#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*start = NULL;

void insert\_at\_begin(int);

void insert\_at\_end(int);

void traverse();

void delete\_from\_begin();

void delete\_from\_end();

int count = 0;

int main () {

int input, data,size;

printf("WELCOME!!!\nDesign your own pizza!\n");

printf("Enter the size of your pizza\n1.Small Rs.200\n2.Medium Rs.300\n3.Large Rs.500\n ");

scanf("%d",&size);

for (;;) {

printf("1. Select a base\n");

printf("2. Select a topping\n");

printf("3. See your current order.\n");

printf("4. Delete the base.\n");

printf("5. Delete the topping.\n");

printf("6. Exit\n");

scanf("%d", &input);

if (input == 1) {

printf("Enter the base\n1.Regular\n2.Thin Crust\n3.Cheese Burst\n");

scanf("%d", &data);

insert\_at\_begin(data);

}

else if (input == 2) {

printf("Enter a topping:\n4.Corn\n5.Capcicum\n6.Mushroom\n");

scanf("%d", &data);

insert\_at\_end(data);

}

else if (input == 3)

traverse();

else if (input == 4)

delete\_from\_begin();

else if (input == 5)

delete\_from\_end();

else if (input == 6)

break;

else

printf("Please enter valid input.\n");

}

return 0;

}

void insert\_at\_begin(int x) {

struct node \*t;

t = (struct node\*)malloc(sizeof(struct node));

count++;

if (start == NULL) {

start = t;

start->data = x;

start->next = NULL;

return;

}

t->data = x;

t->next = start;

start = t;

}

void insert\_at\_end(int x) {

struct node \*t, \*temp;

t = (struct node\*)malloc(sizeof(struct node));

count++;

if (start == NULL) {

start = t;

start->data = x;

start->next = NULL;

return;

}

temp = start;

while (temp->next != NULL)

temp = temp->next;

temp->next = t;

t->data = x;

t->next = NULL;

}

void traverse() {

struct node \*t;

t = start;

if (t == NULL) {

printf("Linked list is empty.\n");

return;

}

printf("There are %d items in your cart.\n", count);

while (t->next!= NULL)

{

if(t->data==1)

          {

            printf("Base:Regular");

          }

          if(t->data==2)

          {

            printf("Base:Thin Crust\n ");

          }

          if (t->data==3)

          {

printf("Base:Cheese Burst\n");

}

if (t->data==4)

          {

printf("Topping:Corn\n");

}

if (t->data==5)

          {

printf("Topping:Capcicum\n");

}

if (t->data==6)

          {

printf("Topping:Mushroom\n");

}

        t = t->next;

}

if(t->data==1)

          {

            printf("Base:Regular");

          }

          if(t->data==2)

          {

            printf("Base:Thin Crust\n ");

          }

          if (t->data==3)

          {

printf("Base:Cheese Burst\n");

}

if (t->data==4)

          {

printf("Topping:Corn\n");

}

if (t->data==5)

          {

printf("Topping:Capcicum\n");

}

if (t->data==6)

          {

printf("Topping:Mushroom\n");

}

}

void delete\_from\_begin() {

struct node \*t;

int n;

if (start == NULL) {

printf("Linked list is already empty.\n");

return;

}

n = start->data;

t = start->next;

free(start);

start = t;

count--;

printf("%d deleted from beginning successfully.\n", n);

}

void delete\_from\_end() {

struct node \*t, \*u;

int n;

if (start == NULL) {

printf("Linked list is already empty.\n");

return;

}

count--;

if (start->next == NULL) {

n = start->data;

free(start);

start = NULL;

printf("%d deleted from end successfully.\n", n);

return;

}

t = start;

while (t->next != NULL) {

u = t;

t = t->next;

}

n = t->data;

u->next = NULL;

free(t);

printf("%d deleted from end successfully.\n", n);

}