ANVITA KUMAR
C-22
Roll No.: 2104097

```
//Write a menu driven code to implement Circular Linked List
#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
struct node
{
   int data;
   struct node *next;
};
struct node *start = NULL;
struct node *createCLL(struct node *start);
struct node *display(struct node *start);
struct node *InsertAtBeginning(struct node *start);
struct node *InsertAtEnd(struct node *start);
struct node *DeleteBeginning(struct node *start);
struct node *DeleteEnd(struct node *start);
struct node *ForwardTraversal(struct node *start);
struct node *BackwardTraversal(struct node *start);
struct node *Count(struct node *start);
int main()
{
   int choice;
   start = createCLL(start);
   printf("\nCIRCULAR LINKED LIST CREATED\n");
   start = display(start);
   do {
      printf("\n\n****List of Operations****");
      printf("\n 1: Insert at beginning");
      printf("\n 2: Insert at end");
      printf("\n 3: Delete from beginning");
```

ANVITA KUMAR
C-22
Roll No.: 2104097

```
        printf("\n 4: Delete from end");
        printf("\n 5: Forward Traversal");
        printf("\n 6: Backward Traversal");
        printf("\n 7: Count number of nodes");
        printf("\n 8: EXIT");
        printf("\n\nEnter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
        case 1:
           start = InsertAtBeginning(start);
           printf("\n");
           start = display(start);
           break;
        case 2:
           start = InsertAtEnd(start);
           printf("\n");
           start = display(start);
           break;
        case 3:
           start = DeleteBeginning(start);
           printf("\n");
           start = display(start);
           break;
        case 4:
           start = DeleteEnd(start);
           printf("\n");
           start = display(start);
           break;
        case 5:
           start = ForwardTraversal(start);
           printf("\n");
```

ANVITA KUMAR
C-22
Roll No.: 2104097

```
            break;
        case 6:
            start = BackwardTraversal(start);
            printf("\n");
            start = display(start);
            break;
        case 7:
            start = Count(start);
            printf("\n");
            break;
        case 8:
            printf("\n\tEXIT POINT");
            break;
        }
    } while (choice != 8);
    return 0;
}
struct node *createCLL(struct node *start)
{
    struct node *new_node, *ptr;
    int num;
    printf("\nEnter a value(enter -1 to end): ");
    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
```

ANVITA KUMAR
C-22
Roll No.: 2104097

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

ANVITA KUMAR
C-22
Roll No.: 2104097

```
   ptr = start;
   while (ptr->next != start)
      ptr = ptr->next;
   ptr->next = new_node;
   new_node->next = start;
   start = new_node;
   return start;
}
struct node *InsertAtEnd(struct node *start)
{
   struct node *ptr, *new_node;
   int num;
   printf("\n 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;
   return start;
}
struct node *DeleteBeginning(struct node *start)
{
   struct node *ptr;
   ptr = start;
   while (ptr->next != start)
      ptr = ptr->next;
   ptr->next = start->next;
   free(start);
```

