Fundamentals of Computer Programming

Chapter 4
Array and Strings

Chere L. (M.Tech)
Lecturer, SWEG, AASTU

Outline

- Basic concepts of Array
- Types of Array
  - ✓ *One Dimensional Arrays*
  - ✓ *Multi-dimensional Arrays*
- Array declaration and initialization
- Accessing and processing Array Elements
- Basics of String
- String declaration and initialization
- String manipulation and operation
  - ✓Input/output, Copying, Comparing, concatenation, etc.
- String library functions and operators

Part II
Array of Character (Strings)

4) Basic of String

What is string?

- ✓ A **string** is a collection of characters .
- ✓ It is usually a meaningful sequence representing the name of an entity.
- ✓ Generally it is combination of two or more characters enclosed in **double quotes**.

- ✓ **Example:**

"Good Morning" // string with 2 words
"Mehak" // string with one word
"B.P." // string with letters and symbols
"" // an empty string

- ✓ The above examples are also known as **string literals**

4) Basic of String (cont'd)

String in C++

- ✓ C++ does not provide with a special data type to store strings.
  - *Thus we use arrays of the type char to store strings*
- ✓ Strings in C++ are always **terminated** using a **null character ('\0')**
- ✓ Strings can be *one dimensional or multi- dimensional* character arrays terminated by a null character ('\0')
- ✓ **String literals** are the values stored in the character array
- ✓ **`Example: "Hi there!"`**

===> would be stored in memory as shown:

H	i		t	h	e	r	e	!	\0

4.1) String Declaration (C Style)

- To declare a character array (string) the syntax is

  char stringName[size]; //One dimensional string

  char stringName[rSize] [cSize]; //Two dimensional string

  rsize: no of strings
  cSize: size of each string

- **Example:** char name[20];

  char stud_Name [30][20];

Note:

- Here **name and stud_name** is a character array or string capable of storing maximum of **19 characters and 570 characters** respectively.
- Since one character is reserved for storing **'\0',** the number of elements that can be stored in a 1D string is always **size-1**
- Incase of 2D string each row should ends with **'\0'** and the maximum number of characters that will stored is **total_size – row_Size**

4.1) String Declaration (C++ Style)

- In C++ a string can be declared with **string object** in addition to that of C-style declaration.
- **Example:**

```
string myString;
string city, country;        string address[10];
```

- **C-strings vs. string objects**

C-strings	string objects
Implemented as arrays of type char	Instance of string class
Terminated with the null character	Not terminated with null character
Compile-time allocation and determination of size	run-time allocation and undetermined size
Fixed size, but do not track their own size	Dynamic size and also track their own size

4.1) String Declaration (C++ Style)

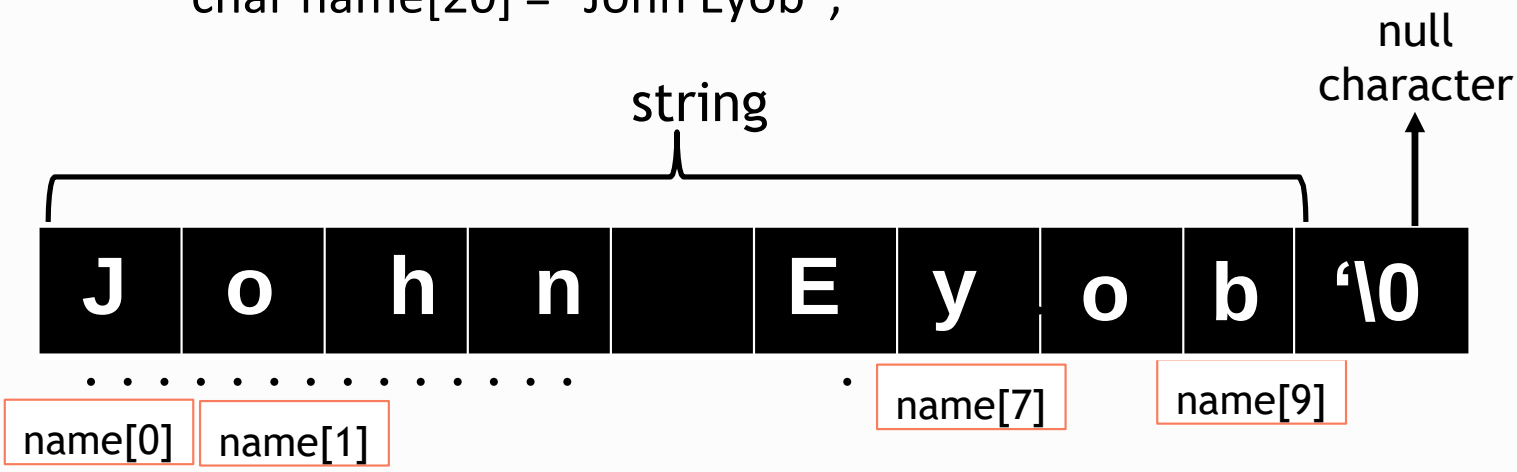
- Unlike C-style string, the **string class library** used to declared a string as regular variables (not as arrays), and they support:
  - ✓ The assignment operator =
  - ✓ Comparison operators ==, !=, etc
  - ✓ The + operator for concatenation
  - ✓ Type conversions from c-strings to string objects
  - ✓ A variety of other member functions

