



성신여자대학교
SUNGSHIN WOMEN'S UNIVERSITY

[팀과제] 문자열 클래스

- 5 조 -

- 학 교 명 : 성신여자대학교
- 과 목 : 객체지향프로그래밍(01)
- 담당교수 : 김종완 교수님
- 제 출 일 : 2015 년 6 월 18 일 목요일
- 학 과 : IT 학부, 융합보안학과
- 학 번 : 201411178, 20131334, 20141229
- 이 름 : 김민정, 이화정, 주혜원

목차

1. 멤버함수 매뉴얼	Page 2
2. 소스코드 + 주석	
String.h	Page 4
StringMemberFunc.h	Page 6
String.cpp	Page 22
3. 각 멤버함수의 실행결과 화면	Page 28

■ 멤버함수 매뉴얼

1. Capacity

- **size** : `size_t size() const;` > 문자열의 길이를 반환
- **length** : `size_t length() const;` > 문자열의 길이를 반환
- **clear** : `void clear();` > 문자열 메모리를 삭제
- **empty** : `bool empty() const;` > 문자열이 비었을 경우 0 또는 1 반환

2. Element access

- **operator[]** : `char operator[](unsigned short offset) const;`
> 상수 객체에 사용되는 상수 변위 연산자
- **at** : `char& at (size_t pos);`
> 문자열의 인덱스 pos 가 가리키는 문자(열)의 참조자 반환
- **back** : `char& back();` > 문자열의 마지막 문자 반환
- **front** : `char& front();` > 문자열의 첫 번째 문자 반환

3. Modifiers

- **operator+=** : `void operator+=(const String&);` > 반환이 없는 현재의 문자열 바꿈
- **append** : `String& append(const String& str);`
> 주어진 문자열에 이어서 자신이 정의한 문자열을 마지막 부분에 추가
- **insert** : `String& insert(size_t pos, const String& str);`
> 주어진 문자열의 pos 번째에 자신이 정의한 문자열을 추가
- **erase** : `String& erase (size_t pos = 0, size_t len = npos);`
> 문자열의 pos 번째 부분에서 삭제할 문자열의 길이 len 만큼을 문자열에서 삭제
- **replace** : `String& replace (size_t pos, size_t len, const String& str);`
> pos 번째 문자열에서 대체될 문자열의 길이 len 을 자신이 정의한
String& str 으로 대체함
- **swap** : `void swap (String& str);` > 주어진 문자열을 자신이 정의한 String& str 로 바꿈
- **lower** : `String& lower();` > 문자열을 소문자로 변환
- **upper** : `String& upper();` > 문자열을 대문자로 변환

4. String operations

- **c_str** : `const char* c_str() const;`
> 문자열의 문자배열의 포인터 값을 return

- **data** : const char* data() const;
 >C++11 표준에서는 c_str 과 동일하여 null 문자를 가진다.
- **copy** : size_t copy (char* s, size_t len, size_t pos = 0) const;
- **find** : size_t find (char c, size_t pos = 0) const;
 size_t find (const String& str, size_t pos = 0) const;
- **find_first_of** : size_t find_first_of (const String& str, size_t pos = 0) const;
 >원 스트링에서 찾는 스트링이 있는 어떠한 문자든지 그 위치를 반환하는 함수
- **find_last_of** : size_t find_last_of (const String& str, size_t pos = npos) const;
 >스트링에서 찾을 문자와 매칭되는 마지막 문자의 위치를 반환
- **substr** : String substr (size_t pos = 0, size_t len = npos) const;
 >원 스트링에서 변수로 입력받은 위치부터 갯수만큼 sub 스트링을 반환
- **compare** : int compare (const String& str) const;
 >원 스트링 a 와 비교할 스트링 b 를 비교해서 전혀 다르다면 0 을,
 >a 가 b 를 포함해 더 많은 스트링이 있다면 +같은 문자의 수,
 >b 가 a 를 포함해 더 많은 스트링이 있다면 -같은 문자의 수를 반환.

5. Member constants

- **npos** : static const size_t npos = -1;

6. 5 조 멤버함수(java string)

- **concat** : String concat(String str);
 >원래 스트링과 변수로 입력받은 스트링을 합친 새로운 스트링을 반환한다.
- **replace** : String replace(char oldChar, char newChar);
 >원 스트링의 oldChar 문자를, newChar 문자로 바꾸어준다.

■ 소스코드 + 주석

String.h

```
1  // [리스트 11.12] 문자열 클래스 사용하기
2  //////////////////////////////////
3  //Homework by team
4  //
5  //String class
6
7  typedef unsigned int size_t;
8
9  // 문자열 클래스
10 class String
11 {
12     public:
13         // 생성자 및 소멸자
14         String();
15         String(const char *const);
16         String(const String &);
17         String (unsigned int n, char c);
18         ~String();
19
20         // 오버로딩 연산자
21         char & operator[](unsigned short offset);
22         char operator[](unsigned short offset) const;
23         String operator+(const String&);
24         void operator+=(const String&);
25         String & operator= (const String &);
26
27         // 일반 접근자
28         unsigned short GetLen()const { return itsLen; }
29         const char * GetString() const { return itsString; }
30
31         //팀과제 멤버함수: 여기에 선언함
32         size_t size() const;
33         size_t length() const;
34         void clear();
```

```

35         bool empty() const;
36         char& at (size_t pos);
37         char& back();
38         char& front();
39
40         String& append(const String& str);
41         String& insert(size_t pos, const String& str);
42         String& erase (size_t pos = 0, size_t len = npos);
43         String& replace (size_t pos, size_t len, const String& str);
44         void swap (String& str);
45         String& lower();
46         String& upper();
47
48         const char* c_str() const;
49         const char* data() const;
50         size_t copy (char* s, size_t len, size_t pos = 0) const;
51         size_t find (char c, size_t pos = 0) const;
52         size_t find (const String& str, size_t pos = 0) const;
53         size_t find_first_of (const String& str, size_t pos = 0) const;
54         size_t find_last_of (const String& str, size_t pos = npos) const;
55         String substr (size_t pos = 0, size_t len = npos) const;
56         int compare (const String& str) const;
57         static const size_t npos = -1 ;
58
59         //JAVA
60         String concat(String str);
61         String replace(char oldChar, char newChar);
62
63
64     private:
65         String (unsigned short); // 전용 생성자
66         char * itsString;
67         unsigned short itsLen;
68     };

