TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: **KĨ THUẬT LẬP TRÌNH**

HƯỚNG DẪN THỰC HÀNH

TUẦN 03

CÁC KĨ THUẬT TRÊN KIỂU DỮ LIỆU CHUỖI

TP.HCM, ngày 10 tháng 05 năm 2020

MỤC LỤC

1 Khái niệm kiểu dữ liệu chuỗi:

Trong ngôn ngữ C, kiểu chuỗi kí tự được xem là một mảng các kí tự (mảng các char).

Cú pháp: char* s = new char[10 + 1];

Cú pháp trên khai báo chuỗi gồm 10 kí tự + 1 kí tự kết thúc chuỗi (kí tự '\0', đây là qui ước bắt buộc đối với một chuỗi). Chúng ta cũng có thể khai báo chuỗi tĩnh ví dụ char s[11].

2 Các thao tác cơ bản với chuỗi kí tự.

Trong phần này, ta sẽ lần lượt được giới thiệu qua các thao tác cơ bản liên quan bao gồm: xác định độ dài, so sánh chuỗi, sao chép chuỗi con của chuỗi, chèn chuỗi con vào chuỗi, xóa chuỗi con trong chuỗi…

2.1 Thao tác xác định độ dài chuỗi.

Đây có thể xem là thao tác cơ bản nhất trong các thao tác trên chuỗi. Như ta đã biết, do chuỗi có kí tự kết thúc là '\0' nên ta có thể lợi dụng dấu hiệu này để thực hiện đếm số kí tự trong chuỗi.

Dòng	
1	int StringLength(char s[]){
2	int i = 0;
3	while(s[i] != '\0'){
4	i++;
5	}
6	return i;
7	}
8	
9	void main(){
10	char s[] = "Hello World";
11	int length = StringLength(s);
12	}

2.2 Thao tác so sánh chuỗi

Việc so sánh chuỗi ta thực hiện theo thứ tự từ điển. Xem bảng ví dụ bên dưới để hiểu rõ thế nào là thứ tự từ điển.

Ví dụ	**Giải thích**
s_0 = "abc"	Hai kí tự đầu của chuỗi s_0 và s_1 giống nhau. Kí tự thứ 3 (chỉ số 2)

s_1 = “abd” $s_0 < s_1$	của s_1 lớn hơn kí tự thứ 3 của s_0.
s_0 = “abc” s_1 = “abcd” $s_0 < s_1$	Ba kí tự đầu của chuỗi s_0 và s_1 giống nhau. Chuỗi s_1 có thêm kí tự thứ 4 (chỉ số 3) trong khi chuỗi s_0 đã hết.
s_0 = “abc” s_1 = “d” $s_0 < s_1$	Kí tự đầu tiên của chuỗi s_1 lớn hơn kí tự đầu tiên của chuỗi s_0.

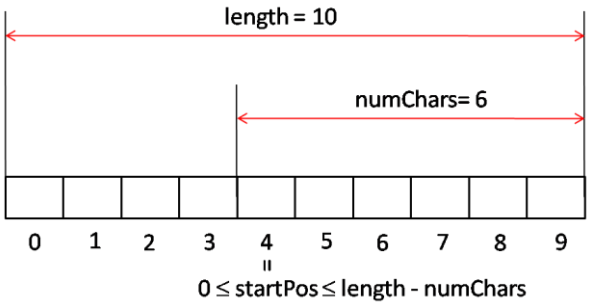
Sau khi đã hiểu thứ tự từ điển ta hãy xem đoạn mã bên dưới minh họa ý tưởng so sánh

Dòng	
0	int CompareString(char* s0, char* s1){
1	int n0 = StringLength(s0);
2	int n1 = StringLength(s1);
3	int i, n = (n0 < n1) ? n0:n1;
4	for(i = 0; i < n; i++){
5	if(s0[i] > s1[i]) return 1;
6	else if (s0[i] < s1[i]) return -1;
7	if(n0 > n) return 1;
8	if(n1 > n) return -1;
9	return 0;
10	}

2.3 Thao tác sao chép chuỗi con của chuỗi.

Trong thao tác này, ta có nhu cầu sao chép một phần chuỗi con trong chuỗi cha ra bên ngoài. Để thực hiện điều này ta cần kiểm tra thật kĩ vị trí bắt đầu sao chép (startPos) cũng như số lượng kí tự được sao chép (numChars).

Ví dụ: Giả sử ta có mảng 10 kí tự (không tính kí tự kết thúc chuỗi ‘\0’). Ta cần sao chép một đoạn chuỗi con dài 6 kí tự (numChars = 6) tại ví trí 4 (startPos = 4).