4.2) String initialization

(a) 1D String Initialization

- **Example 1:**

char name[20] = “John Eyob”;



*- The above string will have 9 characters and 1 space for the null. Thus size of **name** will be 10.*

- **Example 2:** **string** name = “John Eyob”;

4.2) String initialization (cont'd)

- **Example 2: omitting string size**

  - Like as we do in array string size can be omitted also

  char myAddress[] = "Addis Ababa, Ethiopia";

  ✓ *In this case the string is initialized to the mentioned string literal and it's size is the number of characters in the string literal plus null character. Here it is 20.*

  ✓ *The null character is automatically inserted at the end of the string.*

- **Example 3: initializing string character by character**

  char city [10] = {'A', 'd', 'a', 'm', 'a', '\0'};

  char myCity [ ] = {'D', 'i', 'r', 'e', 'd', 'e', 'w', 'a', '\0'};

Note: The '\0' has to be inserted by the programmer.

4.2) String initialization (cont'd)

- Some more examples of string initialization

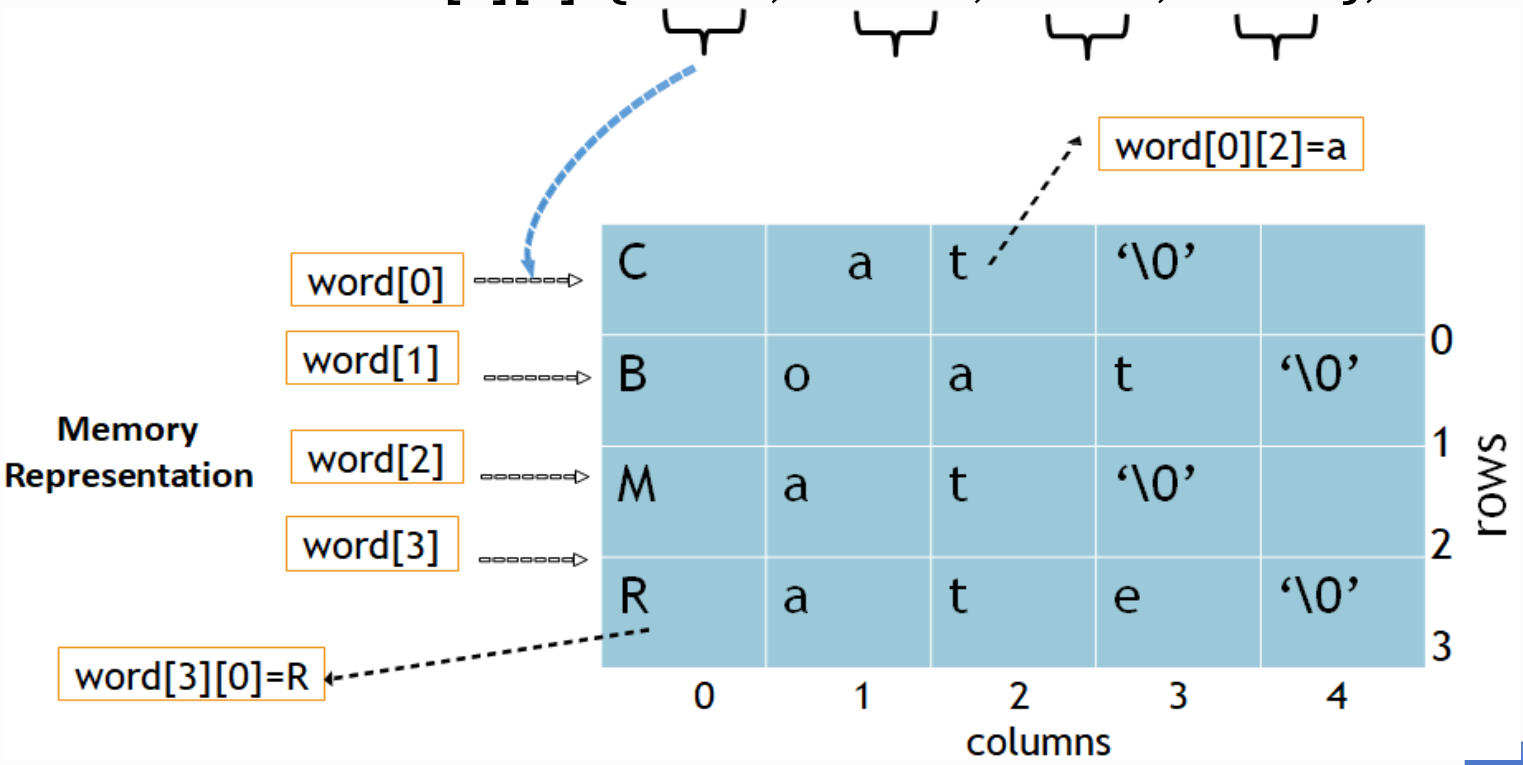
Initialization	Memory representation
char animal[7]="Lion";	L i o n '\0'
char location[]="Aksum City";	A k s u m C i t y '\0'
char serial_no[]="A011";	A 0 1 1 '\0'
char name [5] = "Gamechis";	//invalid, out of bound
char company[10] = "Ethiotel";	E t h i o t e l '\0'
char country [] = 'Ethioipia';	//invalid, string must enclosed within double quote

4.2) String initialization (cont'd)

(b) Initializing 2 D Strings - 2D string can be initialize as follows