```

StringMemberFunc.h

```
1  // [리스트 11.12] 문자열 클래스 사용하기
2  //////////////////////////////////
3  //Homework by team
4  //
5  //Member functions of String class
6
7  typedef unsigned int size_t;
8
9  // 0바이트의 문자열 생성하는 기본 생성자
10 String::String()
11 {
12     itsString = new char[1];
13     itsString[0] = '\0';
14     itsLen=0;
15 }
16
17 // 전용(helper) 생성자
18 // 필요한 만큼의 길이의 문자열을 만드는
19 // 클래스 메소드에 의해서만 사용됨
20 String::String(unsigned short len)
21 {
22     itsString = new char[len+1];
23     for (unsigned short i = 0; i<=len; i++)
24         itsString[i] = '\0';
25     itsLen=len;
26 }
27
28 // 문자 배열을 문자열로 변환함
29 String::String(const char * const cString)
30 {
31     itsLen = strlen(cString);
32     itsString = new char[itsLen+1];
33     for (unsigned short i = 0; i<itsLen; i++)
34         itsString[i] = cString[i];
35     itsString[itsLen]='\0';
```

```

36 }
37
38 // 복사 생성자
39 String::String (const String & rhs)
40 {
41     itsLen=rhs.GetLen();
42     itsString = new char[itsLen+1];
43     for (unsigned short i = 0; i<itsLen;i++)
44         itsString[i] = rhs[i];
45     itsString[itsLen] = '\0';
46 }
47
48 //6번 fill 생성자로 n개의 c를 갖는 스트링 객체를 생성하는 생성자
49 String::String (unsigned int n, char c)
50 {
51     itsString = new char[n]; //캐릭터 포인터 itsString에 n개의 char배열을 할당한다.
52     for(unsigned int i=0; i<n ; i++) {itsString[i] = c;} //for문을 통해 n개의 c를 배열에 넣는다.
53     itsString[n] = '\0'; //스트링의 마지막에 널문자를 넣어준다.
54     itsLen = n ; //길이는 n 그대로이다.
55 }
56
57
58 // 소멸자, 메모리 방출함
59 String::~String()
60 {
61     delete [] itsString;
62     itsLen = 0;
63 }
64
65 // 대입 연산자, 기존 메모리 방출함
66 // 그리고 문자열과 길이를 복사
67 String& String::operator=(const String & rhs)
68 {
69     if (this == &rhs)
70         return *this;
71     delete [] itsString;
72     itsLen=rhs.GetLen();

```



```

73     itsString = new char[itsLen+1];
74     for (unsigned short i = 0; i<itsLen;i++)
75         itsString[i] = rhs[i];
76     itsString[itsLen] = '\0';
77     return *this;
78 }
79
80 // 비상수 변위 연산자
81 // 문자의 참조자 반환하여
82 // 나중에 바꿀 수 있게 함
83 char & String::operator[](unsigned short offset)
84 {
85     if (offset > itsLen)
86         return itsString[itsLen-1];
87     else
88         return itsString[offset];
89 }
90
91 // 상수 객체에 사용되는
92 // 상수 변위 연산자
93 char String::operator[](unsigned short offset) const
94 {
95     if (offset > itsLen)
96         return itsString[itsLen-1];
97     else
98         return itsString[offset];
99 }
100
101 // 현재의 문자열에 rhs를 추가하여
102 // 새로운 문자열을 만듦
103 String String::operator+(const String& rhs)
104 {
105     unsigned short i,totalLen = itsLen + rhs.GetLen();
106     String temp(totalLen);
107     for (i = 0; i<itsLen; i++)
108         temp[i] = itsString[i];
109     for (unsigned short j = 0; j<rhs.GetLen(); j++, i++)

```

```

110         temp[i] = rhs[j];
111     temp[totalLen]='W0';
112     return temp;
113 }
114
115 // 반환이 없는 현재의 문자열 바꿈
116 void String::operator+=(const String& rhs)
117 {
118     unsigned short rhsLen = rhs.GetLen();
119     unsigned short i,totalLen = itsLen + rhsLen;
120     String temp(totalLen);
121     for (i = 0; i<itsLen; i++)
122         temp[i] = itsString[i];
123     for (unsigned short j = 0; j<rhs.GetLen(); j++, i++)
124         temp[i] = rhs[i-itsLen];
125     temp[totalLen]='W0';
126     *this = temp;
127 }
128
129 //////////////////////////////////////////////////
130 // Implement member functions here!!!
131 //
132
133 //문자열의 길이 반환
134 //string::length() 함수와 유사
135 size_t String::size() const
136 {
137     return itsLen;
138 }
139
140 //문자열의 길이 반환
141 //string::size() 함수와 유사
142 size_t String::length() const
143 {
144     unsigned short count = 0, i = 0;
145     //반복문의 조건을 문자열 배열이 null을 만나면 빠져나오도록 함
146     /*(itsString + i) != 'W0'으로 해도 무관

```

```

147     while(itsString[i] != '\0'){
148         i++;
149         count++; //반복문의 조건을 충족할 때까지 문자열 길이를 카운트
150     }
151     return count;
152 }
153
154 //문자열 메모리 삭제
155 //동적 메모리 할당했을 경우 delete로 메모리 삭제
156 void String::clear()
157 {
158     delete [] itsString;
159     itsLen = 0;
160     itsString = new char[1]; //null을 저장할 메모리 할당
161     itsString[0] = '\0';//문자열 메모리 null로 초기화
162 }
163
164 //문자열이 비었을 경우 테스트
165 //반환값은 0 또는 1
166 bool String::empty() const
167 {
168     if(length() == 0)
169         return 1;
170     else
171         return 0;
172 }
173
174 //문자열에서 인덱스 pos가 가리키는 문자의 주소 반환
175 char& String::at (size_t pos) //unsigned int pos
176 {
177     if (pos >= 0 && pos <= itsLen){
178         //인덱스 pos가 가리키는 문자는 문자변수 at_string의 주소
179         char& at_string = itsString[pos];
180         return at_string; //참조자 반환
181     }
182     else
183         return itsString[itsLen];

```

```

184 }
185
186 //문자열의 마지막 문자 반환
187 char& String::back()
188 {
189     return itsString[itsLen - 1];
190     //배열 인덱스를 [length() - 1]로 해도 무관
191     //만약 itsString[itsLen]로 반환 시 '\0' 반환되니 주의
192 }
193
194 //문자열의 첫번째 문자 반환
195 char& String::front()
196 {
197     return itsString[0]; //배열의 첫번째 데이터 반환
198 }
199
200 //주어진 문자열에 이어서 자신이 정의한 문자열을 마지막 부분에 추가
201 //문자열의 마지막 문자열이 됨
202 String& String::append(const String& str)
203 {
204     unsigned short ij=0;
205     unsigned short totalLen = itsLen + str.length();
206     // totalLen = 기존 문자열의 길이 + 새로 추가된 문자열의 길이
207
208     char *temp = new char[itsLen]; // temp 동적할당
209     for (j = 0; j < itsLen; j++) {
210         temp[j] = itsString[j]; // temp에 기존 itsString 저장
211     }
212     itsString = new char[totalLen+1]; // itsString 동적 할당
213     for (i = 0; i < itsLen; i++) {
214         itsString[i] = temp[i]; //itsString에 temp저장.
215     }
216     for (j = 0; j < str.length(); i++, j++) {
217         itsString[i] = str[j]; //itsString에 새로 추가된 str 저장
218     }
219     itsString[totalLen] = '\0'; //null문자 삽입
220     itsLen = totalLen; //itsLen totalLen으로 바꿈.

