代码：

# include "stdio.h"

# include "stdlib.h"

//链式串的头文件

typedef struct node

{

char data;

struct node \*next;

}linkstrnode;

//链式串的创建

linkstrnode \*strcreate(linkstrnode \*S)

{

char ch;

linkstrnode \*p,\*r = NULL;

S = NULL;

while((ch = getchar()) != '\n')

{

p = (linkstrnode \*)malloc(sizeof(linkstrnode));

p->data = ch;

if(S == NULL)

S = p;

else

r->next = p;

r = p;

}

if(r != NULL)

r->next = NULL;

return S;

}

//字符串插入运算

linkstrnode \*strinsert(linkstrnode \*s,int i,linkstrnode \*T)

{

int k = 1;

linkstrnode \*p = s,\*q = NULL;

while(p && k < i - 1) /\*用p查找S中第i - 1个元素的位置\*/

{

p = p->next;

k ++;

}

if(!p)

printf("error!\n"); /\*第i - 1个元素不存在\*/

else

{

q = T;

while(q && q->next) /\*用q查找T中最后一个元素的位置\*/

q = q->next;

q->next = p->next; /\*将T连接到S中的第i个位置上\*/

p->next = T;

}

return s;

}

//链式串的删除

linkstrnode \*strdelete(linkstrnode \*S,int i,int len)

{

int k = 1;

linkstrnode \*p = S,\*q = NULL,\*r = NULL;

//用p查找s的第i个元素，q始终跟踪p的前驱

while(p && k < i)

{

q = p;

p = p->next;

k ++;

}

//S的第i个元素不存在，则出错

if(!p)

printf("error!\n");

else

{

k = 1;

//p从第i个元素开始查找长度为len字串的最后元素

while(k < len && p)

{

p = p->next;

k ++;

}

if(!p)

printf("error2\n");

else

{

//被删除的字串位于s的最前面

if(!q)

{

r = S;

S = p->next;

}

//被删除的字串位于s的中间或最后的情形

else

{

r = q->next;

q->next = p->next;

}

p->next = NULL;

//回收被删除的字串的空间

while(r != NULL)

{

p = r;

r = r->next;

free(p);

}

}

}

return S;

}

void strcompare(linkstrnode \*st1,linkstrnode \*st2)

{

int i = 0;

if(st1 && st2 && st1->data == st2->data)

{

st1 = st1->next;

st2 = st2->next;

}

if((st1 && !st2) && (st1 && st2 && st1->data > st2->data) )

printf("第一个字符串大于第二个字符串!\n");

else

printf("第二个字符串大于第一个字符串!\n");

}

//字符串连接

linkstrnode \*strconcat(linkstrnode \*s1,linkstrnode \*s2)

{

linkstrnode \*p = NULL,\*st = s1;;

if(!s1)

{

s1 = s2;

return NULL;

}

else

if(s2)

{

p = s1;

while(p->next)

p = p->next;

p->next = s2;

}

return st;

}

//求字串的运算

linkstrnode \*substring(linkstrnode \*S,int i,int len)

{

int k = 1;

linkstrnode \*p = S,\*q = NULL,\*r = NULL,\*t = NULL;

linkstrnode \*s = NULL;

//用p查找s中的第i个字符串

while(p && k < i)

{

p = p->next;

k ++;

}

if(!p)

{

printf("error1\n");

return NULL;

}

//处理S中的第i个字符不存在的情况

else

{

r = (linkstrnode \*)malloc(sizeof(linkstrnode));

r->data = p->data;

r->next = NULL;

k = 1;

q = r; //用q始终指向子串的最后一个字符串的位置

while(p->next && k < len) //取长度为len的字串

{

p = p->next;

k ++;

t = (linkstrnode \*) malloc (sizeof(linkstrnode));

t->data = p->data;

q->next = t;

q = t;

}

if(k < len)

{

printf("error2\n");

return NULL;

}

//处理子串的尾部

else

{

q->next = NULL;

return r;

}

}

}

void display(linkstrnode \*s)

{

linkstrnode \*p=s;

while(p)

{

printf("%c",p->data);

p=p->next;

}

printf("\n");

}

int main()

{

int m = 0,n = 0;

int i = 0,j = 0;

linkstrnode \*st1 = NULL,\*st2 = NULL,\*st3 = NULL;

printf("请输入一个字符串：\n");

st1 = strcreate(st1);

printf("输入的字符串为：\n");

display(st1);

printf("请输入想要插入的字符串：\n");

st2 = strcreate(st2);

printf("请输入想要插入的位置：\n");

scanf("%d",&m);

st1 = strinsert(st1,m,st2);

printf("在st1的第%d个位置上插入st2后的字符串为：\n",m);

display(st1);

printf("请输入想要删除的字符串的位置以及长度：\n");

scanf("%d%d",&i,&j);

getchar();

st1 = strdelete(st1,i,j);

printf("st1在位置%d处删除长度为%d的字符串后的字符串为:\n",i,j);

display(st1);

printf("请输入想要连接的字符串：\n");

st3 = strcreate(st3);

st1 = strconcat(st1,st3);

printf("连接后的字符串为：\n");

display(st1);

printf("请输入想要求得字串得位置及长度\n");

i = 0,j = 0;

scanf("%d%d",&i,&j);

getchar();

st1 = substring(st1,i,j);

printf("求得的字符串为：\n");

display(st1);

st2 = NULL;

printf("请输入一个字符串，进行比较：\n");

st2 = strcreate(st2);

printf("结果为：\n");

strcompare(st1,st2);

return 0;

}

结果演示：