- **Example 1:**

char word[4][5]={"Cat", "Boat", "Mat","Rate"};



4.2) String initialization (cont'd)

- **Example 2: Omitting string rowSize (number of strings)**

```
char name[][12] = {"Mr. Biniam", "Mr. Abush",
                   "Miss Nardos", "Mrs. Kidest",
                   "Dr. Andualem", "Eng. Yodit"};
```

- **#strings (rowSize) = 6**

2nd index

1st index

Columns

0	1	2	3	4	5	6	7	8	9	10	11	12	rows
M	**r**	**.**		**B**	**i**	**n**	**i**	**a**	**m**	**\0**			0
M	r	.		A	b	u	s	h	\0				1
M	i	s	s		N	a	r	d	o	s	\0		2
M	r	s	.		K	i	d	e	s	t	\0		3
D	r	.		A	n	d	u	a	l	e	m	\0	4
E	n	g	.		Y	o	d	i	t	\0			5

name[2][2]

name[5][7]

name[4][8]

4.2) String initialization (cont'd)

- **Example 3: initializing string objects**

```
string address = "Addis Ababa";

string name[12] = {"Mr. Biniam", "Mr. Abush",
                   "Miss Nardos", "Mrs. Kidest",
                   "Dr. Andualem", "Eng. Yodit"};
```

- **Example 4: initializing 2D strings character by character**

```
char myName[][6] = { {'C', 'H', 'A', 'L', 'A', '\0'},
                     {'B', 'O', 'N', 'S', 'A', '\0'}
                     {'H', 'A', 'G', 'O', 'S', '\0' };
```

4.2) String initialization (cont'd)

- **Example 6: initializing string after declaration**

```
char mystring [6];
        mystring[0] = 'H';          mystring[1] = 'e';
        mystring[2] = 'l';          mystring[3] = 'l';
        mystring[4] = 'o';          mystring[5] = '\0';
```

Note: Like wise 2D strings can be initialized after declaration.