```

```

221     delete temp; // 동적 할당한 temp 소멸
222     return *this; // 새로운 itsString return
223 }
224
225 //주어진 문자열의 pos번째에 자신이 정의한 문자열을 추가
226 String&String::insert(size_t pos, const String& str){
227     unsigned short i, j, k=0;
228     unsigned short totalLen = itsLen + str.length();
229     //totalLen = 기존 문자열의 길이 + 새로 추가될 문자열의 길이
230
231     char *temp = new char[itsLen]; //temp 동적 할당
232     for (unsigned short j = 0; j < itsLen; j++) {
233         temp[j] = itsString[j]; //temp에 itsString 저장
234     }
235
236     itsString = new char[totalLen+1];
237     for (i = 0; i < pos; i++) {
238         itsString[i] = temp[i]; //pos전까지 itsString에 기존 문자열 저장
239     }
240     for (j = 0; j < str.length(); i++, j++) {
241         itsString[i] = str[j]; //itsString에 새로운 문자열인 str저장.
242     }
243     for (k = pos; k < itsLen; i++, k++) {
244         itsString[i] = temp[k]; // pos이후부터 itsString에 기존 문자열 저장.
245     }
246     itsString[totalLen] = '\0'; //null문자 삽입
247     itsLen = totalLen; // itsString에 totalLen저장.
248     delete temp; //temp 소멸
249
250     return *this; //새로운 itsString return
251 }
252
253 //문자열의 pos번째 부분에서 삭제할 문자열의 길이인 len만큼을 문자열에서 삭제
254 String&String::erase (size_t pos, size_t len )
255 {
256     unsigned short totalLen = itsLen - len; // totalLen = 기존문자열 길이 - 삭제할 문자열의 길이
257     char *temp = new char[itsLen]; //temp 동적 할당

```

```

258     for (unsigned short j = 0; j < itsLen; j++) {
259         temp[j] = itsString[j]; // temp에 itsString 저장.
260     }
261
262     itsString = new char[totalLen+1]; //itsString 동적 할당.
263     for (unsigned short i = 0; i < totalLen; i++) {
264         if (i < pos)
265             itsString[i] = temp[i];
266             //삭제될 위치 전까지는 기존 문자열(temp)와 itsString 일대일 대응
267         else
268             itsString[i] = temp[i+len];
269             //삭제될 위치 이후부터는 itsString[i]=기존문자열에 삭제된 문자열의 길이를
270             더한 만큼의 요소가 대응(temp[len+i])
271     }
272     itsString[totalLen] = '\0'; //null문자 삽입
273     itsLen = totalLen; // itsLen에 totalLen 저장.
274     delete temp; // temp 소멸
275     return *this; // 새로운 itsString return
276 }
277
278 //문자열의 pos번째 부분에서 대체될 문자열의 길이 len을
279 자신이 정의한 String& str으로 대체함
280 String&String::replace (size_t pos, size_t len, const String& str)
281 {
282     unsigned short i,j,k=0;
283     unsigned short totalLen = itsLen -(len-pos)+str.length();
284     //totalLen = 기존문자열길이-(대체될 문자열의 길이) + 대체한 문자열의 길이
285
286     char *temp = new char[itsLen]; //새롭게 temp 할당
287     for (unsigned short j = 0; j < itsLen; j++) {
288         temp[j] = itsString[j]; //temp에 기존 문자열 itsString 저장
289     }
290
291     itsString = new char[totalLen+1]; //itsString 동적 할당.
292     for (i = 0; i < pos; i++) {
293         itsString[i] = temp[i]; //대체될 문자열 이전까지(pos)문자열 itsString에 저장
294     }

```

```

295     for (j = 0; j < str.length(); i++, j++) {
296         itsString[i] = str[j]; // itsString에 새로 대체될 문자열 저장
297     }
298     for (k = len+1; k < itsLen; i++, k++) {
299         itsString[i] = temp[k]; // itsString에 대체된 문자열 이후의 문자열 저장
300     }
301
302     itsString[totalLen] = '\0'; // itsString에 null문자 삽입
303     itsLen = totalLen; //itsLen에 totalLen 저장
304     delete temp; // temp 소멸.
305
306     return *this; // 새로운 itsString 반환
307 }
308
309 //주어진 문자열을 자신이 정의한 String& str로 바꿈
310 void String::swap (String& str)
311 {
312     char *temp = new char[itsLen]; //temp 새로 할당
313     for(unsigned short i=0;i<itsLen;i++){
314         temp[i]=str[i]; // temp에 바꿀 문자열 저장
315     }
316     for(unsigned short i=0;i<itsLen;i++){
317         str[i]=itsString[i]; //str에 기존 문자열 저장
318     }
319     for(unsigned short i=0;i<itsLen;i++){
320         itsString[i]=temp[i]; //itsString에 temp 저장(바꿀 문자열)
321     }
322     delete temp;
323 }
324
325 //문자열을 소문자로 변환
326 String&String::lower()
327 {
328     char *thatsString = new char[itsLen]; //itsString 새로 할당
329     for(unsigned short i=0;i<itsLen;i++){
330         if(('A' <= itsString[i] && (itsString[i] <= 'Z')) //A~Z사이일 때 a~z로 바꾼다(아스키코드)
331             itsString[i] += 32;

