Consolidated Assignment 6 Report

This report contains the graded results for the newest of each exercise submitted to the assignment checker prior to 5/14/2020 1:12:10 PM PDT.

Student Name: Shaun Chemplavil
Student ID: U08713628
Contact e-mail: shaun.chemplavil@gmail.com
C/C++ Programming I (Section 146359)

Submitted:
   Exercise 0: 4/23/2020 3:19:23 PM PDT
   Exercise 1: 4/23/2020 4:23:07 PM PDT
   Exercise 2: 5/8/2020 8:36:17 AM PDT
   Exercise 3: 5/8/2020 8:36:39 AM PDT

Score (out of 20 possible): 17

THIS WAS SENT FROM A NOTIFICATION-ONLY ADDRESS THAT CANNOT ACCEPT INCOMING MAIL. For help please contact the instructor at the email address provided on the "Announcements" page of the course website. The assignment checker DOES NOT GRADE your submissions but merely reports on issues so you can correct them and resubmit, thereby avoiding unnecessary credit loss. ALL GRADING IS DONE MANUALLY BY THE INSTRUCTOR after the assignment deadline based solely upon the NEWEST submission of each exercise. BE WARY of correcting minor issues after the deadline because a late deduction will usually be much greater than a minor issue deduction.

```
   From:  <mailto:shaun.chemplavil@gmail.com>
   Subject: C1A6E0_U08713628
   Submitted: 4/23/2020 3:19:23 PM PDT
   Course: C/C++ Programming I (Section 146359)
   Student's name: Shaun Chemplavil
   Contact email: shaun.chemplavil@gmail.com
   Student ID: U08713628
   Assignment 6, Exercise 0
   Exercise point value: 6
   File submitted:
      C1A6E0_Quiz.txt
```

NOTE: The assignment checker does not check the correctness of quiz answers for this assignment.

Your submission has been accepted and will be graded manually by the instructor. You may resubmit it as many times as you wish before the assignment deadline. BE WARY of correcting minor issues after the deadline because a late deduction will usually be much greater than a minor issue deduction.

-3

```
Shaun Chemplavil U08713628
shaun.chemplavil@gmail.com
C/C++ Programming I : Fundamental Programming Concepts
146359 Raymond L. Mitchell, Jr., M.S.
04/23/2020
C1A6E0_Quiz.txt
Answers to Quiz

1. A <---D
2. D
3. A <---C
4. B
5. B <---E
6. A
```

THIS WAS SENT FROM A NOTIFICATION-ONLY ADDRESS THAT CANNOT ACCEPT INCOMING MAIL. For help please contact the instructor at the email address provided on the "Announcements" page of the course website. The assignment checker DOES NOT GRADE your submissions but merely reports on issues so you can correct them and resubmit, thereby avoiding unnecessary credit loss. ALL GRADING IS DONE MANUALLY BY THE INSTRUCTOR after the assignment deadline based solely upon the NEWEST submission of each exercise. BE WARY of correcting minor issues after the deadline because a late deduction will usually be much greater than a minor issue deduction.

```
   From: Shaun Chemplavil <mailto:shaun.chemplavil@gmail.com>
   Subject: C1A6E1_U08713628
   Submitted: 4/23/2020 4:23:07 PM PDT
   Course: C/C++ Programming I (Section 146359)
   Student's name: Shaun Chemplavil
   Contact email: shaun.chemplavil@gmail.com
   Student ID: U08713628
   Assignment 6, Exercise 1
   Exercise point value: 4
   Files submitted:
      C1A6E1_main.c
      C1A6E1_MyStrlen.c

"Compile-time" results:
   No "compile-time" issues;
"Run-time" results:
   Program ran - No errors detected during preliminary testing (SEE ATTACHMENT);
```

```
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 04 / 23 / 2020
// C1A6E1_main.c
// Win10
// Visual C++ 19.0
//
// This program prompts the user to enter a string and returns the length
// using strlen and MyStrlen
//

#include <stdio.h>
#include <string.h>

#define LENGTH  129

size_t MyStrlen(const char *s1);

int main(void)
{
   char input[LENGTH];

   // get the users string
   printf("Enter a string: ");
   fgets(input, LENGTH, stdin);

   // remove newline character from string
   input[strcspn(input, "\n")] = '\0';

   // display results
   printf("strlen(\"%s\") returned %d\n", input, (int)strlen(input));
   printf("MyStrlen(\"%s\") returned %d\n", input, (int)MyStrlen(input));

   return 0;
}
```

```
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 04 / 23 / 2020
// C1A6E1_MyStrlen.c
// Win10
// Visual C++ 19.0
//
// This function calculates the length of a string
//

#include <stddef.h>

size_t MyStrlen(const char *s1)
{
   const char * const START = s1;

   // Increment pointer until the end of s1 is reached
   while (*s1)
      s1++;

   //  Determine the total length of input string
   return (size_t)(s1 - START);
}
```

```
********** C1 ASSIGNMENT 6 EXERCISE 1 AUTOMATIC PROGRAM RUN RESULTS **********

************* THE RESULTS BELOW HAVE BEEN PARTIALLY CHECKED AND **************
*************   NO ERRORS WERE FOUND.  HOWEVER, THIS DOES NOT   **************
*************  NECESSARILY MEAN THAT THERE ARE NO ERRORS.  THE  **************
*************  INSTRUCTOR WILL DO A MORE THOROUGH CHECK DURING  **************
*************                  MANUAL GRADING.                  **************


------------------------------ START OF 1ST RUN ------------------------------

Enter a string: a
strlen("a") returned 1
MyStrlen("a") returned 1

------------------------------- END OF 1ST RUN -------------------------------

------------------------------ START OF 2ND RUN ------------------------------

Enter a string: HELLO
strlen("HELLO") returned 5
MyStrlen("HELLO") returned 5

------------------------------- END OF 2ND RUN -------------------------------

------------------------------ START OF 3RD RUN ------------------------------

Enter a string: Programming is fun
strlen("Programming is fun") returned 18
MyStrlen("Programming is fun") returned 18

------------------------------- END OF 3RD RUN -------------------------------

------------------------------ START OF 4TH RUN ------------------------------

Enter a string:
strlen("") returned 0
MyStrlen("") returned 0

------------------------------- END OF 4TH RUN -------------------------------

------------------------------ START OF 5TH RUN ------------------------------

Enter a string: &=#~!
strlen("&=#~!") returned 5
MyStrlen("&=#~!") returned 5

------------------------------- END OF 5TH RUN -------------------------------

------------------------------ START OF 6TH RUN ------------------------------

Enter a string: aaaaaaaaaaaaaaa
strlen("aaaaaaaaaaaaaaa") returned 15
MyStrlen("aaaaaaaaaaaaaaa") returned 15

------------------------------- END OF 6TH RUN -------------------------------

------------------------------ START OF 7TH RUN ------------------------------

Enter a string: The cat in the fiddle, The cow jumped over the moon.
```

```
strlen("The cat in the fiddle, The cow jumped over the moon.") returned 52
MyStrlen("The cat in the fiddle, The cow jumped over the moon.") returned 52

------------------------------ END OF 7TH RUN ------------------------------
```

