Strings in C++

Strings

- C++ provides the following two types of string representations:
  - The C-style character string.
  - The string class type introduced with Standard C++.
- The C-Style Character String:
  - It originated in the C language and has continued to be supported in C++.
  - This string is actually a one-dimensional array of characters which is terminated by a nullcharacter '\0’.
  - Thus a null-terminated string contains the characters that comprise the string followed by a null.
- The following declaration and initialization create a string consisting of the word “Hello”. To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word “Hello”.

```
char str[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

C-style String Initialization

```
int main ()
{
char str[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
cout<< “string    : ";
cout<< str<< endl;
return 0;
}
```

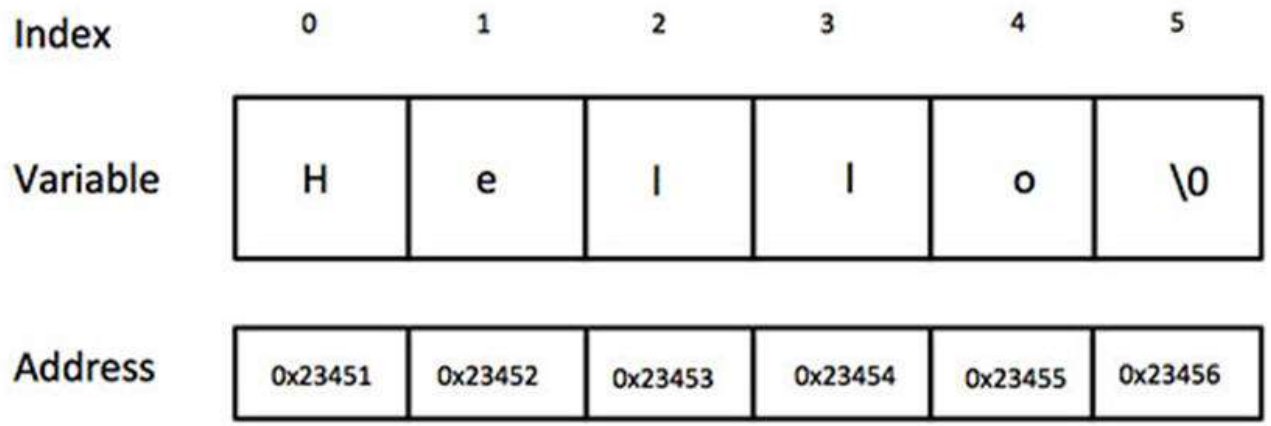
- If you follow the rule of array initialization, you can also initialize and declare a C-style string as follows:

```
char str[ ] = "Hello";
```

- You do not need to actually place the null character at the end of a string constant. The C++ compiler automatically places the '\0' at the end of the string when it initializes the array.

C-style String Initialization

- Following is the memory presentation of the above defined string in C/C++:



Example on C-style String Declaration

- To declare a C-style string, simply declare a char array and assign it a value:

  Char st[ ]= “welcome”;

- Although “welcome” is only 7 letters, this actually declares an array of length 8. The following program prints out the length of the string, and then the ASCII values of all of the characters:

```
main()
{
char st[]="welcome";
int i;
cout<< sizeof(st) << endl;
for (int j = 0; j < sizeof(st); j++)
{
i=st[j];
cout<<"  character  "<< st[j]<<"  value   "<<i<<endl;}
}
```

Example on C-style String Declaration (cont.)

Output:

```
8
 character  w  value   119
 character  e  value   101
 character  l  value   108
 character  c  value   99
 character  o  value   111
 character  m  value   109
 character  e  value   101
 character     value   0
```

- That 0 is the ASCII code of the null terminator that has been appended to the end of the string.

```
main( )
{ char s1[]="computer C++";
cout<< sizeof(s1)<<endl;
intx;
for(int i=0; s1[i]; i++)
{ x=s1[i]; cout<< s1[i]<<"    "<<x<<endl;}}
```

Output

```
13
c    99
o    111
m    109
p    112
u    117
t    116
e    101
r    114
32
C    67
+    43
+    43
```

Size of Built-in Data Types

Type	Size
bool, char, char8_t, unsigned char, signed char	**1 byte**
char16_t, short, usigned short, wchar_t	2 bytes
char32_t, float, int, unsigned int, long, unsigned long	4 bytes
double, long double, long long, unsigned long long	8 bytes

Note on C-style String Initialization

- One important point to note is that strings follow ALL the same rules as arrays. This means you can initialize the string upon creation, but you can not assign values to it using the assignment operator after that!