```

```

332     }
333     return *this; // 새로운 itsString return
334     delete thatsString; // 할당한 itsString 소멸
335 }
336
337 //문자열을 대문자로 변환
338 String&String::upper()
339 {
340     char *thatsString = new char[itsLen]; //itsString 새로 할당
341     for(unsigned short i=0; i<itsLen; i++){
342         if(('a'<=itsString[i]&&(itsString[i]<='z'))//a~z사이일 때 A~Z로 바꾼다(아스키코드)
343             itsString[i]-=32;
344     }
345     return *this; // 새로운 itsString return
346     delete thatsString; // 할당한 itsString 소멸
347 }
348
349 const char* String::c_str() const //문자열의 문자배열의 포인터값을 return
350 {
351     char* temp = new char[itsLen+1];
352     //char 포인터 temp에 문자열길이+1만큼의 char배열을 할당한다.
353     for (unsigned short i = 0; i<itsLen; i++){ //문자열 길이 만큼 반복한다.
354         temp[i] = itsString[i]; //temp에 원itsString배열의 내용을 넣어준다
355     }
356     temp[itsLen+1] = '\0'; //temp의 마지막 인덱스에 널문자를 넣어준다.
357     return temp; //char 포인터 temp를 반환한다.
358 }
359
360 const char* String::data() const
361 { //C++11 표준에서는 c_str과 동일하여 null문자를 추가한다.
362     char* temp = new char[itsLen+1];
363     //char 포인터 temp에 문자열길이+1만큼의 char배열을 할당한다.
364     for (unsigned short i = 0; i<itsLen; i++){ //문자열 길이 만큼 반복한다.
365         temp[i] = itsString[i]; //temp에 원itsString배열의 내용을 넣어준다
366     }
367     temp[itsLen+1] = '\0'; //temp의 마지막 인덱스에 널문자를 넣어준다.
368     return temp; //char 포인터 temp를 반환한다.

```



```

369 }
370
371 size_t String::copy (char* s, size_t len, unsigned int pos) const
372 {
373     unsigned short i,j; //원 문자열의 인덱스i와 copy할 문자열의 인덱스 j를 선언한다.
374     for(i=0,j=pos; i<len; i++,j++){ //len만큼 반복한다.
375         s[i]=itsString[j]; //s에 itsString의 값을 pos부터 시작해서 넣어준다.
376     }
377     return i;
378 }
379
380 size_t String::find (char c, size_t pos) const
381 {
382     size_t offset=pos; // offset에 pos를 할당한다.
383         //함수를 사용할 때 pos를 따로 지정하지 않으면 0으로 초기화된다.
384     while(pos<itsLen){ //문자열의 길이만큼 반복한다.
385         if(itsString[pos]==c){ //pos인덱스의 itsString가 c와 같다면
386             offset=pos; //offset은 pos이다.
387             break; //break문으로 반복문을 나온다.
388         }
389         else
390             pos++;
391         //pos인덱스의 itsString이 c와 다르면, pos를 증가시켜 다음 pos를 c와 비교하도록 한다.
392     }
393     if(itsString[offset] != c) offset=npos;
394     //구해진 offset인덱스의 itsString이 c가 아니라면 offset에 npos를 할당한다.
395
396     return offset; //offset을 반환한다.
397 }
398
399 size_t String::find (const String& str, size_t pos) const
400 {
401     size_t offset=pos; //offset에 시작값 pos를 할당한다.
402     unsigned short i; //unsigned short로 str.itsString의 인덱스로 쓸 i를 선언한다.
403
404     //비교구문
405     for(i=0; pos<itsLen; pos++){ //str.itsString의 인덱스 i를 0으로 초기화한다.

```

```

406 //pos가 문자열의 길이보다 작을 때만 실행된다, pos++로 itsString의 인덱스를 증가시킨다.
407     if (itsString[pos]==str.itsString[i]) //itsString[pos]와 str.itsString[i]가 같다면
408     {
409         offset=pos; //offset에 pos를 할당한다.
410         break; //그리고 break문으로 빠져나온다.
411     }
412     else {
413         continue; //만약 다르다면, continue로 찾을 때 까지 반복문을 실행한다.
414     }
415 }
416
417 for(i=0; i<str.itsLen; i++, pos++)
418 //원 스트링에서 str의 첫 문자와 같은 부분부터 str의 문자열을 비교하기 위해
419 str의 문자열 길이만큼 반복한다.
420 { //str의 첫 인덱스부터 다시비교하기 위해 i를 0으로 초기화한다.
421     if(itsString[pos]==str.itsString[i]) {
422         //원 스트링을 주어진 pos부터 시작하고, 비교할 string을 인덱스 0부터 비교한다.
423         offset=pos; // 같으면 offset에 pos를 할당하고
424         continue; // 다시 반복문을 실행한다.
425         원 스트링과 찾는 스트링 자체가 같은지를 비교하기 위해서이다.
426     }
427     else { //아니라면 원 스트링과 비교할 스트링이 서로 다르다는 의미이므로
428         offset=np; //offset에 np를 할당하고
429         break;} // 반복문에서 빠져나온다.
430     }
431 }
432 if(offset!=np) // 이렇게 찾은 offset이 np가 아니라면
433     offset=offset-str.itsLen+1;
434     //offset - str.itsLen+1 을 offset에 할당한다.
435     //offset에서 str.itsLen+1을 빼는 이유는 원 스트링과 비교할 스트링을 비교하는 과정에서
436     offset이비교할 스트링의 수만큼 증가했기 때문에 비교할 스트링의 null문자를 포함한
437     갯수만큼 다시 빼주어야 시작하는 위치를 offset에 할당할 수 있다.
438
439     return offset; //찾은 offset을 반환한다.
440 }
441
442