- **Example 7: Invalid string initialization/assignment**

```
char mystring[6];
    mystring="Hello";                // not allowed
    mystring[] = "Hello";                //illegal
    mystring = { 'H', 'e', 'l', 'l', 'o', '\0' };  //neither would be valid
```

4.3) String input/output

- A string is displayed using a simple **cout<< stream** statement
- However, input a string or character array can be performed through any one of the following

No	Input method	Descriptions
1	**cin>> stream**	• Inputs a string without spaces • The **>> operator** stops input when it encounters a space • **Syntax: cin>>str;**
2	**get() function**	• Used to input either single character or a line of text with spaces • **Syntax 1: cin.get(ch);** where **ch** is a character • **Syntax 2: cin.get(str, n);** where **str** is string and **n** specify the size of string to be read.

4.3) String input/output (cont'd)

No	Input method	Descriptions
3	**gets() function**	• Can be used to input a single line of text including spaces. • As soon as the enter is pressed it stops input • **Syntax: gets(str);** where **str** is a string
4	**getline() function**	• Can be used to input multiple lines of text. • **Syntax: cin.getline(string, MAX, Delimiter)** were - **String** is the character array - **Max** is the maximum number of characters allowed input - **Delimeter** is the character which when encountered in the input stream stops the input

Note: *it is no needed to use loop to* ***input or display a string*** *unless the character array (string) is 2D and we need to read/print multiple strings.*

4.3) String input/output (cont'd)

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

int main(){
    char city[30];
    cout<<"\nEnter name of the cities\n";
    cout<<"city 1: ";
    cin>>city;
    cin.ignore();
    cout<<"The city you entered: "<<city<<endl;

    cout<<"\ncity 2: ";
    cin.get(city, 30);
    cin.ignore();
    cout<<"The city you entered: "<<city<<endl;

    cout<<"\ncity 3: ";
    gets(city);
    cin.ignore();
    cout<<"The city you entered: "<<city<<endl;

  return 0;
}
```

C:\Users\Habesh\Documents\Untitled2.exe

```
Enter name of the cities
city 1: Adama
The city you entered: Adama

city 2: Addis Ababa
The city you entered: Addis Ababa

city 3: Diredewa, Ethiopia.

The city you entered: Diredewa, Ethiopia.

--------------------------------
Process exited af
Press any key to
```

C:\Users\Habesh\Documents\Untitled2.exe

```
Enter name of the cities
city 1: Addis Ababa
The city you entered: Addis

city 2: The city you entered: Ababa

city 3: Adama

The city you entered: Adama

--------------------------------
Process exited after 21.21 seconds w
Press any key to continue . . .
```

4.3) String input/output (cont'd)

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

int main()
{
    char address[30];

    cout<<"\nEnter Address: ";
    cin.getline(address, 30, '?');
    cout<<"\nYou entered: "<<address<<endl;

  return 0;
}
```

```
C:\Users\Habesh\Documents\

Enter Address:
Addis Ababa,
Aksum, Gonder,
Asosa?

You entered:
Addis Ababa,
Aksum, Gonder,
```

```
C:\Users\Habesh\Documents\Untitle

Enter Address:
Addis Ababa,
Diredawa?

You entered:
Addis Ababa,
Diredawa

--------------------------
Process exited after 18.52 s
Press any key to continue .
```

Note:

- ✓ *The getline function continues to input the string until either the **maximum number of characters** are input or it **encounters the delimiter** character whichever comes first.*

4.4) String Operation/manipulations

- **Assignment/copy and comparison operation**

✓ In C-style, strings cannot be copied or compared using the simple assignment or comparison operator as follow.

```
char str1[20],   str2[20];
str2=str1;          // Not allowed
if(str1==str2)      //Not allowed
        { ..... }
```