Hình 1: Sao chép chuỗi con từ chuỗi cha

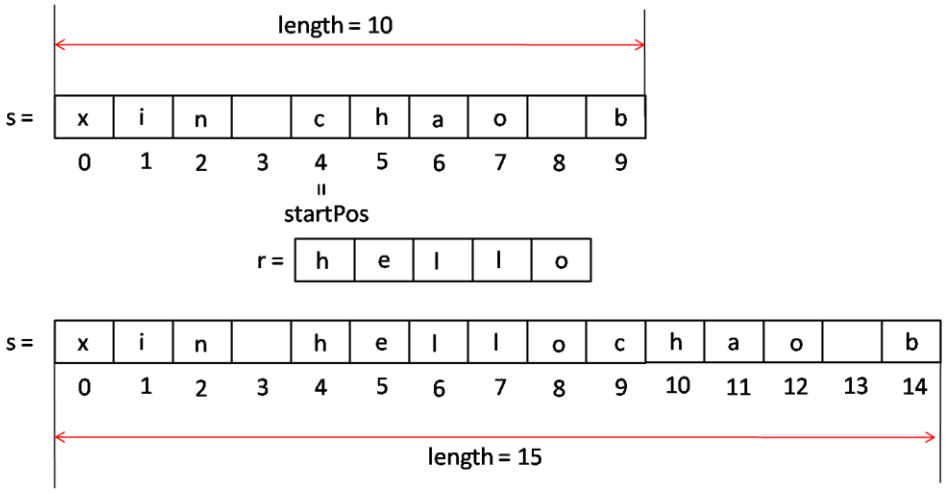
Đoạn mã bên dưới sử dụng hàm strncpy để thực hiện nhanh việc sao chép này.

Dòng	
1	void CopySubString(char* dest, char* src, int startPos, int numChars){
2	strncpy(dest, src + startPos, numChars);
5	dest[numChars] = '\0';
9	}

2.4 Chèn chuỗi con vào chuỗi cha

Ở thao tác này, ta có nhu cầu chèn một chuỗi con vào sau chuỗi cha. Để thực hiện thao tác này ta cần phải kiểm tra rất kĩ vị trí cần chèn.

Ví dụ: ta có **chuỗi s** có 10 kí tự, **chuỗi con r** có 5 kí tự và chèn tại vị trí 3 của s. Như vậy ta cần đẩy các kí tự tại vị trí 3 của s ra 5 vị trí (Xem hình vẽ)



Hình 2: Chèn chuỗi con vào chuỗi cha

Bên dưới là đoạn mã minh họa việc sao chép chuỗi

Dòng	
1	void InsertSubString(char* str, char* substr, int startPos){
2	int length = StringLength(str);
3	int sublength = StringLength(substr);
4	char* temp;
5	if(startPos > length) startPos = length;
6	if(startPos < length){
7	temp = new char[length – startPos + 1];
8	strcpy(temp, str + startPos);
9	strcpy(str + startPos, substr);
10	strcpy(str + startPos + sublength, temp);

11	delete[] temp;
12	}
13	else strcpy(str + startPos, substr);
23	}

3 Bài tập.

Xem kĩ các ví dụ trên, sinh viên hãy xây dựng lại các hàm sau.

- Hàm char* StringNCopy(char* **ChuỗiKQ**, char* **ChuỗiNguồn**, int **SốLượngKíTự**) thực hiện sao chép **SốLượngKíTự** từ **ChuỗiNguồn** vào **ChuỗiKQ**. Hàm này trả về địa chỉ của **ChuỗiKQ**. Sau đó tích hợp hàm vừa xây dựng vào thao tác 2.3 để kiểm tra(thay thế cho hàm strncpy).

- Hàm char* StringCopy(char* **ChuỗiKQ**, char* **ChuỗiNguồn**) thực hiện sao chép nội dung từ **ChuỗiNguồn** sang **ChuỗiKQ**. Hàm trả về địa chỉ chuỗi đích. Sau đó tích hợp hàm vừa xây dựng vào thao tác 2.4 để kiểm tra(thay thế cho hàm strcpy)

- Hàm void DeleteSubString(char* **ChuỗiNguồn**, int **ViTriBatDauXoa**, int **SoLuongKiTuXoa**) thực hiện xóa SoLuongKiTuXoa kí tự trong ChuỗiNguồn tại ViTriBatDauXoa.

- Hàm int FindSubString(char* **ChuỗiCha**, char* **ChuỗiCon**, int **ViTriBatDauTim**) thực hiện tìm vị trí xuất hiện đầu tiên của **ChuỗiCon** trong **ChuỗiCha** tính từ **ViTriBatDauTim**. Nếu không tìm được hàm trả về chỉ số -1.

- Hàm bool IsSubString(char* **ChuỗiCha**, char* **ChuỗiCon**) thực hiện kiểm tra xem **ChuỗiCon** có tồn tại trong **ChuỗiCha** hay không.

- Hàm int CountMatches(char* **ChuỗiCha**, char* **ChuỗiCon**) thực hiện đếm số lần xuất hiện **ChuỗiCon** trong **ChuỗiCha**. Nếu không tồn tại hàm trả về 0.