```

```

443 size_t String::find_first_of (const String& str, size_t pos) const
444 //원 스트링에서 찾는 스트링이 있는 어떠한 문자든지 그 위치를 반환하는 함수이다.
445 {
446     size_t i=0, count=0;
447     // 원래 스트링의 인덱스이자, 반환할 변수 i와 스트링을 비교해서 같은 문자의 갯수 count를
448     0으로 할당한다.
449     size_t j; // 찾을 스트링의 인덱스 j를 선언한다.
450     for(i=pos; i<itsLen; i++){
451         // 원래 스트링 인덱스 i에 pos를 할당하고, 원래 스트링의 길이만큼 반복한다.
452         for(j=0; j<str.itsLen; j++){
453             // 이중 반복문으로 찾을 스트링의 인덱스를 0~찾을 스트링의 갯수 만큼 반복한다.
454             if(itsString[i]==str.itsString[j]) {count++; break;}
455             //만약 같다면 count를 증가하고 break문으로 반복문을 빠져나온다.
456             else continue; //같지 않다면 continue문으로 두번째 반복문을 다시 진행한다.
457         }
458         if(itsString[i]==str.itsString[j]) {break;}
459         //다시 break문을 쓰는 이유는 이중 반복문에서 나오기 위해서이다.
460     }
461     if(count==0) {i=npos;}
462     //만약 같아서 증가한 count가 하나도 없고 0이라면, 같은 문자가 없다는 의미이므로
463     i에 npos를 할당한다.
464     return i; // 그리고 i를 반환한다.
465 }
466
467
468 size_t String::find_last_of (const String& str, size_t pos) const
469 // 스트링에서 찾을 문자와 매칭되는 마지막 문자의 위치를 반환한다.
470 {
471     if(pos==npos) pos=itsLen;
472     //만약 주어진 pos가 npos라면 pos에 스트링의 마지막 인덱스인 itsLen을 할당한다.
473     size_t i, count=0;
474     // 원래 스트링의 인덱스이자, 반환할 변수 i와 스트링을 비교해서 같은 문자의 개수
475     변수 count를 0으로 할당한다.
476     size_t j; // 찾을 스트링의 인덱스를 j로 선언한다.
477
478     for(i=pos; i>0; i--){ // i에 pos를 할당한다 pos의 기본값이 npos면, itsLen으로 맨 마지막 값이다.
479         // i는 점점 감소하므로, i가 0보다 클 때 실행하도록 한다.

```

```

480         for(j=0; j<str.itsLen; j++){
481             // j는 0으로 초기화 하고, j가 비교할 스트링의 갯수만큼 반복하고, j는 점점 증가한다.
482             if(itsString[i]==str.itsString[j]) {
483                 count++;
484                 break;
485             }
486             // 원래 스트링과 찾을 스트링이 같다면 count를 증가하고, break문으로
487             빠져나온다.
488             else
489                 break;
490             //만약 다르다면 break문으로 빠져나와 첫 반복문으로 가도록 한다.
491         }
492         if(itsString[i]==str.itsString[j]) break;
493         // 위에서 원 스트링과 비교할 스트링이 같다면 이중 반복문을 나오기 위해 break문을 쓴다.
494     }
495     if(count==0) i=npow;
496     // 만약 같아서 증가한 count가 하나도 없고 0이라면, 같은 문자가 없다는 의미이므로 i에 npow를
497     할당한다.
498     return i; //그리고 i를 반환한다.
499 }
500
501
502 String String ::substr (size_t pos, size_t len) const
503 //원 스트링에서 변수로 입력받은 위치부터, 갯수만큼 sub스트링을 반환하는 함수이다.
504 {
505     if(len==npow) len=itsLen-pos;
506     //만약 주어진 길이가 npow라면, 스트링의 끝까지 포함해야 하므로 전체 길이에서 pos를 뺀다.
507     String temp(len); //len만큼의 길이를 갖는 임시 스트링 temp를 생성한다.
508     for(unsigned short j=0; j<len; j++, pos++){
509         // temp 스트링의 index j를 0으로 초기화하고, 길이 len만큼 반복한다.
510         temp.itsString[j]= itsString[pos]; //temp의 스트링에 원 스트링의 pos부터 할당한다.
511         pos도 하나씩 증가시켜, 원 스트링의 다음 인덱스로 넘어가게 해준다.
512     }
513     temp.itsString[len] = '\0'; //temp의 마지막 인덱스에 널문자를 넣어 문자열로 만들어준다.
514     temp.itsLen=len; //temp의 itsLen변수를 len으로 초기화한다.
515     return temp; //temp를 반환한다.
516 }

```

```

517
518
519 int String::compare (const String& str) const
520 //원 스트링 a와 비교할 스트링 b를 비교해서 전혀 다르다면 0을,
521 // a가 b를 포함해 더 많은 스트링이 있다면 +같은 문자의 수,
522 // b가 a를 포함해 더 많은 스트링이 있다면 -같은 문자의 수를 반환한다.
523 {
524     unsigned short len = itsLen, i;
525     //len을 원래 스트링의 itsLen로 초기화하고, 원래 스트링의 인덱스 i를 선언한다.
526
527     int value=0; // 반환할 값 value를 0으로 초기화한다.
528
529     //비교구문
530     for(i=0; i<len; i++){ // i를 0부터 itsLen만큼 반복한다.
531         if (itsString[i]==str.itsString[i])
532             continue;
533             //원래 스트링과, 비교할 스트링이 같다면 반복문을 계속 진행한다.
534         else
535             break; // 만약 다르다면 break문으로 반복문을 나온다.
536     }
537     if(itsLen>str.itsLen) // 원 스트링의 길이가 비교할 스트링보다 크다면
538         {value = itsLen - i;} //value에서 itsLen에서 i를 뺀다.
539         // i는 같은 문자의 수이기 때문이다. 그러면 값이 양수가 된다.
540     else if (itsLen<str.itsLen) // 원 스트링의 길이가 비교할 스트링보다 작다면
541         {value = i-str.itsLen;}
542         // value에 i에서 비교할 스트링의 수를 뺀다. 그러면 값이 음수가 된다.
543     return value; // value를 반환한다.
544 }
545
546
547 //java
548
549 String String::concat(String str)
550 // 원래 스트링과 변수로 입력받은 스트링을 합친 새로운 스트링을 반환한다.
551 {
552     unsigned short totalLen=itsLen+str.itsLen;
553     //원래 스트링 길이와 합칠 스트링의 길이를 더한 변수를 만든다.

```

```

554 String temp(totalLen); //totalLen만큼의 임시 스트링 객체를 만든다.
555 for(unsigned short i=0; i<itsLen; i++) {
556     temp.itsString[i] = itsString[i];
557 } //임시스트링에 원 스트링을 넣어준다.
558 for(unsigned short i=itsLen, j=0; i<totalLen; i++, j++) {
559     temp.itsString[i] = str.itsString[j];
560 } //원 스트링을 넣고난 다음 인덱스부터 더할 스트링을 넣어준다.
561
562 temp.itsString[totalLen] = '\0';
563 // 임시 스트링의 마지막 인덱스에 널문자를 넣어 문자열로 만든다.
564 temp.itsLen=totalLen;
565 // 임시 스트링의 itsLen변수를 totalLen으로 초기화한다.
566
567 return temp; // 임시 스트링 temp를 반환한다.
568 }
569
570 String String::replace(char oldChar, char newChar)
571 //원 스트링의 oldChar문자를, newChar문자로 바꾸어준다.
572 {
573     String temp(itsLen); //원 스트링의 길이 만큼 임시 스트링을 선언한다.
574     for (unsigned short i = 0; i<itsLen; i++) { // 원 스트링의 길이만큼 반복한다.
575         if(itsString[i]==oldChar) //원 스트링의 인덱스의 내용이 oldChar이면
576         {
577             temp.itsString[i]=newChar;
578         } //인덱스에 newChar값을 할당한다.
579         else {
580             temp.itsString[i]=itsString[i];
581         } //만약 아니라면 원 스트링 값이 임시 스트링에 들어가게 한다.
582     }
583     temp.itsString[itsLen] = '\0';
584     // 임시 스트링의 마지막 인덱스에 널 문자를 넣어 문자열로 만들어준다.
585     temp.itsLen = itsLen; // 임시 스트링의 itsLen 변수를 원 스트링의 itsLen으로 초기화한다.
586     return temp; //임시 스트링을 반환한다.
587 }

