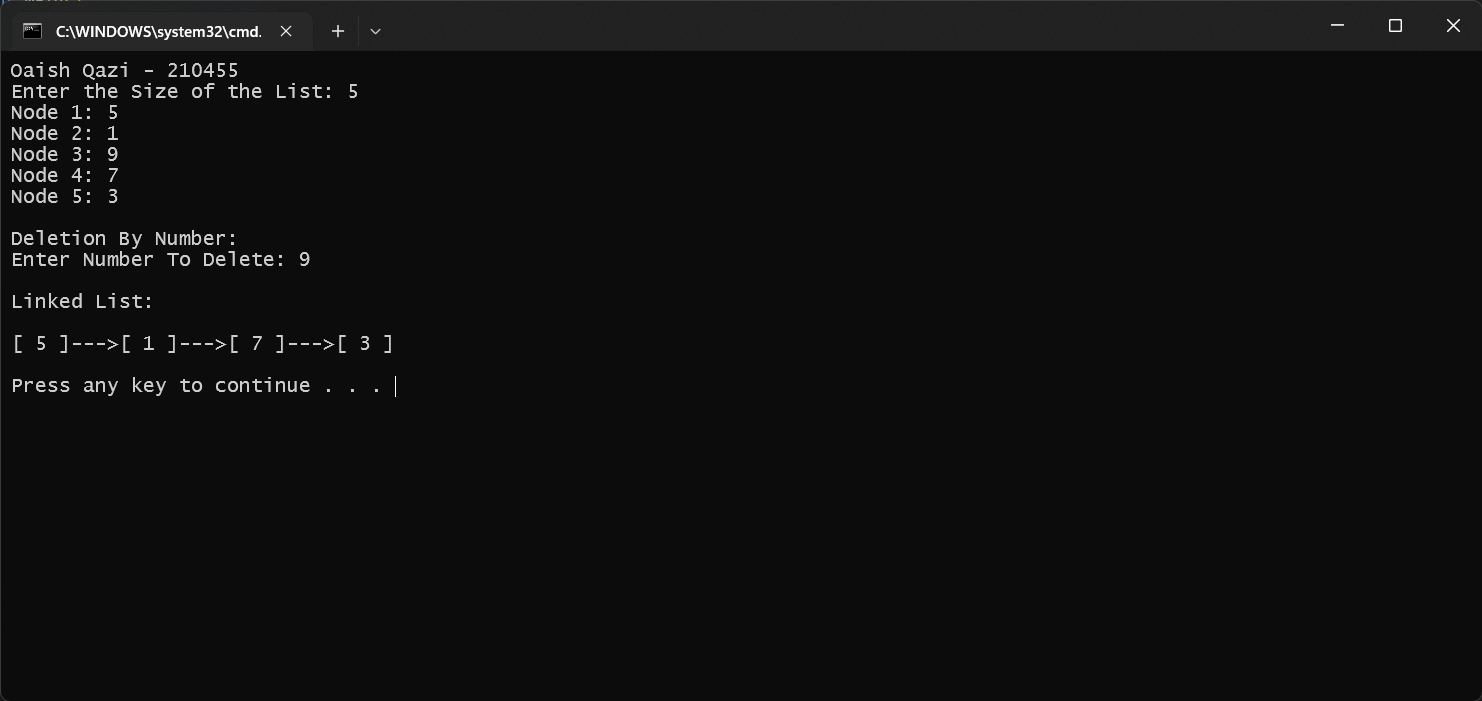
printf("\nLinked List: \n\n");

printList(start);

return 0;

}

OUTPUT:



* Deletion By Index

CODE:

#include <stdio.h>

#include <stdlib.h>

typedef struct linked\_list

{

int data;

struct linked\_list \*next;

} node;

int printList(node \*ptr)

{

printf("[ %d ]", ptr->data);

ptr = ptr->next;

while (ptr != NULL){

printf("--->[ %d ]", ptr->data);

ptr = ptr->next;

}

printf("\n");

return 0;

}

void deleteIndex(node \*\*q, int index)

{

node \*old, \*temp; int i = 1; old = \*q;

if (index == 1){

\*q = old->next;

free(old); return; }

temp = \*q;

while (i < index)

{

old = temp;

temp = temp->next;

if (temp == NULL) {

printf(" link list size exceed");

return; }

i++; }

old->next = temp->next; free(temp);

}

void main()

{

node \*start, \*ptr, \*temp;

int num, n, index, i = 1;

start = (node \*)malloc(sizeof(node));

printf("Oaish Qazi - 210455\n");

printf("Enter the Size of the List: ");

scanf("%d", &n);

printf("Node %d: ", i);

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

start->next = NULL;

temp = start;

for (i = 2; i <= n; i++)

{

ptr = (node \*)malloc(sizeof(node));

printf("Node %d: ", i);

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

ptr->next = NULL;

temp->next = ptr;

temp = ptr;

}

printf("\nDeletion By Index:\n");

printf("Enter Index: "); scanf("%d", &index);

deleteIndex(&start, index);

printf("\nLinked List: \n\n");

printList(start);

}

OUTPUT:

