#include <stdio.h>

#include <malloc.h>

struct node

{

int data;

struct node \*next;

};

struct node \*start = NULL;

struct node \*create\_cll(struct node \*);

struct node \*display(struct node \*);

struct node \*insert\_beg(struct node \*);

struct node \*insert\_end(struct node \*);

struct node \*delete\_beg(struct node \*);

struct node \*delete\_end(struct node \*);

struct node \*delete\_after(struct node \*);

struct node \*delete\_list(struct node \*);

int main()

{

int option;

do

{

printf("\n\n \*\*\*\*\*MAIN MENU \*\*\*\*\*");

printf("\n 1: Create a list");

printf("\n 2: Display the list");

printf("\n 3: Add a node at the beginning");

printf("\n 4: Add a node at the end");

printf("\n 5: Delete a node from the beginning");

printf("\n 6: Delete a node from the end");

printf("\n 7: Delete a node after a given node");

printf("\n 8: Delete the entire list");

printf("\n 9: EXIT");

printf("\n\n Enter your option : ");

scanf("%d", &option);

switch(option)

{

case 1: start = create\_cll(start);

printf("\n CIRCULAR LINKED LIST CREATED");

break;

case 2: start = display(start);

break;

case 3: start = insert\_beg(start);

break;

case 4: start = insert\_end(start);

break;

case 5: start = delete\_beg(start);

break;

case 6: start = delete\_end(start);

break;

case 7: start = delete\_after(start);

break;

case 8: start = delete\_list(start);

printf("\n CIRCULAR LINKED LIST DELETED");

break;

}

}while(option !=9);

return 0;

}

struct node \*create\_cll(struct node \*start)

{

struct node \*new\_node, \*ptr;

int num;

printf("\n Enter –1 to end");

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

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

{ ptr = start;

while(ptr -> next != start)

ptr = ptr -> next;

ptr -> next = new\_node;

new\_node -> next = start;

}

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

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 \*insert\_beg(struct node \*start)

{

struct node \*new\_node, \*ptr;

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;

start = new\_node;

return start;

}

struct node \*insert\_end(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 \*delete\_beg(struct node \*start)

{

struct node \*ptr;

ptr = start;

while(ptr -> next != start)

ptr = ptr -> next;

ptr -> next = start -> next;

free(start);

start = ptr -> next;

return start;

}

struct node \*delete\_end(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 \*delete\_after(struct node \*start)

{

struct node \*ptr, \*preptr;

int val;

printf("\n Enter the value after which the node has to deleted : ");

scanf("%d", &val);

ptr = start;

preptr = ptr;

while(preptr -> data != val)

{

preptr = ptr;

ptr = ptr -> next;

}

preptr -> next = ptr -> next;

if(ptr == start)

start = preptr -> next;

free(ptr);

return start;

}

struct node \*delete\_list(struct node \*start)

{

struct node \*ptr;

ptr = start;

while(ptr -> next != start)

start = delete\_end(start);

free(start);

return start;

}