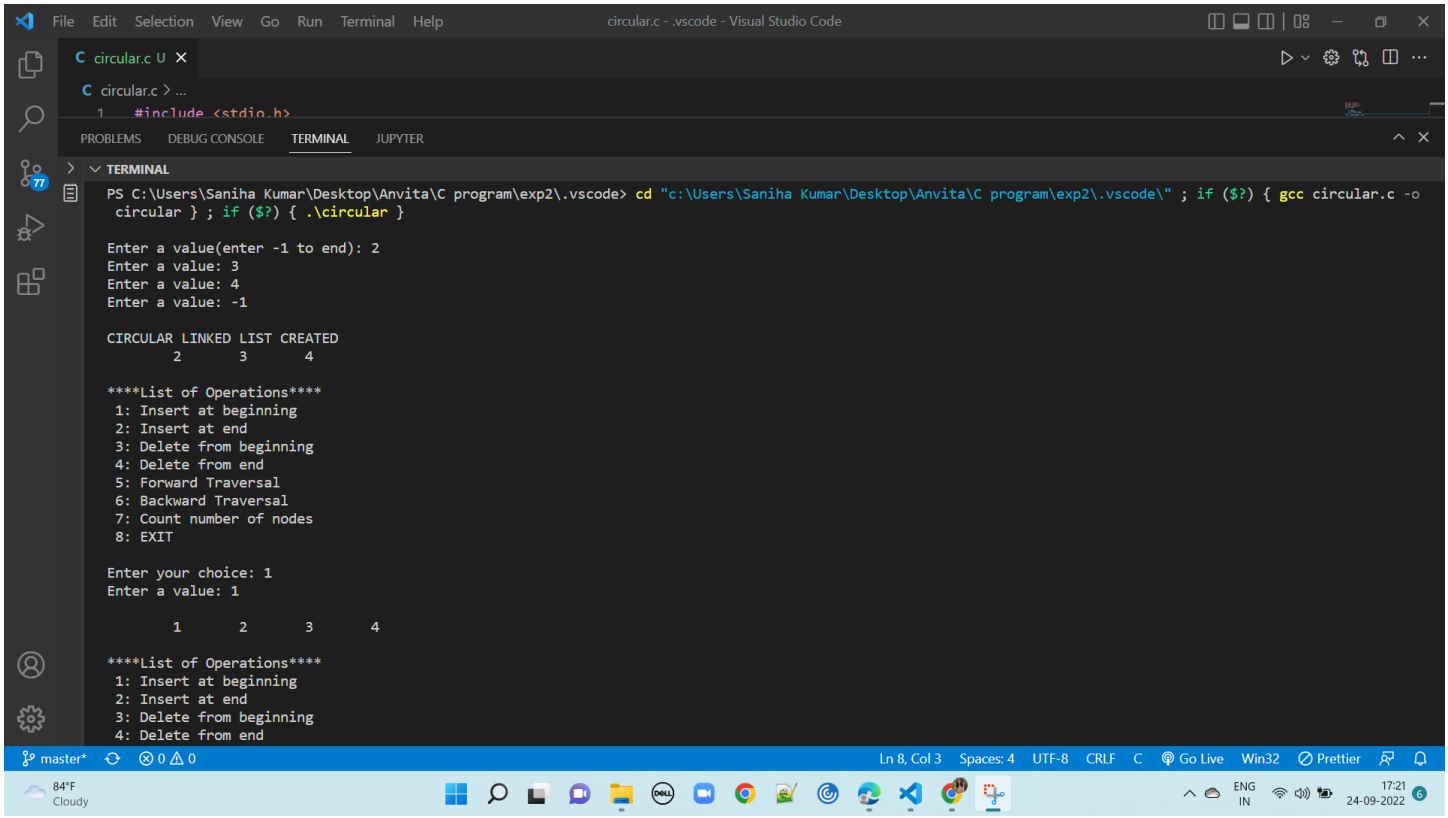
ANVITA KUMAR
C-22
Roll No.: 2104097

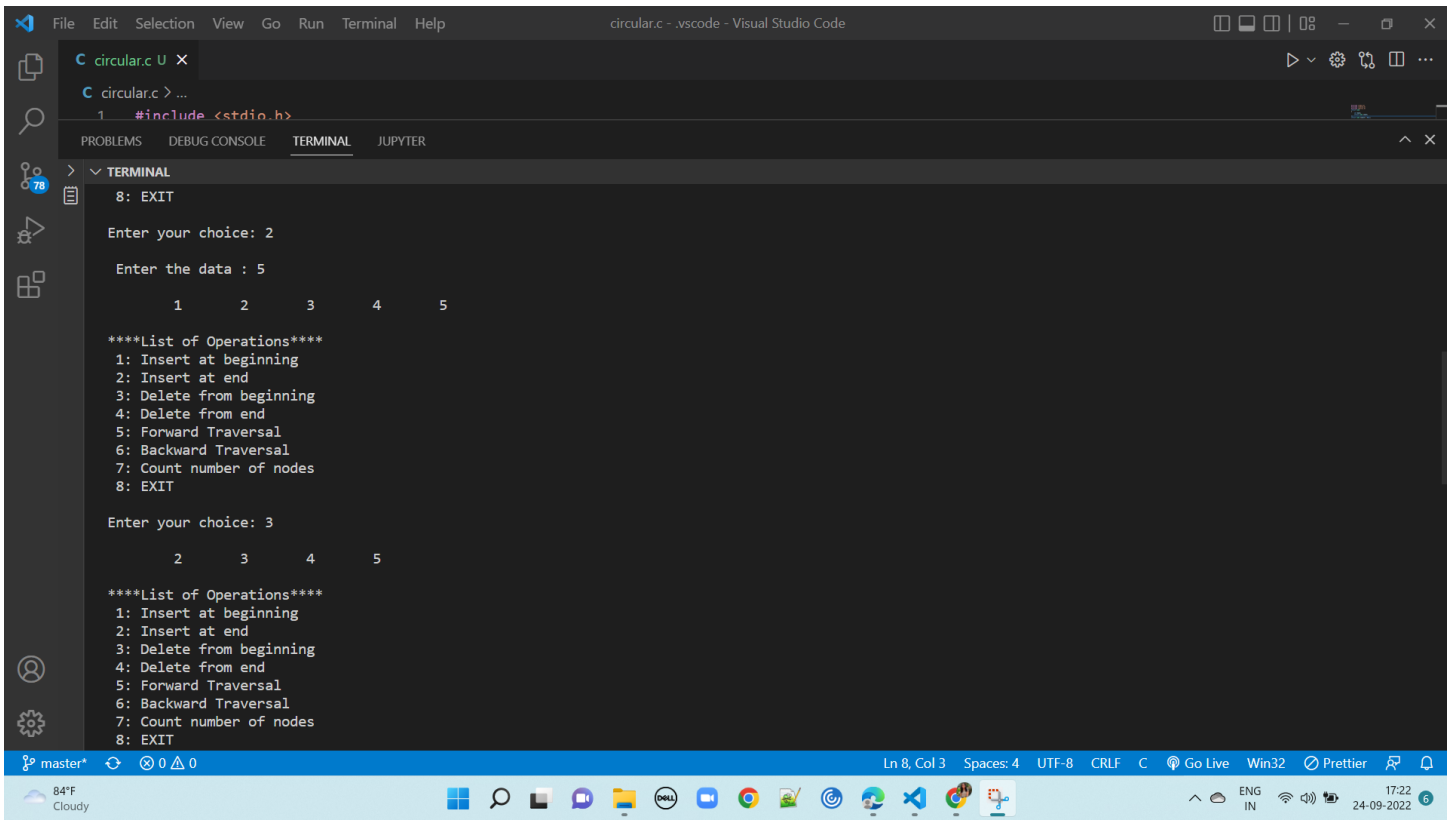
```
   start = ptr->next;
   return start;
}
struct node *DeleteEnd(struct node *start)
{
   struct node *ptr, *preptr;
   ptr = start;
   while (ptr->next != start) {
      preptr = ptr;
      ptr = ptr->next;
   }
   preptr->next = ptr->next;
   free(ptr);
   return start;
}
struct node *ForwardTraversal(struct node *start)
{
    struct node *ptr;
    ptr = start;
    if (ptr == NULL) {
       printf("\tEmpty List!");
    }
    else
    {
       printf("\n");
       while (ptr->next != start) {
          printf("\t%d", ptr->data);
          ptr = ptr->next;
       }
       printf("\t%d", ptr->data);
    }
```

ANVITA KUMAR
C-22
Roll No.: 2104097

```
    return start;
}
struct node *BackwardTraversal(struct node *start)
{
   struct node* prev = start;
   struct node *current = start;
   struct node *temp = start;
   current=current->next;
   temp=temp->next->next;
   while (current != start) {
      current->next = prev;
      prev = current;
      current = temp;
      temp = current->next;
   }
   start = prev;
   current->next = start;
}
struct node *Count(struct node *start)
{
   int i=0;
   struct node *current = start;
   do {
      start = start->next;
      i++;
   } while (current != start);
   printf("Number of nodes in the list: %d", i);
}
```

ANVITA KUMAR
C-22
Roll No.: 2104097





ANVITA KUMAR
C-22
Roll No.: 2104097