Strings - algumas funções úteis...

Profs. André, Salles

Departamento de Informática
Universidade Federal de Viçosa

INF 333 - 2024/1

Exemplo de problema que envolve manipulação de strings

[retirado do capítulo 3 do livro Programming Challenges]

Exemplo: renomeando empresas

Descrição

Empresas tem mudado de nome com mais frequência ultimamente: umas se juntam, outras são compradas, e há ainda as que mudam de nome por questão de marketing.

Um problema é saber o nome atual de uma empresa ao ler documentos antigos. Sua empresa, Digiscam (antes Algorist Technologies), te passou a tarefa de manter um banco de dados com as mudanças de nome e fazer as substituições apropriadas em alguns documentos.

O programa terá como entrada uma lista de mudanças de nome e várias linhas do texto a ser corrigido. Devem ser substituídas apenas ocorrências exatas dos nomes. Haverá no máximo 100 mudanças de nome, e no máximo 1000 caracteres por linha.

Exemplo: renomeando empresas

Exemplo de entrada

```
4
"Anderson Consulting" to "Accenture"
"Enron" to "Dynegy"
"DEC" to "Compaq"
"TWA" to "American"
5
Anderson Accounting begat Anderson Consulting, which
offered advice to Enron before it DECLARED bankruptcy,
which made Anderson
Consulting quite happy it changed its name
in the first place!
```

Exemplo de saída

```
Anderson Accounting begat Accenture, which
offered advice to Dynegy before it CompaqLARED bankruptcy,
which made Anderson
Consulting quite happy it changed its name
in the first place!!
```

Exemplo: renomeando empresas

Estruturas de dados relevantes

```
#define MAXLEN 1001 /* longest possible string */
#define MAXCHANGES 101 /* maximum number of name changes */

typedef char string[MAXLEN];

string mergers[MAXCHANGES][2]; /* store before/after corporate nam
int nmergers; /* the number of different name changes */
```

Exemplo: renomeando empresas

Leitura das mudanças de nome

```
read_quoted_string(char *s)
{
  int i=0; /* counter */
  char c; /* latest character */

  while ((c=getchar()) != '\"') ;

  while ((c=getchar()) != '\"') {
    s[i] = c;
    i = i+1;
  }

  s[i] = '\0';
}
```

Exemplo: renomeando empresas

Leitura das mudanças de nome

```
read_changes()
{
  int i; /* counter */

  scanf("%d\n",&nmergers);
  for (i=0; i<nmergers; i++) {
    read_quoted_string(&(mergers[i][0]));
    read_quoted_string(&(mergers[i][1]));
  }
}
```

Exemplo: renomeando empresas

Procurar um *match*

```
/* Return the position of the first occurrence of the pattern p
in the text t, or -1 if it does not occur. */

int findmatch(char *p, char *t)
{
  int i,j; /* counters */
  int plen, tlen; /* string lengths */

  plen = strlen(p);
  tlen = strlen(t);

  for (i=0; i<=(tlen-plen); i=i+1) {
    j=0;
    while ((j<plen) && (t[i+j]==p[j]))
      j = j+1;
    if (j == plen) return(i);
  }

  return(-1);
```

Exemplo: renomeando empresas

Substituição

```
/* Replace the substring of length xlen starting at position pos i
string s with the contents of string y.
*/

replace_x_with_y(char *s, int pos, int xlen, char *y)
{
  int i; /* counter */
  int slen,ylen; /* lengths of relevant strings */

  slen = strlen(s);
  ylen = strlen(y);

  if (xlen >= ylen)
    for (i=(pos+xlen); i<=slen; i++) s[i+(ylen-xlen)] = s[i];
  else
    for (i=slen; i>=(pos+xlen); i--) s[i+(ylen-xlen)] = s[i];

  for (i=0; i<ylen; i++) s[pos+i] = y[i];
}
```

Exemplo: renomeando empresas

Função principal

```
main()
{
  string s; /* input string */
  char c; /* input character */
  int nlines; /* number of lines in text */
  int i,j; /* counters */
  int pos; /* position of pattern in string */

  read_changes();
  scanf("%d\n",&nlines);
  ...
```