```

String.cpp

```
1  // [리스트 11.12] 문자열 클래스 사용하기
2  //////////////////////////////////
3  //Homework by team
4  //
5  //Coding member functions for my String class.
6
7  #include <iostream> //Need not header file any more!!!
8
9  #include "String.h"
10 #include "StringMemberFunc.h"
11
12 using namespace std;
13
14
15 int main()
16 {
17     String s1("initial test");
18     cout << "S1:\t" << s1.GetString() << endl;
19
20     char * temp = "Hello World";
21     s1 = temp;
22     cout << "S1:\t" << s1.GetString() << endl;
23
24     char tempTwo[20];
25     strcpy(tempTwo, "nice to be here!");
26     s1 += tempTwo;
27     cout << "TempTwo:\t" << tempTwo << endl;
28     cout << "S1:\t" << s1.GetString() << endl;
29
30     cout << "S1[4]:\t" << s1[4] << endl;
31     s1[4]='x';
32     cout << "S1:\t" << s1.GetString() << endl;
33
34     cout << "S1[999]:\t" << s1[999] << endl;
35
36     String s2(" Another string");
```

```

37     String s3;
38     s3 = s1+s2;
39     cout << "S3:₩t" << s3.GetString() << endl;
40
41     String s4;
42     s4 = "Why does this work?";
43     cout << "S4:₩t" << s4.GetString() << endl;
44
45     ///////////////////////////////////
46     //Don't modify codes upper~~ ^^;;
47     //Just do your job below.
48
49     cout << "₩n---Test new member functions below.---₩n₩n";
50     //First, test your new 10 member functions of Homework!!!
51
52
53     String* pLittlePrince = new String("₩"My life is very monotonous,₩" the fox said. ₩"I hunt chickens;
54     men hunt me. ₩
55     All the chickens are just alike, and all the men are just alike. And, in consequence, I am a little bored.
56     ₩
57     But if you tame me, it will be as if the sun came to shine on my life . ₩
58     I shall know the sound of a step that will be different from all the others. ₩
59     Other steps send me hurrying back underneath the ground. Yours will call me, like music, out of my
60     burrow. ₩
61     And then look: you see the grain-fields down yonder? I do not eat bread. Wheat is of no use to me.
62     ₩
63     The wheat fields have nothing to say to me. And that is sad. But you have hair that is the colour of
64     gold. ₩
65     Think how wonderful that will be when you have tamed me! The grain, which is also golden, will
66     bring me bac k the thought of you. ₩
67     And I shall love to listen to the wind in the wheat...₩" The fox gazed at the little prince, for a long
68     time. ₩
69     ₩"Please-- tame me!₩" he said.");
70     //어린왕자 text에서 substr로 일정 길이만 잘라온 text로 테스트함
71
72     String stringAppend = " This is the end of the story. ";
73     String stringSwap = (*pLittlePrince).substr(0,29); //"₩"My life is very monotonous,

```



```

74
75
76 //String substr (size_t pos = 0, size_t len = npos) const; 원 스트링에서 변수로 입력받은 위치부터,
77 갯수만큼 sub스트링을 반환하는 함수.
78 String test = (*pLittlePrince).substr(0,253);
79 //₩"My life is very monotonous,₩" the fox said. ₩"I hunt chickens; men hunt me. ₩
80 //All the chickens are just alike, and all the men are just alike. And, in consequence, I am a little
81 bored. ₩
82 //But if you tame me, it will be as if the sun came to shine on my life . ₩
83
84 cout << "*" Test functions with this text : ₩n" << test.GetString() << endl; //문자열 전체를 출력
85
86 cout << "₩n Test: length() = " << test.length() << endl; //문자열의 길이(크기) 출력
87
88 char r = test.at(10); //문자열의 10번째 문자찾기
89 if(r == String::npos)
90     cout << "범위를 벗어남₩n";
91 else
92     cout << " Test: at(10) = " << r << endl;
93
94 cout << " Test: front() = " << test.front() << endl; //문자열의 처음
95
96 test.append(stringAppend); //문자열에 이어서 stringAppend 추가
97 cout << "₩n Test: append(stringAppend) = ₩n" << test.GetString() << endl;
98
99 test.upper(); //문자열을 대문자로 변경
100 cout<< "₩n Test: upper() = ₩n" << test.GetString() <<endl;
101
102 test.swap(stringSwap); // 주어진 문자열을 stringSwap으로 통변경
103 cout<<"₩n Test: swap(stringSwap) = ₩n" << test.GetString() <<endl;
104
105
106 //size_t copy (char* s, size_t len, size_t pos = 0) const; //문자형 포인터에 원 스트링의 pos부터 len의
107 길이를 갖는 문자들을 저장
108 char buffer[20];
109 size_t length = test.copy(buffer,5,16); //버퍼에 원래 스트링의 16번째 인덱스부터 5개의 문자열을 저장.
110 buffer[length]='₩0';

