#include <stdio.h>

#include <malloc.h>

struct node

{

struct node \*next;

int data;

struct node \*prev;

};

struct node \*start = NULL;

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

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

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

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

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

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

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

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

struct node \*delete\_before(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: Add a node before a given node");

printf("\n 6: Add a node after a given node");

printf("\n 7: Delete a node from the beginning");

printf("\n 8: Delete a node from the end");

printf("\n 9: Delete a node before a given node");

printf("\n 10: Delete a node after a given node");

printf("\n 11: Delete the entire list");

printf("\n 12: EXIT");

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

scanf("%d", &option);

switch (option)

{

case 1:

start = create\_ll(start);

printf("\n DOUBLY 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 = insert\_before(start);

break;

case 6:

start = insert\_after(start);

break;

case 7:

start = delete\_beg(start);

break;

case 8:

start = delete\_end(start);

break;

case 9:

start = delete\_before(start);

break;

case 10:

start = delete\_after(start);

break;

case 11:

start = delete\_list(start);

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

break;

}

} while (option != 12);

return 0;

}

struct node \*create\_ll(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)

{

if (start == NULL)

{

new\_node = (struct node \*)malloc(sizeof(struct node));

new\_node->prev = NULL;

new\_node->data = num;

new\_node->next = NULL;

start = new\_node;

}

else

{

ptr = start;

new\_node = (struct node \*)malloc(sizeof(struct node));

new\_node->data = num;

while (ptr->next != NULL)

ptr = ptr->next;

ptr->next = new\_node;

new\_node->prev = ptr;

new\_node->next = NULL;

}

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

scanf("%d", &num);

}

return start;

}

struct node \*display(struct node \*start)

{

struct node \*ptr;

ptr = start;

while (ptr != NULL)

{

printf("\t %d", ptr->data);

ptr = ptr->next;

}

return start;

}

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

{

struct node \*new\_node;

int num;

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

scanf("%d", &num);

new\_node = (struct node \*)malloc(sizeof(struct node));

new\_node->data = num;

start->prev = new\_node;

new\_node->next = start;

new\_node->prev = NULL;

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 != NULL)

ptr = ptr->next;

ptr->next = new\_node;

new\_node->prev = ptr;

new\_node->next = NULL;

return start;

}

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

{

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

int num, val;

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

scanf("%d", &num);

printf("\n Enter the value before which the data has to be inserted:");

scanf("%d", &val);

new\_node = (struct node \*)malloc(sizeof(struct node));

new\_node->data = num;

ptr = start;

while (ptr->data != val)

ptr = ptr->next;

new\_node->next = ptr;

new\_node->prev = ptr->prev;

ptr->prev->next = new\_node;

ptr->prev = new\_node;

return start;

}

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

{

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

int num, val;

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

scanf("%d", &num);

printf("\n Enter the value after which the data has to be inserted:");

scanf("%d", &val);

new\_node = (struct node \*)malloc(sizeof(struct node));

new\_node->data = num;

ptr = start;

while (ptr->data != val)

ptr = ptr->next;

new\_node->prev = ptr;

new\_node->next = ptr->next;

ptr->next->prev = new\_node;

ptr->next = new\_node;

return start;

}

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

{

struct node \*ptr;

ptr = start;

start = start->next;

start->prev = NULL;

free(ptr);

return start;

}

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

{

struct node \*ptr;

ptr = start;

while (ptr->next != NULL)

ptr = ptr->next;

ptr->prev->next = NULL;

free(ptr);

return start;

}

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

{

struct node \*ptr, \*temp;

int val;

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

scanf("%d", &val);

ptr = start;

while (ptr->data != val)

ptr = ptr->next;

temp = ptr->next;

ptr->next = temp->next;

temp->next->prev = ptr;

free(temp);

return start;

}

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

{

struct node \*ptr, \*temp;

int val;

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

scanf("%d", &val);

ptr = start;

while (ptr->data != val)

ptr = ptr->next;

temp = ptr->prev;

if (temp == start)

start = delete\_beg(start);

else

{

ptr->prev = temp->prev;

temp->prev->next = ptr;

}

free(temp);

return start;

}

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

{

while (start != NULL)

start = delete\_beg(start);

return start;

}