Exemplo: renomeando empresas

Função principal

```
  ...
  for (i=1; i<=nlines; i=i+1) {
    /* read text line */
    j=0;
    while ((c=getchar()) != '\n') {
      s[j] = c;
      j = j+1;
    }
    s[j] = '\0';

    for (j=0; j<nmergers; j=j+1)
      while ((pos=findmatch(mergers[j][0],s)) != -1) {
        replace_x_with_y(s, pos,
        strlen(mergers[j][0]), mergers[j][1]);
      }

    printf("%s\n",s);
  }
}
```

Funções úteis C

```
#include <cstring>
```

[retirados de `http://www.cplusplus.com/reference/clibrary/cstring/`*]*

strncpy

Exemplo

```
int main ()
{
  char str1[]= "To be or not to be";
  char str2[6];
  strncpy (str2,str1,5);
  str2[5]='\0';
  cout << str2;
  return 0;
}
```

Resultado

```
To be
```

- `strncpy` NÃO acrescenta '\0'!!
- a não ser que seja atingido o '\0' durante a cópia...

strstr

Exemplo

```
int main ()
{
  char str[] ="This is a simple string";
  char * pch;
  pch = strstr (str,"simple");
  strncpy (pch,"sample",6);
  cout << str;
  return 0;
}
```

Resultado

```
This is a sample string
```

strtok

Exemplo

```
int main ()
{
  char str[] ="- This, a sample string.";
  char * pch;
  pch = strtok (str," ,.-");
  while (pch != NULL)
  {
    cout << pch << endl;
    pch = strtok (NULL, " ,.-");
  }
  return 0;
}
```

Resultado

```
This
a
sample
string
```

- Note que o uso dessas funções facilitam bastante a manipulação de strings
- No exemplo dado, a função `replace_x_with_y` pode usar uma chamada apropriada a `strncpy`
- A função `findmatch` pode ser substituída por uma chamada apropriada a `strstr`
- Isso poupa muito tempo de programação, e diminui muito a chance de erros de índice no array de caracteres

Funções úteis C++

```
#include <string>
```

[retirados de `http://www.cplusplus.com/reference/string/string/`*]*

string::iterator, string::begin, string::end

Exemplo

```
int main ()
{
  string str ("Test string");
  string::iterator it;
  for ( it=str.begin() ; it < str.end(); it++ )
    cout << *it;
  return 0;
}
```

Resultado

```
Test string
```

string::compare

Exemplo

```
string str1 ("green apple");
string str2 ("red apple");

if (str1.compare(str2) != 0)
  cout << str1 << " is not " << str2 << "\n";

if (str1.compare(6,5,"apple") == 0)
  cout << "still, " << str1 << " is an apple\n";

if (str2.compare(str2.size()-5,5,"apple") == 0)
  cout << "and " << str2 << " is also an apple\n";
```

Resultado

```
green apple is not red apple
still, green apple is an apple
and red apple is also an apple
```

string::find_first_of

Exemplo

```
...
string str("Replace the vowels in this sentence by asterisks.");
size_t found;

found = str.find_first_of("aeiou");
while (found != string::npos)
{
  str[found] = '*';
  found = str.find_first_of("aeiou",found+1);
}
cout << str << endl;
```

Resultado

```
R*pl*c* th* v*w*ls *n th*s s*nt*nc* by *st*r*sks.
```

string::find_first_not_of

Exemplo

```
...
string str ("look for non-alphabetic characters...");
size_t found;

found = str.find_first_not_of("abcdefghijklmnopqrstuvwxyz ");
if (found != string::npos)
{
  cout << "First non-alphabetic character is " << str[found];
  cout << " at position " << int(found) << endl;
}
```

Resultado

```
First non-alphabetic character is - at position 12
```

string::replace

Exemplo

```
...
string base="this is a test string.";
string str2="n example";
string str3="sample phrase";

// Using positions:                  0123456789*123456789*12345
string str=base;                  // "this is a test string."
str.replace(9,5,str2);            // "this is an example string."
str.replace(19,6,str3,7,6);       // "this is an example phrase."
str.replace(8,10,"just all",6); // "this is just a phrase."
str.replace(8,6,"a short");       // "this is a short phrase."
str.replace(22,1,3,'!');          // "this is a short phrase!!!"
```

string::replace

Exemplo

```
...
string base="this is a short phrase!!!";
string str3="sample phrase";
string str4="useful.";

// Using iterators:                          0123456789*123456789*
string::iterator it = str.begin();    //  ^
str.replace(it,str.end()-3,str3);     // "sample phrase!!!"
str.replace(it,it+6,"replace it",7); // "replace phrase!!!"
it+=8;                                //          ^
str.replace(it,it+6,"is cool");       // "replace is cool!!!"
str.replace(it+4,str.end()-4,4,'o'); // "replace is cooool!!!"
it+=3;                                //             ^
str.replace(it,str.end(),str4.begin(),str4.end());
                                      // "replace is useful."
```