```

```

111         cout << "WnWn Test: copy(buffer,5,16) = buffer contains:" << buffer << 'Wn';
112
113 //size_t find (const String& str, size_t pos = 0) const; test에서 my가 위치한 자리를 반환
114 String test2 = (*pLittlePrince).substr(246,2); //my
115 size_t found = test.find(test2);
116 if (found!=String::npos)
117     cout << "WnWn Test: find(test2) = first 'my' found at: " << found << 'Wn';
118
119 //int compare (const String& str) const; //W"My life is very monotonous,"와 my 비교
120 cout << "WnWn Test: compare(test2) = Wn" << test.GetString() << " compared with " <<
121 test2.GetString() << " is : " << test.compare(test2) <<'Wn';
122
123 //size_t find_first_of (const String& str, size_t pos = 0) const; test에서 aeiou의 내용이 있으면 *로 바꿈
124 found = test.find_first_of("aeiou");
125 while (found!=String::npos)
126 {
127     test[found]='*';
128     found=test.find_first_of("aeiou",found+1);
129 }
130 cout << "WnWn Test: find_first_of('aeiou', found+1) = " << test.GetString() << 'Wn';
131
132
133 //JAVA
134
135 String test3 = (*pLittlePrince).substr(0,29); //W"My life is very monotonous,W
136
137 //String concat(String str); //원래 스트링과 변수로 입력한 스트링을 합치는 함수
138 cout << "WnWn Java Test: concat('all the time.') = Wn" << test3.concat(" all the time.").GetString()
139 <<'Wn';
140
141 // String replace(char oldChar, char newChar);// 스트링의 oldChar을 newChar로 바꾸는 함수
142 cout << "Wn Java Test: replace('i','T') = " << test3.replace('i','T').GetString()<<'Wn';
143 //스트링의i를 T로 바꿈
144
145
146
147

```

```

148
149 cout << "WnWn---Count words and alphabets in The Little Prince's text.---WnWn";
150
151 cout << pLittlePrince -> GetLen() << " characters by teacherWn" ;
152 //Test length: 911 characters. But should make a new member function 'length()'.
153 cout << pLittlePrince->length() << " characters by 5조" << endl; //문자열 길이 출력

154 //Second, print out 'The Little Prince's text' and count words and alphabets all.
155 //
156
157 cout << "Wn* Printed out 'The Little Prince' by at() function :Wn";
158 for(unsigned int i = 0; i < pLittlePrince->length(); i++) //문자열을 0부터 문자열의 길이만큼
159     cout << pLittlePrince->at(i); //인덱스를 사용해서 한문자씩 출력
160
161
162 //count alphabets
163 cout << "WnWn* Count alphabets :Wn";
164
165 char lowercase = 'a';
166 char uppercase = 'A';
167 unsigned short countAlpa[26], countBeta[26];
168 //countAlpa는 소문자배열, countBeta는 대문자배열
169 //알파벳은 A(a)부터 Z(z)까지 26. 배열 메모리는 26.
170
171 for(unsigned short i = 0; i < 26; i++){
172     //for문을 통해 두 배열은 각각 소문자, 대문자 저장
173     //countAlpa[0] = 'a' ... countAlpa[25] = 'z'
174     //countBeta[0] = 'A' ... countBeta[25] = 'Z'
175     unsigned short countLow = 0, countUp = 0;
176     //countLow는 소문자 카운트변수, countUp은 대문자 카운트변수
177     for(unsigned short j = 0; j < pLittlePrince->length(); j++){
178         //pLittlePrince의 문자 at(j)가 특정 소문자와 같다는 조건에 충족하면 countLow +1 문자열
179         //길이만큼 반복
180         if(pLittlePrince->at(j) == (lowercase+i))
181             countLow++;
182         //pLittlePrince의 문자 at(j)가 특정 대문자와 같다면 countUp +1 문자열 길이만큼 반복
183         else if(pLittlePrince->at(j) == (uppercase+i))

```

```

184         countUp++;
185     }
186     countAlpa[i] = countLow; //카운트된 i번째 소문자를 배열에 저장
187     countBeta[i] = countUp; //카운트된 i번째 대문자를 배열에 저장
188
189     //i는 unsigned short이기 때문에 static_cast<char>로 데이터타입을 char로 바꿈
190     //출력은 특정 알파벳의 소문자의 갯수+대문자의 갯수로 합쳐서 출력. 대소문자 구분 안함
191     cout << " " << static_cast<char>(uppercase + i) << " : " << countAlpa[i] + countBeta[i] << endl;
192 }
193
194
195 //count words
196 unsigned short countWord = 0;
197 for(unsigned short i = 0; i < pLittlePrince->length(); i++){ //0부터 문자열길이만큼
198     //문자열은 단어와 단어사이의 ' ' (공백)으로 구성되어있기 때문에 함수 at()으로 ' '공백을 찾으면 카운트
199     if(pLittlePrince->at(i) == ' ')
200         countWord++;
201 }
202
203 //단어수는 공백+1이므로 countWord+1로 출력
204 cout << "\n* Count words : " << countWord + 1 << " words" << endl;
205
206 //자유기억공간의 메모리에 저장돼있는 내용을 지우는 것이기에
207 //void clear() 함수는 프로그램 마지막에 구현함
208 pLittlePrince->clear();
209 cout << "\n Test: clear() = " << pLittlePrince->GetString() << endl;
210
211
212 system("pause");
213
214 return 0;
215 }

```

- 각 멤버함수의 실행결과 화면

> String.cpp 내의 새로운 10 개 멤버함수 테스트

```
C:\Users\User\Desktop\객지\팀과제\Debug\5조-String클래스.exe

---Test new member functions below.---

* Test functions with this text :
"My life is very monotonous," the fox said. "I hunt chickens; men hunt me. All t
he chickens are just alike, and all the men are just alike. And, in consequence,
I am a little bored. But if you tame me, it will be as if the sun came to shine
on my life .

Test: length() = 253
Test: at(10) = s
Test: front() = "

Test: append(stringAppend) =
"My life is very monotonous," the fox said. "I hunt chickens; men hunt me. All t
he chickens are just alike, and all the men are just alike. And, in consequence,
I am a little bored. But if you tame me, it will be as if the sun came to shine
on my life . This is the end of the story.

Test: upper() =
"MY LIFE IS VERY MONOTONOUS," THE FOX SAID. "I HUNT CHICKENS; MEN HUNT ME. ALL T
HE CHICKENS ARE JUST ALIKE, AND ALL THE MEN ARE JUST ALIKE. AND, IN CONSEQUENCE,
I AM A LITTLE BORED. BUT IF YOU TAME ME, IT WILL BE AS IF THE SUN CAME TO SHINE
ON MY LIFE . THIS IS THE END OF THE STORY.
```

```
C:\Users\User\Desktop\객지\팀과제\Debug\5조-String클래스.exe

Test: swap(stringSwap) =
"My life is very monotonous,"

Test: copy(buffer,5,16) = buffer contains: mono

Test: find(test2) = first 'my' found at: 3

Test: compare(test2) =
"My life is very monotonous," compared with 1 is : 284

Test: find_first_of('aeiou', found+1) = "My l*f* *s v*ry m*n*t*n*s,"

Java Test: concat('all the time.') =
"My life is very monotonous," all the time.

Java Test: replace('i','I') = "My lIfe Is very monotonous,"
```

```
C:\Users\User\Desktop\객지 팀과제\Debug\5조-String클래스.exe
---Count words and alphabets in The Little Prince's text.---

911 characters by teacher
911 characters by 5조

* Printed out 'The Little Prince' by at(<) function :
"My life is very monotonous," the fox said. "I hunt chickens; men hunt me. All t
he chickens are just alike, and all the men are just alike. And, in consequence,
I am a little bored. But if you tame me, it will be as if the sun came to shine
on my life . I shall know the sound of a step that will be different from all t
he others. Other steps send me hurrying back underneath the ground. Yours will c
all me, like music, out of my burrow. And then look: you see the grain-fields do
wn yonder? I do not eat bread. Wheat is of no use to me. The wheat fields have n
othing to say to me. And that is sad. But you have hair that is the colour of go
ld. Think how wonderful that will be when you have tamed me! The grain, which is
also golden, will bring me bac k the thought of you. And I shall love to listen
to the wind in the wheat..." The fox gazed at the little prince, for a long tim
e. "Please-- tame me!" he said.

C:\Users\User\Desktop\객지 팀과제\Debug\5조-String클래스.exe
* Count alphabets :
A : 50
B : 11
C : 14
D : 26
E : 86
F : 18
G : 11
H : 48
I : 48
J : 2
K : 10
L : 43
M : 23
N : 46
O : 53
P : 4
Q : 1
R : 25
S : 35
T : 65
U : 26
V : 5
W : 16
X : 2
Y : 13
Z : 1

* Count words : 187 words

Test: clear(<) =
계속하려면 아무 키나 누르십시오 . . .
```