THIS WAS SENT FROM A NOTIFICATION-ONLY ADDRESS THAT CANNOT ACCEPT INCOMING MAIL. For help please contact the instructor at the email address provided on the "Announcements" page of the course website. The assignment checker DOES NOT GRADE your submissions but merely reports on issues so you can correct them and resubmit, thereby avoiding unnecessary credit loss. ALL GRADING IS DONE MANUALLY BY THE INSTRUCTOR after the assignment deadline based solely upon the NEWEST submission of each exercise. BE WARY of correcting minor issues after the deadline because a late deduction will usually be much greater than a minor issue deduction.

```
   From: Shaun Chemplavil <mailto:shaun.chemplavil@gmail.com>
   Subject: C1A6E2_U08713628
   Submitted: 5/8/2020 8:36:17 AM PDT
   Course: C/C++ Programming I (Section 146359)
   Student's name: Shaun Chemplavil
   Contact email: shaun.chemplavil@gmail.com
   Student ID: U08713628
   Assignment 6, Exercise 2
   Exercise point value: 4
   Files submitted:
      C1A6E2_MyStrcmp.c
      C1A6E2_main.c

"Compile-time" results:
   No "compile-time" issues;
"Run-time" results:
   Program ran - No errors detected during preliminary testing (SEE ATTACHMENT);
```

```
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 05 / 08 / 2020
// C1A6E2_main.c
// Win10
// Visual C++ 19.0
//
// This program prompts the user to enter 2 strings and compare them
// using strcmp and MyStrcmp
//

#include <stdio.h>
#include <string.h>

#define LENGTH  129

int MyStrcmp(const char *s1, const char *s2);

int main(void)
{
   char input1[LENGTH], input2[LENGTH];

   // get the users strings
   printf("Enter first string: ");
   fgets(input1, LENGTH, stdin);

   printf("Enter second string: ");
   fgets(input2, LENGTH, stdin);

   // remove newline character from strings
   input1[strcspn(input1, "\n")] = '\0';
   input2[strcspn(input2, "\n")] = '\0';

   // display results
   printf(
      "strcmp(\"%s\",\"%s\") returned %d\n",
      input1, input2, strcmp(input1, input2));
   printf(
      "MyStrcmp(\"%s\",\"%s\") returned %d\n",
      input1, input2, MyStrcmp(input1, input2));

   return 0;
}
```

```
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 05 / 08 / 2020
// C1A6E2_MyStrcmp.c
// Win10
// Visual C++ 19.0
//
// This function compare the contents of two strings identified to the pointers
// passed by the user
// Will return:  < 0  when s1 < s2
//                 0  when s1 = s2
//               > 0  when s1 > s2
//

int MyStrcmp(const char *s1, const char *s2)
{
   // Increment pointers until the s1 does not equal s2
   // AND ensure we do not increment past null character '\0'
   while (*s1 == *s2 && *s1 != '\0')
   {
      ++s1;
      ++s2;
   }

   return (int)(*s1 - *s2);
}
```

```
********** C1 ASSIGNMENT 6 EXERCISE 2 AUTOMATIC PROGRAM RUN RESULTS **********

************* THE RESULTS BELOW HAVE BEEN PARTIALLY CHECKED AND **************
*************   NO ERRORS WERE FOUND.  HOWEVER, THIS DOES NOT   **************
*************  NECESSARILY MEAN THAT THERE ARE NO ERRORS.  THE  **************
*************  INSTRUCTOR WILL DO A MORE THOROUGH CHECK DURING  **************
*************                  MANUAL GRADING.                  **************


------------------------------ START OF 1ST RUN ------------------------------

Enter first string: a
Enter second string: B
strcmp("a","B") returned 1
MyStrcmp("a","B") returned 31

------------------------------- END OF 1ST RUN -------------------------------

------------------------------ START OF 2ND RUN ------------------------------

Enter first string: HE
Enter second string: HELLO
strcmp("HE","HELLO") returned -1
MyStrcmp("HE","HELLO") returned -76

------------------------------- END OF 2ND RUN -------------------------------