```
char szString[ ] = "string";  // ok
szString= "rope";             // not ok!
```

- You can read text into a string using cin:

```
char szString[255];
cin>> szString;
cout<< "You entered: " << szString<< endl;
```

- Why did we declare the string in the previous example to be 255 characters long?
- The answer is that we don't know how many characters the user is going to enter, so we are using this array of 255 characters as a buffer.
- A buffer is memory set aside temporarily to hold data.
- If the user were to enter more characters than our array could hold, we would get a buffer overflow. A buffer overflow occurs when the program tries to store more data in a buffer than the buffer can hold.
- Buffer overflow results in other memory being overwritten, which usually causes a program crash, but can cause any number of other issues.
- By making our buffer 255 characters long, we are guessing that the user will not enter this many characters.
- Although this is commonly seen in C/C++ programming, it is poor programming.
- The recommended way of reading strings using cin is using "getline", as follows:

```
char szString[30];
cin.getline(szString, 30);
cout<< "You entered: " << szString<< endl;
```

- This call to cin.getline() will read up to 30 characters into szString (leaving room for the null terminator!).
- Any excess characters will be discarded. In this way, we guarantee that buffer overflow will not occur.

String Built-in Functions and their Purpose

strcpy(S1, S2)	Copies string S2 into string S1.
strcat(S1, S2)	Concatenates string S2 onto the end of string S1.
strlen(S1)	Returns the length of string S1.
strcmp(S1, S2)	Returns 0 if S1 and S2 are the same, less than 0 if S1<S2, greater than 0 if S1>S2.
strchr(S1, ch)	Returns a pointer to the first occurrence of character ch in string S1.

- Note: in strcpy() and strcat(), to avoid overflows, the size of the destination string shall be long enough to contain the source string (including the terminating null character), and should not overlap in memory with the source.

String Built-in Functions and their Purpose

strcpy(S1, S2)	Copies string S2 into string S1.
strcat(S1, S2)	Concatenates string S2 onto the end of string S1.
strlen(S1)	Returns the length of string S1.
strcmp(S1, S2)	Returns 0 if S1 and S2 are the same, less than 0 if S1<S2, greater than 0 if S1>S2.
strchr(S1, ch)	Returns a pointer to the first occurrence of character ch in string S1.

- Note: strcmp() compares the **two strings lexicographically** (the same way used to sort words alphabetically in dictionaries), meaning that it starts comparison character by character, starting from the first character.
- If first character in both strings are equal, then this function will check the second character, if this is also equal then it will check the third and so on.
- This process continues until a mismatch between the two strings is encountered or a character in either string is NULL.
- When S1 and S2 are not the same, the value returned by strcmp() function is the difference between the ASCII values of the first unmatched characters in S1 and S2.

```
#include <iostream>
#include <string.h>
using namespace std;

int main ()
{char str1[20] = "Hello";
char str2[20] = " and welcome";
char str3[30];
int len;
cout <<"length of str1= "<<strlen(str1)<<" length of str2= "<<strlen(str2)<<endl;
// What is the size allocated for str1 in memory?
cout <<"size of str1= "<< sizeof(str1)<<endl;
// copy str1 into str3
strcpy(str3,str1);
cout<<"strcpy(str3, str1): " <<"str3 "<< str3 << endl;
// concatenates str1 and str2
strcat(str1,str2);
cout<<"strcat(str1, str2): " <<"str1 "<< str1 << endl;
// total lenghth of str1 after concatenation
len = strlen(str1);
cout<<"strlen (str1) : " << len << endl ;
return 0;}
```

Output:

length of str1= 5 length of str2= 12

size of str1= 20

strcpy(str3, str1): str3 Hello

strcat(str1, str2): str1 Hello and welcome

strlen (str1) : 17

```
#include <iostream>
#include <string.h>
using namespace std;

//Examples on string functions
main ( )
{
   char s1[25]="hello now";
   char s2[25]="computer c++";
   char s3[25]="computer electronics";
   cout<<"length of s1= "<<strlen (s1)<< ", s1= "<<s1<<endl<<
      "length of s2= "<<strlen (s2)<<", s2= "<<s2<<endl<<
      "length of s3= "<<strlen (s3)<<", s3= "<<s3<<endl<<endl;
   int y= strcmp (s1,s2);
   cout <<"y= "<<y<<" ";
   if(y) cout <<"not matched strings"<<endl<<endl;
   else cout <<"matched strings"<<endl;
   strcat(s1," welcome");
   cout<<"length after strcat "<<strlen (s1)<<" new string "<<s1<<endl<<endl;
   strcpy(s2,s3);
   y=strcmp (s2,s3); cout <<"y= "<<y<<" ";
   if(y) cout <<"not matched strings"<< endl;
   else cout <<"matched strings"<<endl;}
```

Output:

length of s1= 9, s1= hello now
length of s2= 12, s2= computer c++
length of s3= 20, s3= computer electronics

y= 5 not matched strings

length after strcat 17 new string hello now welcome

y= 0 matched strings

