#include <stdio.h>

#include <stdlib.h>

struct node {

int x,y,z;

struct node \*next;

};

struct node \*create\_node(int);

struct node \*get\_tail(struct node \*);

void add\_node(int,struct node \*);

void release\_list(struct node \*);

struct node \*get\_node(int,int,struct node \*);

void insert\_node(struct node \*,int,struct node \*);

void delete\_node(int,struct node \*);

void print\_list(int,struct node \*);

int main(){

struct node \*head;

struct node \*element;

head = create\_node(100);

add\_node(200,head);

add\_node(400,head);

add\_node(500,head);

printf("\n");

element = create\_node(300);

printf("\n");

insert\_node(element,2,head);

printf("\n");

printf("挿入後の単方向のリスト");

print\_list(0,head);

printf("\n");

delete\_node(2,head);

printf("\n");

printf("削除後の単方向のリスト");

print\_list(0,head);

printf("\n");

release\_list(head);

return 0;

}

struct node \*create\_node(int data){

struct node \*element;

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

element->x = data;

element->y = data;

element->z = data;

element->next = NULL;

printf("x = %d created\n",element->x);

printf("y = %d created\n",element->y);

printf("z = %d created\n",element->z);

return element;

}

struct node \*get\_tail(struct node \*element){

if(element->next == NULL){

return element;

}

else{

return get\_tail(element->next);

}

}

void add\_node(int data,struct node \*head){

struct node \*element;

struct node \*tail;

element = create\_node(data);

tail = get\_tail(head);

tail->next = element;

}

void release\_list(struct node \*element){

if(element->next != NULL){

release\_list(element->next);

}

else{

free(element);

printf("x = %d released\n",element->x);

printf("y = %d released\n",element->y);

printf("z = %d released\n",element->z);

}

}

struct node \*get\_node(int i,int index,struct node \*element){

if(index == i){

return element;

}

else{

return get\_node(i+1,index,element->next);

}

}

void insert\_node(struct node \*element,int index,struct node \*head){

struct node \*prev;

struct node \*next;

prev = get\_node(0,index-1,head);

next = prev->next;

prev->next = element;

element->next = next;

printf("x = %d inserted\n",element->x);

printf("y = %d inserted\n",element->y);

printf("z = %d inserted\n",element->z);

}

void delete\_node(int index,struct node \*head){

struct node \*prev;

struct node \*element;

struct node \*next;

prev = get\_node(0,index-1,head);

element=prev->next;

next = element->next;

prev->next = next;

printf("x = %d deleted\n",element->x);

printf("y = %d deleted\n",element->y);

printf("z = %d deleted\n",element->z);

free(element);

}

void print\_list(int i,struct node \*element){

printf("x[%d] = %d\n",i,element->x);

printf("y[%d] = %d\n",i,element->y);

printf("z[%d] = %d\n",i,element->z);

if(element->next != NULL){

print\_list(i+1,element->next);

}

}

**実行結果**

x = 100 created

y = 100 created

z = 100 created

x = 200 created

y = 200 created

z = 200 created

x = 400 created

y = 400 created

z = 400 created

x = 500 created

y = 500 created

z = 500 created

x = 300 created

y = 300 created

z = 300 created

x = 300 inserted

y = 300 inserted

z = 300 inserted

挿入後の単方向のリストx[0] = 100

y[0] = 100

z[0] = 100

x[1] = 200

y[1] = 200

z[1] = 200

x[2] = 300

y[2] = 300

z[2] = 300

x[3] = 400

y[3] = 400

z[3] = 400

x[4] = 500

y[4] = 500

z[4] = 500

x = 300 deleted

y = 300 deleted

z = 300 deleted

削除後の単方向のリストx[0] = 100

y[0] = 100

z[0] = 100

x[1] = 200

y[1] = 200

z[1] = 200

x[2] = 400

y[2] = 400

z[2] = 400

x[3] = 500

y[3] = 500

z[3] = 500

x = 0 released

y = 0 released

z = 0 released