✓ However, using the C++ string objects the above two string operations are valid

```
str2=str1;          if(str1==str2)     //both are valid
```

4.4) String Operation/manipulations (cont'd)

- **Assignment/copy and comparison operation**
  - ✓ In C-style, strings cannot be copied or compared using the simple assignment or comparison operator as follow.

```
char str1[20],   str2[20];
str2=str1;          // Not allowed
if(str1==str2)      //Not allowed
        { ..... }
```

  - ✓ However, using the C++ string objects the above two string operations are valid

```
str2=str1;          if(str1==str2)     //both are valid
```

4.4) String Operation/manipulations (cont'd)

- **Other string operations**
  - ✓ Find the string length
  - ✓ Search string or substring
  - ✓ Characters case conversion
  - ✓ Reverse or swap string
  - ✓ Concatenating strings
  - ✓ String tokenization etc.
  - ✓ Modifying (replace) string
- The above mentioned string manipulations can be performed either through hard coding or using library functions

4.4) String Manipulations and Library Functions

- **Here below list of string manipulation library functions**

String operations	Function	Description
String copying	strcpy(str1, str2);	Copies string str2 (source string) into the character array str1 (destination string). The value of str1 is returned.
	strncpy(str1, str2, size_t n);	Copies at most n characters of string s2 into the array s1. The value of s1 is returned.
String concatenation	strcat (str1, str2);	Appends string s2 to string s1. The value of s1 is returned.
	strncat (str1, str2, size_t n);	Appends at most n characters of string s2 to string s1. The value of s1 is returned.

4.4) String Library Functions (cont'd)

- **Here below list of string manipulation library functions**

String operations	Function	Description
String comparison	strcmp(str1, str2);	Compares string str1 with string str2. The function returns a value of • zero, if str1 is equal to str2 • less than zero, if str1 is less than str2 • greater than zero, if str1 greater than str2
	strncmp(str1, str2, size n);	Compares up to n characters of string str1 with string str2. It works in the fashion as strcmp().
	int stricmp(str1, str2);	Compares string str1 with string str2 in regardless of their cases (upper case or lower case.
	strnicmp(str1, str2, size n);	Compares up to n characters of string str1 with string str2 in regardless of their cases

4.4) String Library Functions (cont'd)

- **Here below list of string manipulation library functions**

String operations	Function	Description
String length	strlen(str);	Determines the length of string str. The number of characters preceding the terminating null character is returned.
Looking for string / character Occurrence	strch(str1, ch);	Returns a the first left occurrence of character **ch** in string str1.
	strrch(str1, ch);	Returns a the first right occurrence of character **ch** in string str1 .
	strstr(str1, str2);	Returns a the first occurrence of string str2 in string str1.
String case conversion	strupr(str1)	Converts lowercase characters in strings to uppercase
	strlwr(str1)	Converts uppercase characters in strings to lowercase

4.4) String Library Functions (cont'd)

- **Here below list of string manipulation library functions**

String operations	Function	Description
Others	strspn(str1, str2)	finds up at what length two strings are identical
	strrev(str)	Reversing all characters of a string
	strtok(str1, s2);	A sequence of calls to strtok breaks string str1 into "tokens"—logical pieces such as words in a line of text—delimited by characters contained in string s2. The first call contains str1 as the first argument, and subsequent calls to continue tokenizing the same string contain NULL as the first argument
	strset(str, ch), strnset(str, ch, 5)	Repalcae character(s) of string to a given character

Note: These are some of the library functions and Many More are available

Relational Operators and library functions supported by C++ String objects

Operator	Working
=	Assignment
+	joining two or more strings
+=	concatenation and assignment
==	Equality
!=	Not equal to
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
[]	Subscription
<<	insertion
>>	Extraction

functions	Descriptions
append()	appends a part of a string to another string
assign()	assigns a partial string
at()	obtains character stored at a specified location
begin()	returns a reference to the start of the string
capacity()	gives the total element that can be stored
compare()	compares a string against the invoking string
empty()	returns true if the string is empty
end()	returns a reference to the end of the string
erase()	removes character as specified
find()	searches for the occurrence of a specified substring
swap()	swaps the given string with the invoking on