```
//Comparing strings as array of characters
main ()
{
   char s1[20]="computer";
   char s2[20]="circuits";
   if(!strcmp(s1,s2)) cout <<"equal strings"<<endl;
   else cout <<"different strings"<<endl;
   int k= strcmp(s1,s2);
   if(k>0) cout <<"k = "<<k<< " string1 is greater than string2"<<endl;
   else cout <<"k = "<<k<<" string1 is less than string 2"<<endl;}
```

Output:

different strings

k = 6 string1 is greater than string2

Class String

- The standard C++ library provides a string class type that supports all the operations mentioned for the strings that are declared as an array of characters, in addition to much more functionality.
- To use class string, the header file <string> or <cstring> must be included.
- Declaration:

```
string str1 = "Hello";

string str2 = "World";

string str3;
```

- In what follows, we will show some string functions that can be used by class string.

Class String Functions

1-Reading strings:

cin>> str1; //reads a string into st1, only until a space appears

getline(cin,str1); //reads with spaces, reading strings till enter (new line)

2-Length of string:

string.length() function

i.e. to get the length of string str1:use str1.length();

3-Append strings:

String s1=”ahmed”; string s2=”mohammed”;

s1+=s2; //then s1= “ahmedmohammed”

string s3=”car”; s3+=” new”; //then s3 will be “car new”

- Or we can use the append function

s1.append(“…student”); // then s1 will be “ahmedmohammed…student”

Class String Functions

4-Substrings:

- To take a part of string s1 from char no. i1 with length equal to i2:

s1.substr(i1,i2);

Ex:

string1="first and..second"; s2= string1.substr(6,5);

cout<< "sub string s2 "<<s2<endl; //sub string s2 will be "and.."

5-Swapping strings swap( ) function:

string1.swap(string2), the two strings will be swapped

Ex:

string s1("first"); string s2("second");

s1.swap(s2); //s1 will be "second" and s2 will be "first"

Class String Functions

6-Insert( ) function:

- It is used to insert string in another string at certain position

s1.insert(k,st); //insert string st into string s1 at index k

Ex: *string2="we are here"; st="not ";*

string2.insert(7,st) // then string string2 will be "we are not here"

7-Find( ) function:

- To find the position of characters or certain string in a string.

string.find(the required characters or string)

Ex: *str1=" electronic";*

cout<<str1.find("c"); //the output will be 3

cout<< str1.find("ron"); //the output will be 5

Ex: *string st1, st2;*

st1="com logic and sequential logic"; st2="logic";

cout<<"character was found at "<<st1.find(st2); // (output= 4)

Class String Functions

8-rfind() function:

- function to find characters in a string backwards.

string.rfind(string)

Example:

string st1,st2;

st1="com logic and sequential logic"; st2="logic";

cout<<" character was found at "<<st1.rfind(st2);//(output= 25)

Examples

1-Dealing strings character by character using data type string

```
for (inti=0; i<stringOne.length(); i++)
cout<<stringOne[i]<<“  ”;  // if stringOne=“hello”,      // the output will be: h  e  l  l  o
```

2-Comparing strings:

Use all logic operators as: if(s1 > s2) or if(s1 == s2) and so on.

```
Ex:  string string2=”computer”;     string string3=”circuits”;
if(string3==string2) cout<<"equal strings"<<endl;
else cout<<" different  strings"<<endl;
if(string3>string2) cout<<"string3 is greater than string2"<<endl;
else cout<< "string3 is less than string2";
```

output:

different strings

string3 is less than string2

```
#include <iostream>
#include <string.h>
using namespace std;

int main ()
{
   string str1 = "Hello";
   string str2 = "World";
   string str3;
   int len;
   // copy str1 into str3
   str3 = str1;
   cout<<"str3 : "<<str3<<endl;
   // concatenates str1 and str2
   str3 = str1 + str2;
   cout<<"str1 + str2 : "<<str3<<endl;
   // total lenghth of str3 after concatenation
   len = str3.length();
   cout<<"str3_length : "<<len<<endl; return 0;}
```

Example 3 (cont.)

Output:

str3 : Hello
str1 + str2 : HelloWorld
str3_length : 10

Example 4

```
// This program uses find function to get number of spaces between words in a certain statement.

