Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Character Strings

Quick Recap

IIT Bombay

- In earlier sessions, we have discussed variables of type char

  char CH; .

- Variable CH can hold the ASCII code of a character.
  - It is a numerical value which can also be treated as an integer number.
- We also discussed string as a sequence of characters.

Overview

- In this session, we will discuss how strings can be meaningfully handled by our programs.
- To perform useful operations on strings , we will use functions provided in C++ library.

C++: A Historical Note

- Bjarne Stroustrup, the inventor of C++, designed this language, essentially building on the language C.
- Language C does not have support for objects and classes, and is classified as a procedural programming language.
- C++ supports Object-Oriented programming style, in addition to permitting all the procedural features of C language.
- In the second part of the programming course, we will study the object-oriented features.
  - Presently we will discuss C-style string handling.

String Handling in C++

IIT Bombay

- Defining and using character strings in C++ can be done in two different ways:
  - Use an array of type char, to store a character string.
  - In the object oriented style of handling a character string, it is defined as an ‘object’ belonging to string class.

    (This will be discussed in the other part of the course)

Defining and Using Character Strings

- We have seen that a string can be stored in a char type array.
- An extra character '\0' is stored at the end.
  - Size of the array should include an extra element to hold this value
- If the string has no blank space within, it can be read directly in an array by using cin.
  - The special character '\0' is automatically inserted at the end of the string by cin.
- If we form a string in our program, in an array by putting characters in it, then it is our responsibility to put the '\0' character at the end

A program to determine length of a string

IIT Bombay

```
#include <iostream>
using namespace std;
int main() {
   char Word[6];    int wordlength =0, i;
   cout << "input string: " << endl;     cin >> Word;
   for (i =0; i < 6; i++){
     if (Word[i] == '\0') break;
   }
   wordlength = i;
   cout <<"Length of the string is " << wordlength << endl;
   return 0;
}
```

Interesting Behavior

IIT Bombay

- If we execute this program using Code::Blocks, and give an input string as:

  Hello

  The program shows the correct length as 5

- If the input string is

  abracadabra

  Program shows the length as 6!

  - Some compilers work differently, and produce a run time error

Array Bounds

IIT Bombay

- The cin command merely reads ALL characters of the string into consecutive elements of the array Word[], and inserts '\0' at the end.
  - This works for an input string which is less than 6 characters long.
- input string abracadabra has 11 characters.
  - C++ still does the same thing. It will now put additional characters in subsequent locations beyond the size of the array Word[].
  - The for loop will exit after completing its normal run.
  - Value of i will be 6, which will get assigned to wordlength.
  - The array word will NOT have a well formed string
- C++ expects us to check if array bounds are exceeded.

A Better Program

IIT Bombay

```
    for (i =0; i < 6; i++){
      if (Word[i] == '\0') break;
    }
    if (i==6){
       cout << "invalid string" << endl;  return -1;
    }
    wordlength = i;
    cout <<"Length of the string is " << wordlength << endl;
    return 0;
}
```

A Better Strategy

IIT Bombay

- Always declare the array with a sufficiently large size.

  e.g. char Word[50];

- C++ provides a library function called strlen(), which will find the length of a string stored in a char array.
- We need to include the corresponding library in our header

  #include <cstring>

- With this inclusion, we can write in our program:

  wordlength = strlen(Word);

Handling Sentences

- A sentence has many words, separated by blank spaces, or by punctuation marks.
- The cin statement of C++ will stop reading input when it encounters a blank space.
  - To get around this, we need a facility to read complete lines of text.
- A library function gets(string) reads such lines.
  - To use it, we need to #include <cstdio>.
- It reads all input characters up to the newline character.

Program to count words in a sentence - I


```
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
   char sentence[200]; int i, length, numwords=0;
   cout << "Enter a string" << endl;
   gets(sentence) ;
   for (i =0; ; i++){
      if (sentence[i] == '\0') break;
   }
   length = i;
```

Program to count words in a sentence - II


```
//count words , and print each on  a different line of output
   for(i= 0; i <length ; i++){
      if (sentence[i] == ' '){
         numwords++;
         cout << endl;
         continue;
      }
      cout << sentence[i];
   }
   cout << endl;
   cout << endl << "Given sentence contains " << numwords << " words" << endl;
   return 0;
}
```

Does it work?

IIT Bombay

- The program does print each word on a separate line, but the word count is 1 less than the actual,
  - because the last word does not have a space after it!
- We should check for end of a word, by looking either for a blank space, or for the end of string.

```
if (sentence[i] == ' ' || sentence[i] == '\0' ){
        numwords++;
        . . .
}
```

- Find any other correction required

Summary

- We studied how to handle simple strings of characters.