Correspondence between the C-library and the C++ string class/object

C Library Functions	C++ string operators/methods
strcpy	= (the assignment operator)
strcat	+= (assign+concat operator)
strcmp	= =, !=, <, >, <=, >=
strchr, strstr strrchr	.find( ) method .rfind( ) method
strlen	.size( ) or .length( ) methods

4.4) String Library Functions (cont'd)

Character handling library functions of ctype.h

Prototype	Description
isdigit(c)	Returns true if c is a digit and false otherwise
isalpha(c)	Returns true if c is a letter and false otherwise
isalnum(c)	Returns true if c is a digit/letter and false otherwise
isxdigit(c)	Returns true if c is a hexadecimal digit and false otherwise
islower(c)	Returns true if c is a lowercase letter and false otherwise
isupper(c)	Returns true if c is an uppercase letter; false otherwise
tolower(c), toupper(c)	If c is an uppercase letter, it returns c as a lowercase letter. Otherwise, leave the character/string unchanged and vice versa
isgraph(c)	Returns true if c is a printing character other than space (' ')
isspace(c)	Returns true if c is a white-space, newline ('n'), space (' '), form feed ('f'), carriage return ('r'), horizontal tab ('t'), or vertical tab ('v') and false otherwise
iscntrl(c)	Returns true if c is a control character and false otherwise
ispunct(c)	Returns true if c is a printing character other than a space, a digit, or a letter and false otherwise
isprint(c)	Returns true value if c is a printing character including space (' ')

- **Example 1: string length**

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

int main()
{
    char word[80];
    cout<<"Enter a string: ";
    gets(word);
    int i;
        //Loop to find length
    for(i=0; word[i]!='\0'; i++);
    cout<<"Length of string: "<<i<<endl;

  return 0;
}
```

(a) Hard coding

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

int main()
{
    char word[80];
    cout<<"Enter a string: ";
    gets(word);

    cout<<"Length of string: ";
    cout<<strlen(word)<<endl;

  return 0;
}
```

(b) Using strlen() library function

```
C:\Users\Habesh\Documents\Untitled2.exe
Enter a string: Addis Ababa
Length of string: 11

--------------------------------
Process exited after 4.488 seconds
Press any key to continue . . .
```

- **Example 2: copying and compare (Hard coding)**

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

int main()
{
    char str1[20], str2[20];
    cout<<"Enter first string: ";
    gets(str1);
    int i;
    for(i=0; str1[i]!='\0';i++)
        str2[i]=str1[i];
    str2[i]='\0';   // to terminate str2 manually

    cout<<"\nCopied String : "<<str2<<endl;

  return 0;
}
```

(a) Copying string

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

int main()
{
    char str1[20], str2[20];
    int flag=0;
    cout<<"Enter first string: ";
    gets(str1);
    cout<<"Enter second string: ";
    gets(str2);
    for(int i=0; str1[i]!='\0'; i++)
        if(str1[i] != str2[i])
        {
            flag++;
            break;
        }
    if (flag==0)
        cout<<"\n strings are equal";
    else
        cout<<"\n strings are not equal";

    return 0;
}
```

(b) String comparison

- **Example 3: copying, concatenate and compare using library functions**

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

int main()
{
    char str1[20], str2[20];
    cout<<"Enter first string: ";   gets(str1);
    cout<<"Enter second string: ";  gets(str2);

    strncpy(str1,str2,5);
    cout<<"copyied nth string: "<<str1<<endl;
    strncat(str1, str2, 5);
    cout<<"nth string concatenation: "<<str1<<endl;

    cout<<"nth string comparision: ";
    if (strncmp(str1,str2,1) == 1)
        cout<<str1<<" > "<<str2<<endl;
    else if (strncmp(str1,str2,1) < 1)
            cout<<str1<<" < "<<str2<<endl;
    else
        cout<<str1<<" = "<<str2<<endl;

