

**CCS0007**

Computer Programming 2 for IT

s

EXERCISE

2

Character and String Manipulation

|  |  |
| --- | --- |
| CAPILI, MARK ANGELO | Joie Ann Maghanoy |
| 1/21/20 | 1/21/20 |

1. **OBJECTIVES**

At the end of this Laboratory exercise, the students must be able to:

* Create a program that applies different C-String functions.
* Create a program process C-String values using user-defined functions

1. **BACKGROUND INFORMATION**

A string stored as an array of characters terminated with ‘\0’.

## Table 1: Some Predefined C-String Functions in <cstring>

|  |  |
| --- | --- |
| FUNCTION | DESCRIPTION |
| **strcpy(Target\_String\_Var,Src\_String)** | **Copies the C-string value Src\_String into the C-string variable Target\_String\_Var** |
| strcpy(Target\_String\_Var,Src\_String,limit) | The same as the two argument strcpy except that at most Limit characters are copied |
| **strcat(Target\_String\_Var,Src\_String)** | **Concatenates the C-String value Src\_String onto the end of C-string in the C-string variable Target\_String\_Var** |
| strcat(Target\_String\_Var,Src\_String, Limit) | The same as the two argument strcat except that at most Limit characters are appended. |
| **strlen(Src\_String)** | **Returns an integer equal to the length of Src\_String. (The null character, ‘\0’, is not counted in the length.** |
| **strcmp(String\_1, String\_2)** | **Returns 0 if String\_1 and String\_2 are the same. Returns a value < 0 if String\_1 is less than String\_2. Returns a value > 0 if String\_1 is greater than String\_2 (that is, returns a nonzero value if String\_1 and String\_2 are different). The order is lexicographic.** |
| strcmp(String\_1, String\_2, Limit) | The same as the two-argument strcmp except that at most Limit characters are compared. |

**Table 2 : Some Predefined character manipulating functions in <cctype>**

|  |  |
| --- | --- |
| FUNCTION | DESCRIPTION |
| toupper(Char\_Exp) | Returns the uppercase version of Char\_Exp (as value of type int). |
| tolower(Char\_Exp) | Returns the lowercase version of Char\_Exp (as value of type int). |
| isupper(Char\_Exp) | Returns true provided Char\_Exp is an uppercase letter; otherwise, returns false. |
| islower(Char\_Exp) | Returns true provided Char\_Exp is an lowercase letter; otherwise, returns false. |
| isalpha(Char\_Exp) | Returns true provided Char\_Exp is a letter of the alphabet; otherwise return false. |
| isdigit(Char\_Exp) | Returns true provided Char\_Exp is one of the digits ‘0’ through ‘9’; otherwise, returns false. |
| isalnum(Char\_Exp) | Returns true provided Char\_Exp is either a letter or a digit; otherwise, returns false. |
| isspace(Char\_Exp) | Returns true provided Char\_Exp is a whitespace character, such as the blank or newline character, otherwise, return false. |
| ispunct(Char\_Exp) | Returns true provided Char\_Exp is a printing character other than whitespace, a digit, or a letter; otherwise return false. |
| isprint(Char\_Exp) | Returns true provided Char\_Exp is a printing characters includes blank space; otherwise returns false. |
| isgraph(Char\_Exp) | Returns true provided Char\_Exp is a printing characters; otherwise returns false. |
| isctrl(Char\_Exp) | Returns true provided Char\_Exp is a control character; otherwise, return false. |

1. **EXPERIMENTAL PROCEDURE**

**Instructions:**

Copy your source codes to be pasted in this document as well as a screen shot of your running output.

Upload your document using the link provided in your canvas.

**ACTIVITY2.1: Compare two strings**

Create a program that will compare two input strings using strcmp( ).

EXAMPLE PROGRAM OUTPUT:

|  |  |
| --- | --- |
|  |  |
|  |  |

**SOURCE CODE: SAMPLE OUTPUTS:**

|  |  |
| --- | --- |
| **#include <iostream>**  **#include <cstring>**  **using namespace std;**  **int main() {**  **//For UI**  **char str1[256];**  **char str2[256];**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "STRING COMPARE \n";**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "Enter a first word: ";**  **cin.getline(str1,256);**  **cout << "Enter a second word: ";**  **cin.getline(str2,256);**  **// algo for determining the result**  **if(strcmp(str1,str2) == 0) {**  **cout << "Equal";**  **} else if (strcmp(str1,str2) == -1) {**  **cout << "Negative";**  **} else if (strcmp(str1,str2) == 1) {**  **cout << "Positive";**  **}**  **return 0;**  **}** |  |

**ACTIVITY 2.2: Copying strings**

Create a program that will copy a string from one variable to another using the strcpy( ) function.

