1.编写算法replace(string &S, string t, string v)，将字符串S中的所有子串t用字符串v替换。

算法思想：可以用数组，也可以用动态链表，有兴趣的同学都可以试试这里给出的是用动态链表的算法，这里通过对Index、Delete和Insert函数的调用，完成将串S中出现的子串T用串v替代

算法：

```
/*用动态链表实现：*/
#include <stdio.h>
#include <stdlib.h>

#define MAXSTRLEN 255

typedef struct
{
  char *ch;
  int length;
} HString;

void StrAssign(HString &S, char *chars)
{
  //串赋值
  char*    c;
  int       i, j;

  if (!S.ch)
    free(S.ch);

  for (i = 0, c = chars; *c; c++, i++ );

  if (!i)
  {
    S.ch = NULL;
    S.length = 0;
  }
  else
  {
    if (!(S.ch = (char*)(malloc(sizeof(char) * i))))
      return;
```

```
    for (j = 0; j < i; j++)
      S.ch[j] = chars[j];
  }
  S.length = i;

}

void Display_String(HString S)
{//串显示
  if (S.ch == NULL)
    return;
  int i;
  for(i = 0; i < S.length; i++)
    printf("%c", S.ch[i]);
  printf("\n");
}

int Index(HString S, HString T,int Pos)
{//在串S中扫描子串T的位置值，如不存在子串T返回0
  int clientLen = 0;
  char * Tclient = T.ch;

  if (Pos >= S.length)
    return -1;
  char * Sclient = S.ch;
  while ((Sclient - S.ch) <= S.length)
  {
    while(*(Tclient) == *(Sclient + Pos))
    {
      if ((Tclient - T.ch) < S.length)
        return Pos;
      Tclient++;
      clientLen++;
      Sclient++;
    }
    Sclient = Sclient - clientLen;
    Pos++;
  }

  return -1;
}
void Delete(HString &S,int pos,int len)
{//在串S中删去从pos位置开始的len个字符
```

```
  int i;

  for (i = 0; i < (S.length - pos); i++)
    S.ch[pos + i] = S.ch[pos + i + len];
  S.length -= len;
}

void Insert(HString &S,int &pos,HString T)
{//在串S的pos位置插入子串T
  S.ch = (char *)realloc(S.ch, T.length + S.length);
  S.length += T.length;
  int i;

  if(pos    != S.length)
  {

    for (i = 0; i < S.length - pos; i++)
    {
      *(S.ch + S.length + T.length-1 - i) = *(S.ch + S.length -1- i);
    }
  }

  for(i = 0; i < T.length; i++)
  {
    S.ch[pos + i] = T.ch[i];
  }
  pos += T.length;

}

void Replace_SubString(HString &S, HString T,HString v)
{//  通过对Index、Delete和Insert函数的调用，完成将串S中出现的子串T用串v
替代
  int pos = 0;
  int posFlag = -1;

  while (1)
  {
    pos = Index(S, T, pos);

    if (pos < posFlag)
      break;
    posFlag = pos;
```

```
    Delete(S, pos, T.length);
    Insert(S, pos, v);
  }

}


void main()
{
  HString S, T, v;

  StrAssign(S, "ahebhechedhe");

  Display_String(S);

  StrAssign(T, "he");

  Display_String(T);

  StrAssign(v, "hello!");

  Display_String(v);

  Replace_SubString(S,T,v);

  Display_String(S);
}
```

2. 编写一算法计算模式串t在串S中出现的频率。

比较简单，直接给出算法：

```
#include<stdio.h>
main()
{
  char a[101],b[21];
    int i,j,t=0;
    printf("\n请输入A字符串(在100个字符以内):");
    scanf("%s",a);
    printf("\n请输入你想要在A里面查找的字符串(在20个字符以内):");
    scanf("%s",b);
    for(i=0,j=0;a[i]!='\0';i++)
    {
            if(a[i]==b[j])
                {
```

```
                    j++;
                }
            if(b[j]=='\0')
                {
                    ++t;
                    j=0;
                }
    }
    printf("\n字符串 %s 在字符串 %s 里面共出现 %d 次\n",b,a,t);

}
```