    return 0;
}
```

copying and concatenating the 1st nth string

Comparing the 1st nth characters of strings

(b) String comparison

- **Example 4: more on string manipulations**

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

int main()
{
    char str1[20], str2[20];
    cout<<"Enter first string: ";
    gets(str1);
    cout<<"Enter second string: ";
    gets(str2);

    cout<<"\n\n 1st string in lowercase: "<<strlwr(str1);
    cout<<"\n 2nd string in upercase: "<<strupr(str2);

    cout<<"\n Concatenation of the two strings is: ";
    cout<<strcat(str1, str2);
    cout<<"\n Reverse of the strings is: ";
    cout<<strrev(str1);

    return 0;
}
```

```
C:\Users\Habesh\Documents\Untitled2.exe
Enter first string: Addis
Enter second string: Ababa


 1st string in lowercase: addis
 2nd string in upercase: ABABA
 Concatenation of the two strings is: addis ABABA
 Reverse of the strings is: ABABA sidda
--------------------------------
Process exited after 7.103 seconds with return value 0
Press any key to continue . . .
```

String case conversion

Reverse string

- **Example 5: string tokenization**

```
using namespace std;
#include <iostream>
#include <cstring>
#include <string>

int main(){
    char sentence[] = "This is a sentence with 7 tokens";
    char *tokenPtr;

    cout<<"The string to be tokenized is:\n" << sentence
        << "\n\nThe tokens are:\n\n";

    // begin tokenization of sentence
    tokenPtr = strtok( sentence, " " );

    while ( tokenPtr != NULL ) {
      cout << tokenPtr << '\n';
      tokenPtr = strtok( NULL, " " );  // get next token
     }
    cout << "\nAfter strtok, sentence = " << sentence << endl;

    return 0;
}
```

```
C:\Users\Habesh\Documents\Untitled2.exe
The string to be tokenized is:
This is a sentence with 7 tokens

The tokens are:

This
is
a
sentence
with
7
tokens

After strtok, sentence = This

--------------------------------
Process exited after 0.1823 seconds
Press any key to continue . . .
```

- **Example 6:** **Program to display the words which start with a capital 'A'**

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

int main()
{
    char word[20][25];
    int n, i;
    cout<<"\nNo of word you wish to input: ";
    cin>>n;
    cin.ignore();

    for(int i=0;i<n;i++)
    {
        cout<<"\n "<<i+1<<": ";
        gets(word[i]);
    }
    cout<<"\n Displaying words starting with 'A'";
    for (i=0; i<n;i++)
        if(word[i][0]=='A') //checking 1st letter of each word

    cout<<"\n"<<word[i];

    return 0;
}
```

```
C:\Users\Habesh\Documents\Untitled2.exe

No of word you wish to input: 4

 1: Adama

 2: Diredawa

 3: Gonder

 4: Aksum

 Displaying words starting with æAÆ
Adama
Aksum
--------------------------------
Process exited after 16.35 seconds with
Press any key to continue . . .
```

Practical Exercises 2 - Strings

1. Write a program to count total number of vowels and consonants present in a string.
2. Design a program to find the frequency of characters within string and display character with largest and smallest frequency respectively.
3. Write a program that find the frequency of vowel, consonant, digit and special character
4. Design a program to check either the word is palindrome or not using loop.
5. Write a program to remove non-alphabet character from string
6. Write a program to store and print the names of your two favorite television programs. Store these programs in two character arrays. Initialize one of the strings (assign it the first program's name) at the time you declare the array. Initialize the second value in the body of the program with the **strcpy()** function.
7. Write an application that inputs a line of text and outputs the text twice, once in all uppercase and once in all lowercase letters.

Reading Resources/Materials

Chapter 13:

- ✓ Diane Zak; An Introduction to Programming with C++ (8th Edition), 2016 Cengage Learning

Chapter 8:

- ✓ Walter Savitch; Problem Solving With C++ [10th edition, University of California, San Diego, 2018

Link:

- ✓ https://www.w3schools.in/category/cplusplus-tutorial/

Thank You
For Your Attention!!