22.01 2024

Singly Linked List

Insertion and Deletion of a node at first position, at end of list.

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

struct node * start = NULL;
struct node *insert_beg (struct node*);
struct node * insert_end(struct node*); struct node *insert_at_pos (struct no
struct node * delete_beg (struct node*);
struct node *delete_end(struct node*); struct node* delete_at_pos (struct
struct node * display (struct node*);

int main()
{ int option;
    do {
     printf("\n *** MENU***");
     printf ("\n 1: Insert at beginning");
     printf ("\n 2: Insert at the end"); printf ("\n 3: Insert at a specific po
     printf ("\n 4: Delete at beginning");
     printf ("\n 5: Delete at the end"); printf ("\n 6: Delete from a specific po
     printf ("\n 7: Display");
     printf ("\n 8: Exit");
     printf (" Enter your option:");
     Scanf ("%d", &option);
     Switch (option)
   { case 1: Start = insert_beg (start);
             break;
     case 2: Start = insert-end (Start);
             break;
    case 3: Start = insert _at - pos(start);
                    break;
    case 4: start = delete_beg(start);
                    break;
    case 5: start = delete_end(start);   case 6: start=delete_at_pos
                    break;                               (start);
                                                    break;
    case 7: start = display(start);
                    break;
    }
}while (option != 8);                   struct node *temp;
  getch();                              while (start != NULL){
  return 0; }                               temp=start;
                                            start=start ->next;
                                            free(temp); }

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;
  printf ("Inserted at the beginning\n");
  return start; }

struct node * insert_end (struct node *start)
{ struct node *ptr, *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 = NULL;
   ptr = start;
   while (ptr -> next != NULL)
        ptr = ptr -> next;
   ptr -> next = new_node;
   printf (" Inserted at the end\n");
   return start; }
```

[illegible]
22/1/23

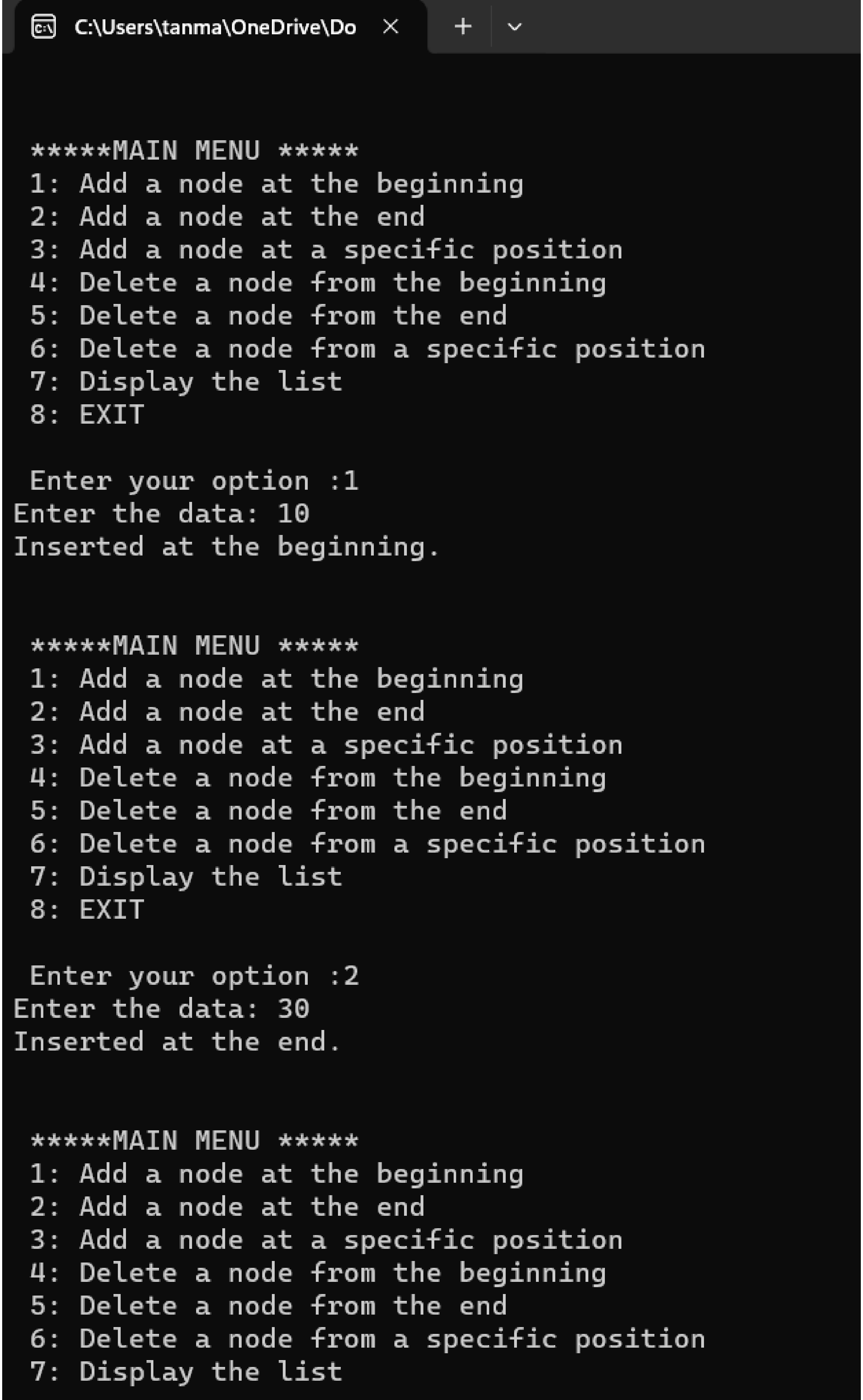
```
Struct node *delete-beg (struct node *Start)
{  Struct node *ptr;
   ptr = start;
   if (ptr -> next == NULL)
    { printf (" Empty List. Can't be deleted");
      return Start; }
   else {  Start = Start -> next;
           free(ptr);
           printf (" Deleted at beginning");
           return Start; }
}
Struct node *delete_end (struct node *Start)
{  Struct node *ptr, *ptr1;
   ptr = start;
   if (ptr -> next == NULL)
   { printf (" Empty List. Can't be deleted");
      return Start; }
   else {  while (ptr -> next != NULL)
              { ptr1 = ptr;
           ptr = ptr -> next; }
             ptr1 -> next = NULL;
             free(ptr);
             printf (" Deleted at the end"];
             return start; }
}