int main()
{
   string str;
   str = "you must study hard";
   int p,n;
   n = str.length ();
   cout <<"length of string "<<n<<endl;
   int i = 1;
   p = str.find (" "); cout <<"location no."<< i<<" is found at "<<p<< endl ;
   while(p>=0)
   {i ++;
   p = str.find (" ",p+1); cout <<"location no."<< i <<" is found at "<<p<< endl ; }
   return 0;}
```

Example 4 (cont.)

length of string 19
location no.1 is found at 3
location no.2 is found at 8
location no.3 is found at 14
location no.4 is found at -1

Example 5

```
main()
{
   string s1, s2, s3,s4,s5;
   getline(cin,s1); //enter s1="new c++"
   getline(cin,s2); // enter s2=" program"
   cout<<"s1 "<<s1<<endl<<"s2 "<<s2<<endl;
   int k1=s1.length(); int k2= s2.length();
   cout<<"length of s1= "<<k1<<" length of s2= "<<k2<<endl;
   s3=s1.substr(4,3);
   cout<<"substring s3 "<<s3<<endl;
   if(s1==s2) cout <<"equal"<<endl;
   else if(s1>s2) cout <<"s1 is greater than s2"<<endl;
   else cout <<"s1 is smaller than s2"<<endl;
```

Example 5 (cont.)

```
int k = s1.find("c");
cout<<"letter c is found at index "<<k<<endl;
s1.insert(4,"example ");
cout<<"string s1 after insert= "<<s1<<endl;
s4="one"; s5="two";
cout <<"s4 "<<s4<<endl<<"s5 "<<s5<<endl;
s4.swap(s5);
cout<<"strings after swap"<<endl<<"s4 "<<s4<<endl <<"s5 "<<s5<<endl;
// strcpy (s1,"ffff"); error
s1+= s2;
cout <<"s1 after concatenation= "<<s1<<endl;}
```

Output:
new c++
program
s1 new c++
s2 program
length of s1= 7 length of s2= 8
substring s3 c++
s1 is greater than s2
letter c is found at index 4
string s1 after insert= new example c++
s4 one
s5 two
strings after swap
s4 two
s5 one
s1 after concatenation= new example c++ program

Strings and Functions

- **We can send string either as an array of characters or as type string.**

```
void fun_1 (char s[], string ss);
 void fun_2 (char s[], string &ss);
main( ) {char st[20] =" c++ program ";
   string s2=" first string";
cout<<" string_1 "<< st<<"     string 2 "<<s2 <<endl<<endl;
fun_1( st, s2); cout<< "strings after calling function_1 "<<endl<<st
<<"      "<< s2<<endl<<endl;
fun_2( st, s2); cout<< "strings after calling function_2  "<<endl<<st
<<"       "<< s2;  }
void fun_1 (char s[],string ss)
{ cout<< "strings in function_1 "<<s<<"       "<<ss<<endl<<endl;
 strcpy ( s, "logic design ");   ss=" second string" ;}
 void fun_2 (char s[],string &ss)
{ cout<< "strings in function_2 "<<s<<"     "<<ss<<endl<<endl;
 strcpy ( s, " new string ");    ss=" second string" ;}
```

Output

string_1 c++ program string 2 first string

strings in function_1 c++ program first string

strings after calling function_1
logic design first string

strings in function_2 logic design first string

strings after calling function_2
new string second string

Example on Sending Data Type String

// send the string as values, so if it is changed in function, it cannot be changed in main

// send the string by reference, so if it is changed in function, it will be changed in main

```
void fun_1(string st1 );
void fun_2(string &st2 );

int main()
{           string st=" test c++";
   cout<<" string in main =  "<<st<<endl<<endl;
  fun_1(st);
 cout<<" string after calling first function=   "<<st<<endl<<endl;
 st=" test c++";
    fun_2(st);
 cout<<" string after calling second function=   "<<st<<endl;  }

            //string call by value
            void fun_1(string st1 )
            {  cout<<"string in function_1=   "<<st1<<endl<<endl;
            st1=" exam c++ "; }

// string call by reference
              void fun_2(string &st2 )
            {  cout<<"string in function_2=  "<<st2<<endl<<endl;
            st2=" exam c++ "; }
```

Example (cont.)

Output

string in main = test c++
string in function_1= test c++
string after calling first function= test c++
string in function_2= test c++
string after calling second function= exam c++

Array of Strings

- It can be declared as follows:

```
char str[5][20];    // two dimensional array of characters
string names[5];   // one dimensional array of strings
```

- Reading array of strings:

Ex 1:

```
char str[5][20];
for(int i=0; i<5; i++)
gets(str[i]) ;
```

Ex 2:

```
string names [5];
for(int i=0; i<5; i++)
getline(cin, names[i]);
```

Example on Using Array of Strings

Write a program that reads names of n students and search for the location of certain name in the array using search function