EXAMPLE PROGRAM OUTPUT:

|  |  |
| --- | --- |
|  |  |

**SOURCE CODE: SAMPLE OUTPUTS:**

|  |  |
| --- | --- |
| **#include <iostream>**  **#include <cstring>**  **using namespace std;**  **int main() {**  **//For UI**  **char str1[256];**  **char str2[256];**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "STRING COPY \n";**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "Enter a first word: ";**  **cin.getline(str1,256);**  **cout << "Enter a second word: ";**  **cin.getline(str2,256);**  **strcpy (str1,str2);**  **//outputing the new value of str1**  **cout << "The New String Value for str1: " << str1;**  **return 0;**  **}** |  |

**ACTIVITY 2.3: Concatenating strings**

Create a program that will concatenate two strings.

|  |  |
| --- | --- |
|  |  |

**SOURCE CODE: SAMPLE OUTPUTS:**

|  |  |
| --- | --- |
| #include <iostream>  #include <cstring>  using namespace std;  int main() {  //For UI  char str1[256];  char str2[256];  cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;  cout << "STRING CONCATENATION \n";  cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;  cout << "Enter a first word: ";  cin.getline(str1,256);  cout << "Enter a second word: ";  cin.getline(str2,256);  strcat(str1,str2);  //outputing the new value of str1  cout << "The New String Value for str1: " << str1;  return 0;  } |  |

**ACTIVITY 2.4: Palindrome**

Convert a program that will determine if the given word input is a palindrome using C-String functions.

NOTE: Palindromes are words that are read the same way either left to right or right to left.

EXAMPLE PROGRAM OUTPUT:

|  |  |
| --- | --- |
|  |  |
|  |  |

**SOURCE CODE: SAMPLE OUTPUTS:**

|  |  |
| --- | --- |
| **#include <iostream>**  **#include <cstring>**  **using namespace std;**  **int main() {**  **//For UI**  **int len,temp;**  **char str1[20];**  **char str2[20];**  **bool isPal;**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "PALINDROME\n";**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "Enter a word: ";**  **cin.getline(str1,20);**  **len = strlen(str1);**  **//outputting the word backwards**  **cout << "The backward of the word is: ";**  **for (int i = 0; i < len ;i++){**  **temp = len - i - 1;**  **cout << str1[temp];**  **}**    **//determining if the word is palindrome**  **for(int i=0 ;i < len ;i++){**  **if(str1[i] != str1[len-i-1]){**  **isPal = 1;**  **break;**  **}**  **}**  **cout << endl;**  **if (isPal) {**  **cout << str1 << " is not palindrome";**  **} else {**  **cout << str1 << " is a palindrome";**  **}**  **return 0;**  **}** |  |

**ACTIVITY 2.5: Uppercase**

Create a program that will accept an input string. Display the same string in all capital form.

EXAMPLE PROGRAM OUTPUT:

|  |  |
| --- | --- |
|  |  |
|  |  |

**SOURCE CODE: SAMPLE OUTPUTS:**

|  |  |
| --- | --- |
| **#include <iostream>**  **#include <cstring>**  **using namespace std;**  **int main()**  **{**  **// making the ui**  **int len;**  **char str1[256];**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "CAPITALIZING EACH WORD" <<endl;**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "Please enter a sentence: ";**  **cin.getline(str1, 256);**  **//gettting the length**  **len = strlen(str1);**  **str1[0] = toupper(str1[0]);**    **// making the letter uppercase before space**  **for (int i = 1; i < len; i++)**  **{**  **if ( str1[i - 1] == ' ' )**  **str1[i] = toupper( str1[i] );**  **else**  **// to lower the rest of the word**  **str1[i] = tolower(str1[i]);**  **}**  **//outputing the results**  **cout << str1 << endl;**  **return 0;**  **}** |  |

**ACTIVITY 2.6: Strings to words**

Create a program that will ask the user to enter some string. The program will split the string in to word and display in reverse vertical order.