Struct node *display (struct node *start)
{     Struct node *ptr;
      ptr = Start;
      if (ptr -> next == NULL)
      { printf (" Empty List.");
         return Start; }
      else { while(ptr -> next != NULL)
               {  printf (" \t %d", ptr -> data);
                     ptr = ptr -> next;
               } return Start; }
}
```

```
Struct node * insert-at-pos (struct node * start)
{ Struct node * new-node, *ptr, *preptr;
  int pos, num;
  printf (" Enter the position to insert at:");
  Scanf (" %d", &pos);
  printf (" Enter the data: ");
  scanf (" %d", &num);
  new-node = (struct node *) malloc (sizeof (struct node));
  new-node -> data = num;
  new-node -> next = NULL;
  if (pos == 1) {
      new-node -> next = start;
      start = new-node;
      printf (" Inserted at position %d \n", pos);
      return start; }
  else { ptr = start;
         int i;
      for (int i = 1; i < pos && ptr != NULL; i++)
      { preptr = ptr;
        ptr = ptr -> next; }
      if (ptr == NULL && pos > i)
      { printf (" Invalid position. \n");
        return start; }
     preptr -> next = new-node;
     new-node -> next = ptr;
     printf (" Inserted at position %d", pos);
     return start; }
}

Struct node * delete-at-pos (struct node * start)
{ struct node *ptr, *preptr;
  int pos;
  printf (" Enter the position to delete");
  scanf (" %d", &pos);
  if (start == NULL)
  { printf (" Empty list. Can't be deleted\n);
    return start; }
}
```

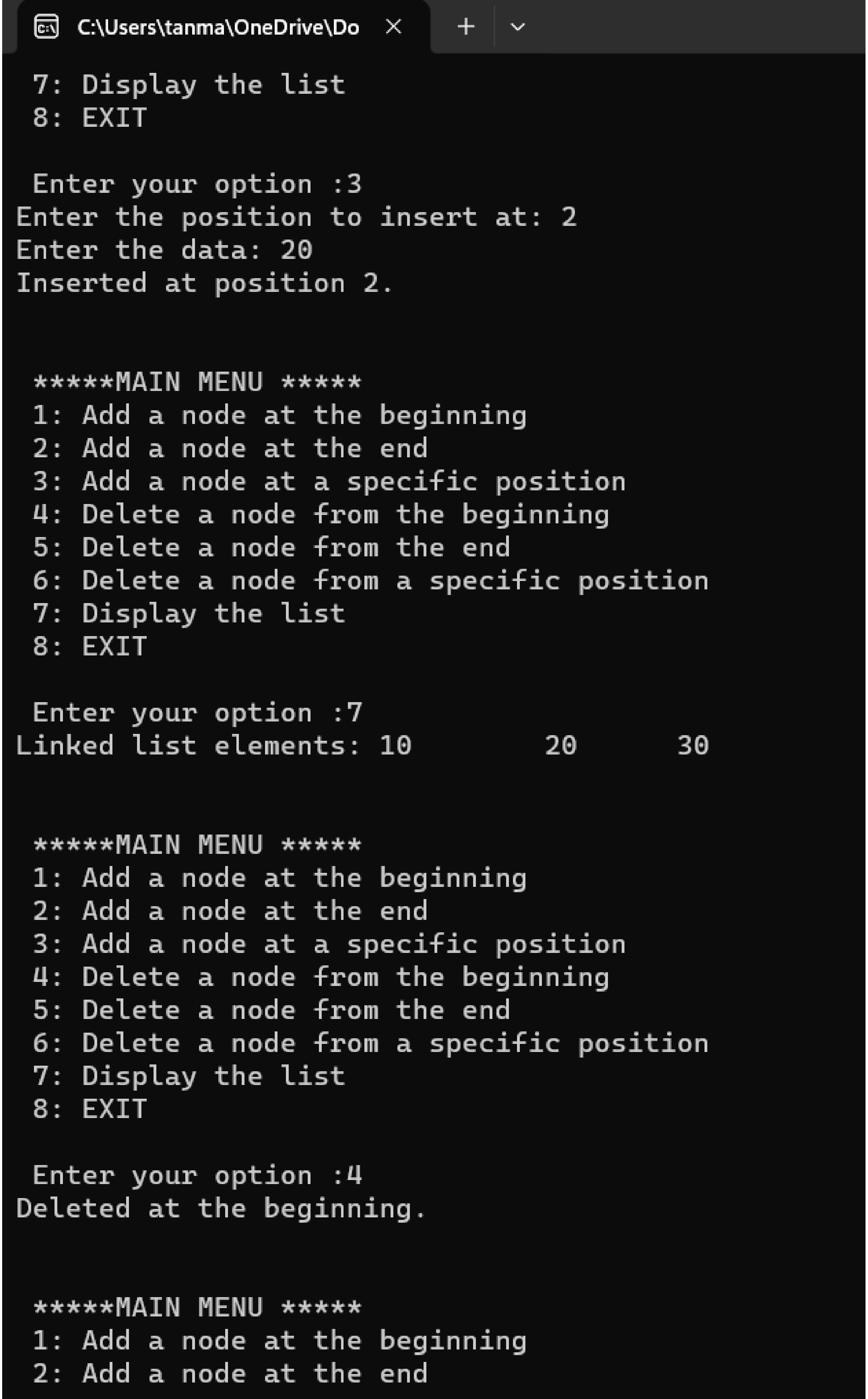
```
    ptr = start;
    if (pos == 1)
   { start = start -> next;
     free(ptr);
     printf (" Deleted at position %d \n", pos);
     return start;
   }
   else {
      for (int i = 1; i < pos && ptr != NULL; i++)
      {  preptr = ptr;
         ptr = ptr -> next; }
      if (ptr == NULL)
       { printf (" Invalid position \n");
         return start;
       }
       preptr -> next = ptr -> next;
       free (ptr);
       printf (" Deleted at position %d \n", pos);
       return start;
    }
}
```

<u>OUTPUT</u>:

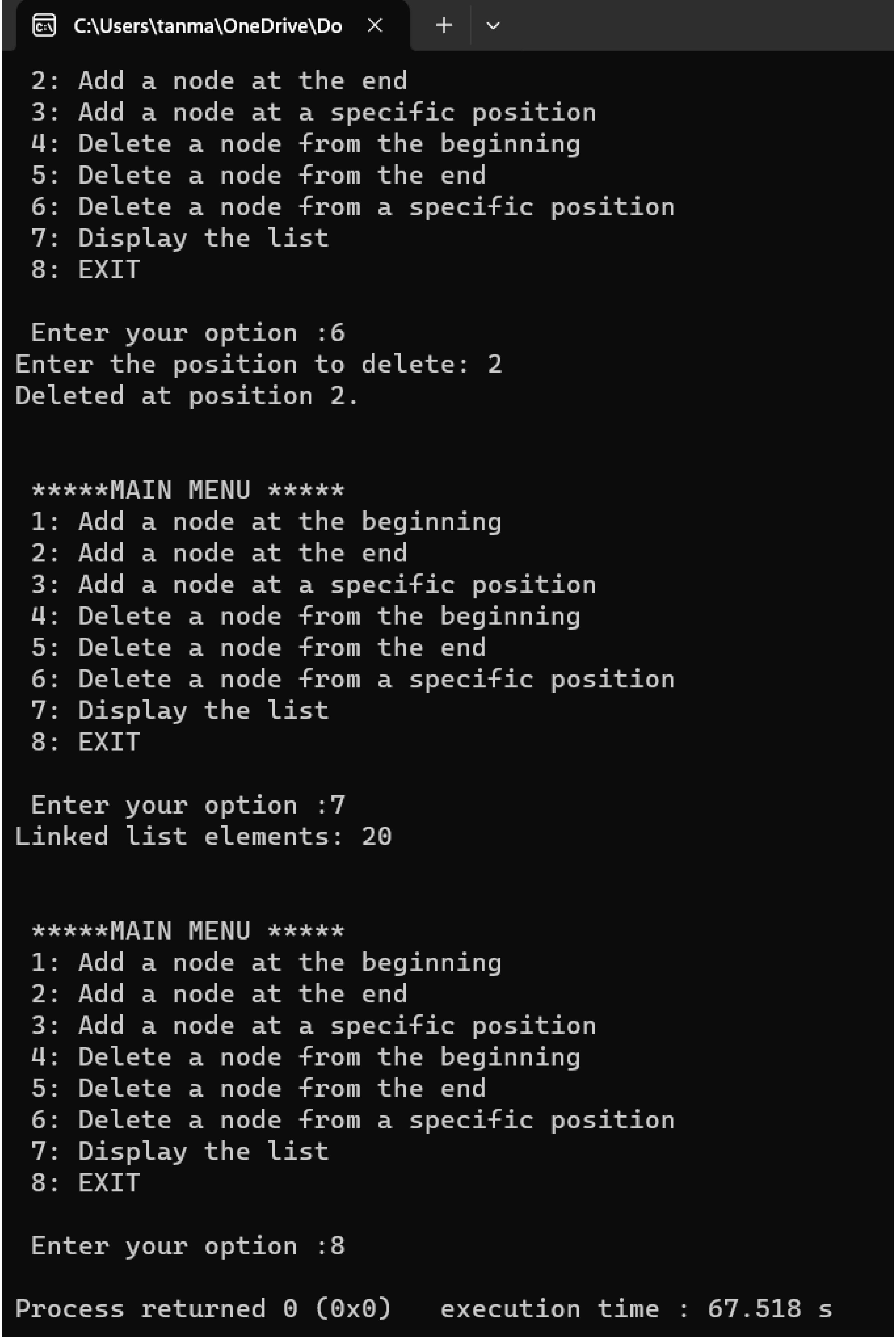
```
   ***Main Menu***
1: Add a node at the beginning.
2: Add a node at the end.
3: Add a node at a specific position.
4: Delete a node from the beginning.
5: Delete a node from the end.
6: Delete a node from a specific position.
7: Display the list.
8: EXIT.
Enter your option: 1
Enter the data: 10
Inserted at the beginning.
```



```
 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :1
Enter the data: 10
Inserted at the beginning.


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :2
Enter the data: 30
Inserted at the end.


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
```



```
 7: Display the list
 8: EXIT

 Enter your option :3
Enter the position to insert at: 2
Enter the data: 20
Inserted at position 2.


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :7
Linked list elements: 10        20      30


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :4
Deleted at the beginning.


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
```



```
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :6
Enter the position to delete: 2
Deleted at position 2.


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :7
Linked list elements: 20


 *****MAIN MENU *****
 1: Add a node at the beginning
 2: Add a node at the end
 3: Add a node at a specific position
 4: Delete a node from the beginning
 5: Delete a node from the end
 6: Delete a node from a specific position
 7: Display the list
 8: EXIT

 Enter your option :8

Process returned 0 (0x0)   execution time : 67.518 s
```