>그 외의 멤버함수 실행결과 화면

- 소스코드

- 어린왕자 text 의 일부를 이용하여 멤버함수 구현

```
1  #include <iostream> //Need not header file any more!!!
2
3  #include "String.h"
4  #include "StringMemberFunc.h"
5
6  using namespace std;
7
8  int main()
9  {
10
11  String* pLittlePrince = new String("₩"My life is very monotonous,₩" the fox said. ₩"I hunt chickens;
12  men hunt me. ₩
13  All the chickens are just alike, and all the men are just alike. And, in consequence, I am a little bored.
14  ₩
15  But if you tame me, it will be as if the sun came to shine on my life .");
16
17
18  String stringInsert = "This story is about Little Prince .:";
19  String stringReplace = "really";
20
21  cout << " Test: size() = " << pLittlePrince->size() << endl;
22  cout << "₩n Test: empty() = " << pLittlePrince->empty() << endl;
23  cout << "₩n Test: back() = " << pLittlePrince->back() << endl;
24
25  pLittlePrince->insert(0, stringInsert);
26  cout << "₩n Test: insert(0, stringInsert) = ₩n " << pLittlePrince->GetString() << endl;
27
28  pLittlePrince->erase(0,35);
29  cout << "₩n Test: erase(0,34) = ₩n" << pLittlePrince->GetString() << endl;
30
31  pLittlePrince->replace(11,15,stringReplace);
32  cout << "₩n Test: replace(11,15,stringReplace) = ₩n" << pLittlePrince->GetString() << endl;
```

```

33
34 pLittlePrince->lower();
35 cout << "Wn Test: lower() = Wn" << pLittlePrince->GetString() << endl;

36 //String (unsigned int n, char c); n개의 문자c를 갖는 스트링 생성자
37 String str(9,'x');
38 cout << "Wn Test: String(9, 'x') = " << str.GetString() << ", Len = " << str.GetLen() << endl;
39
40
41 //String substr (size_t pos = 0, size_t len = npos) const; 원 스트링에서 변수로 입력받은 위치부터,
42 갯수만큼 sub스트링을 반환하는 함수.
43 String test = (*pLittlePrince).substr(0,29); //W"My life is very monotonous"
44
45 //const char* c_str() const; 스트링을 문자배열로 만들어 주는 함수
46 cout << "Wn Test: c_str() = " << test.c_str() << endl;
47
48 //const char* data() const; 스트링을 문자배열로 만들어 주는 함수
49 cout << "Wn Test: data() = " << test.data() << endl;
50
51 //size_t find (char c, size_t pos = 0) const; test에서 공백의 자리를 반환
52 size_t found=test.find(' ');
53 if (found!=String::npos)
54     cout << "Wn Test: find(' ') = space found at " << found << 'Wn';
55
56 //size_t find_last_of (const String& str,size_t pos = npos) const; 특정 단어를 뒤에서부터 찾을.
57 cout << "Wn Test : find_last_of('y') = WnWt 원 스트링 : " << test.GetString() << 'Wn';
58 cout << "Wt pos전에 y가 있는 곳: " << test.find_last_of("y") << 'Wn';
59 cout << endl;
60
61 system("pause");
62
63 return 0;
64 }

```


- 출력화면

```
C:\Users\User\Desktop\객지\팀플\Debug\5조-String클래스.exe

Test: size() = 253

Test: empty() = 0

Test: back() = .

Test: insert(0, stringInsert) =
This story is about Little Prince : "My life is very monotonous," the fox said.
"I hunt chickens; men hunt me. All the chickens are just alike, and all the men
are just alike. And, in consequence, I am a little bored. But if you tame me, it
will be as if the sun came to shine on my life .

Test: erase(0,34) =
"My life is very monotonous," the fox said. "I hunt chickens; men hunt me. All t
he chickens are just alike, and all the men are just alike. And, in consequence,
I am a little bored. But if you tame me, it will be as if the sun came to shine
on my life .

Test: replace(11,15,stringReplace) =
"My life isreally monotonous," the fox said. "I hunt chickens; men hunt me. All
the chickens are just alike, and all the men are just alike. And, in consequence
, I am a little bored. But if you tame me, it will be as if the sun came to shin
e on my life .
```

```
C:\Users\User\Desktop\객지\팀플\Debug\5조-String클래스.exe

Test: lower() =
"my life isreally monotonous," the fox said. "i hunt chickens; men hunt me. all
the chickens are just alike, and all the men are just alike. and, in consequence
, i am a little bored. but if you tame me, it will be as if the sun came to shin
e on my life .

Test: String(9, 'x') = xxxxxxxxx, Len = 9

Test: c_str() = "my life isreally monotonous,

Test: data() = "my life isreally monotonous,

Test: find(' ') = space found at 3

Test : find_last_of('y') =
원 스트링 : "my life isreally monotonous,
pos전에 y가 있는 곳: 16

계속하려면 아무 키나 누르십시오 . . .
```