EXAMPLE PROGRAM OUTPUT:

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| **#include <iostream>**  **#include <cstring>**  **using namespace std;**  **int main()**  **{**  **//making the ui**  **string str;**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "STRINGS TO WORDS" <<endl;**  **cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**  **cout << "Please enter a sentence: ";**  **getline(cin,str);**  **int i = str.length() - 1;**  **int start, end = i + 1;**  **string result = "";**  **//string to words algo**  **for(i; i >=0; i--) {**  **if(str[i] == ' ')**  **{**  **start = i + 1;**  **while(start != end){**  **result += str[start++];**  **}**  **//make every word go to new line**  **result += " \n";**  **end = i;**  **}**  **}**  **start = 0;**  **while(start != end){**  **result += str[start++];**  **}**  **//outputing the result**  **cout << result;**    **return 0;**  **}** |  |

1. **Assessment**

|  |  |
| --- | --- |
| Department | Information Technology |
| Subject Code | CCS0007 |
| Description | COMPUTER PROGRAMMING 2 FOR IT |
| Term/Academic Year |  |

|  |  |
| --- | --- |
| Topic | Character and String Functions |
| Lab Activity No | 2 |
| Lab Activity | **Character and String Manipulation** |
| CLO | **3** |

**Note: The following rubrics/metrics will be used to grade students’ output in the lab exercise.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Trait** | **(Excellent)** | **(Good)** | **(Fair)** | **(Poor)** |
| **Requirement Specification(30pts)** | Able to identify correctly all input and output and provide alternative.  **(28-20pts)** | Able to identify correctly all input and output  **(25-17pts)** | Able to identify only one input or output  **(22-14pts)** | Unable to identify any input and output  **(20-11pts)** |
| **Data type(20pts)** | Able to apply required data type or data structure and produce correct results **(18-20pts)** | Able to apply required data type or data structure and produce partially correct results **(15-17pts)** | Able to identify required data type or data structure but does apply correctly **(12-14pts)** | Unable to identify required data type  **(9-11pts)** |
| **Input Validation(20pts)** | The program works and meets all specifications. Does exception al checking for errors and out-of- range data **(18-20pts)** | The program works and meets all specifications. Does some checking for errors and out of range data **(15-17pts)** | The program produces correct results but does not display correctly Does not check for errors and out of range data **(12-14pts)** | The program produce s incorrect results **(9-11pts)** |
| **Free from syntax, logic, and runtime errors (10pts)** | Unable to run program **(10pts)** | Able to run program but have logic error **(8-9pts)** | Able to run program correctly without any logic error and display inappropriate output **(6-7pts)** | Able to run program correctly without any logic error and display appropriate output **(5pts)** |
| **Delivery (10pts)** | The program was delivered on time **(10pts)** | The program was delivered after 5 minutes from the time required. **(8-9pts)** | The program was delivered after 10 minutes from the time required. **(6-7pts)** | The program was delivered after 15 (or more) minutes from the time required. **(5pts)** |
| **Use of Comments**  **(10pts)** | Specific purpose is noted for each function, control structure, input requirements, and output results. **(10pts)** | Specific purpose is noted for each function and control structure. **(8-9pts)** | Purpose is noted for each function. **(6-7pts)** | No comments included. **(5pts)** |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.1 |
| **Lab Activity** | Compare two strings |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.2 |
| **Lab Activity** | Copying strings |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.3 |
| **Lab Activity** | Concatenating strings |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.4 |
| **Lab Activity** | Palindrome |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.5 |
| **Lab Activity** | Uppercase |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |

|  |  |
| --- | --- |
| **Topic** | Character and String Manipulation |
| **Lab Activity No** | 2.6 |
| **Lab Activity** | Strings to words |
| **CLO** | 3 |
| Requirement Specification (30pts) |  |
| Data type (20pts) |  |
| Input Validation (20pts) |  |
| Free from syntax, logic, and runtime errors (10pts) |  |
| Delivery (10pts) |  |
| Use of Comments (10pts) |  |
| **TOTAL** |  |