string::substr

Exemplo

```
string str="We think in generalities, but we live in details.";
                                // quoting Alfred N. Whitehead
string str2, str3;
size_t pos;

str2 = str.substr (12,12); // "generalities"

pos = str.find("live");     // posicao de "live" em str
str3 = str.substr (pos);   // pega de "live" até o fim

cout << str2 << ' ' << str3 << endl;
```

Resultado

```
generalities live in details.
```

string::insert

Exemplo

```
string str="to be question";
string str2="the ";
string str3="or not to be";
string::iterator it;

str.insert(6,str2);                 // to be (the )question
str.insert(6,str3,3,4);             // to be (not )the question
str.insert(10,"that is cool",8);    // to be not (that is )the question
str.insert(10,"to be ");            // to be not (to be )that is the que
str.insert(15,1,':');               // to be not to be(:) that is the qu
it = str.insert(str.begin()+5,','); // to be(,) not to be: that is the q
str.insert (str.end(),3,'.');       // to be, not to be: that is the que
str.insert (it+2,str3.begin(),str3.begin()+3); // (or )

cout << str << endl;
```

Resultado

```
to be, or not to be: that is the question...
```

string::erase

Exemplo

```
string str ("This is an example phrase.");
string::iterator it;

str.erase (10,8);
cout << str << endl;          // "This is an phrase."

it=str.begin()+9;
str.erase (it);
cout << str << endl;          // "This is a phrase."

str.erase (str.begin()+5, str.end()-7);
cout << str << endl;          // "This phrase."
```

string::reverse_iterator, string::rbegin, string::rend

Exemplo

```
int main ()
{
  string str ("now step live...");
  string::reverse_iterator rit;

  for ( rit=str.rbegin() ; rit < str.rend(); rit++ )
    cout << *rit;
  return 0;
}
```

Resultado

```
...evil pets won
```

- Note que o uso dessas funções facilitam ainda mais a manipulação de strings
- No exemplo dado, a função `replace_x_with_y` pode ser substituída por uma chamada apropriada a `string::replace`
- A função `findmatch` pode ser substituída por uma chamada apropriada a `string::find`
- Mais uma vez, isso poupa muito tempo de programação, e diminui muito a chance de erros de índice no acesso a strings

stringstream #include <sstream>

- Permite criar um “stream” bidirecional a partir de um string (ou simplesmente “jogando dados” no stream).

- Pode-se escrever no stringstream da mesma forma que no `cout`

```
stringstream ss;
int a = 10, b = 7;
ss << 2*a << " " << b << endl;      //"20 7"
```

- Pode-se ler de um stringstream da mesma forma que do `cin`

```
int x, y;
ss >> x >> b;    //x recebe 20 e b recebe 7
```

stringstream – Exemplos de uso

- Converter string de número em um número

```
string s = "3.14159265";
stringstream ss(s);
double x;
ss >> x;
```

stringstream – Exemplos de uso

- Criar um stringstream a partir de um string

```
string s = "30 40 50";
stringstream ss(s);
```

- Obter um string a partir de um stringstream

```
ss >> s;
```

stringstream – Exemplos de uso

- Criar strings no formato arquivo_1.txt, arquivo_2.txt, ...

```
string v[10];
for(int i=1;i<=10;i++) {
    stringstream ss;
    ss << "arquivo_" << i << ".txt";
    ss >> v[i-1];
}

for(auto s:v)
  cout << s << endl;
```

stringstream – Exemplos de uso

- Ler linhas e criar um vector com cada palavra de cada linha.

```
vector<string> v;
string s, linha;
int cont = 0;

while(getline(cin,linha)) {
  stringstream ss(linha);
  while(ss >> s) {
    v.push_back(s);
    cont++;
  }
}

cout << cont << " palavras" << endl;
```

- Em C tem sprintf e sscanf que fazem algo semelhante

- sprintf “escreve” em um string (array de char)

```
char s[50];
int a = 10, b = 7;
sprintf(s,"%d %d",2*a,b);
```

- sscanf “lê” de um string (array de char)

```
int x, y;
sscanf(s,"%d %d",&x,&y);
```