------------------------------ START OF 3RD RUN ------------------------------

Enter first string: HE
Enter second string: EHLLO
strcmp("HE","EHLLO") returned 1
MyStrcmp("HE","EHLLO") returned 3

------------------------------- END OF 3RD RUN -------------------------------

------------------------------ START OF 4TH RUN ------------------------------

Enter first string: &=#
Enter second string: ~!
strcmp("&=#","~!") returned -1
MyStrcmp("&=#","~!") returned -88

------------------------------- END OF 4TH RUN -------------------------------

------------------------------ START OF 5TH RUN ------------------------------

Enter first string:
Enter second string:
strcmp("","") returned 0
MyStrcmp("","") returned 0

------------------------------- END OF 5TH RUN -------------------------------

------------------------------ START OF 6TH RUN ------------------------------

Enter first string: @
Enter second string: @
strcmp("@","@") returned 0
MyStrcmp("@","@") returned 0
```

```
------------------------------- END OF 6TH RUN -------------------------------

------------------------------ START OF 7TH RUN ------------------------------

Enter first string: aaaaaaaaaaaaaaa
Enter second string: z
strcmp("aaaaaaaaaaaaaaa","z") returned -1
MyStrcmp("aaaaaaaaaaaaaaa","z") returned -25

------------------------------- END OF 7TH RUN -------------------------------

------------------------------ START OF 8TH RUN ------------------------------

Enter first string: The cat
Enter second string: in the fiddle
strcmp("The cat","in the fiddle") returned -1
MyStrcmp("The cat","in the fiddle") returned -21

------------------------------- END OF 8TH RUN -------------------------------
```

```
THIS WAS SENT FROM A NOTIFICATION-ONLY ADDRESS THAT CANNOT ACCEPT INCOMING MAIL.
For help please contact the instructor at the email address provided on the
"Announcements" page of the course website.  The assignment checker DOES NOT GRADE your
submissions but merely reports on issues so you can correct them and resubmit, thereby
avoiding unnecessary credit loss.  ALL GRADING IS DONE MANUALLY BY THE INSTRUCTOR after
the assignment deadline based solely upon the NEWEST submission of each exercise.  BE
WARY of correcting minor issues after the deadline because a late deduction will
usually be much greater than a minor issue deduction.


   From: Shaun Chemplavil <mailto:shaun.chemplavil@gmail.com>
   Subject: C1A6E3_U08713628
   Submitted: 5/8/2020 8:36:39 AM PDT
   Course: C/C++ Programming I (Section 146359)
   Student's name: Shaun Chemplavil
   Contact email: shaun.chemplavil@gmail.com
   Student ID: U08713628
   Assignment 6, Exercise 3
   Exercise point value: 6
   Files submitted:
      C1A6E3_main.c
      C1A6E3_GetSubstring.c

"Compile-time" results:
   No "compile-time" issues;
"Run-time" results:
   Program ran - No errors detected during preliminary testing (SEE ATTACHMENT);
```

```c
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 05 / 08 / 2020
// C1A6E3_GetSubstring.c
// Win10
// Visual C++ 19.0
//
// This function extracts a string of characters via a user defined start index
//  and character count
//

#define NULL_CHAR '\0'

char *GetSubstring(const char source[], int start, int count, char result[])
{
   //retain original result address
   char *result_copy = result;

   // Increment source pointer until we hit start offset (or null character)
   while (start-- > 0 && (*source != '\0'))
   {
      source++;
   }

   // copy source array count indicies to result (or until null character)
   while ((count-- > 0) && (*source != '\0'))
   {
      *result++ = *source++;
   }

   // Append Null Character to result
   *result++ = NULL_CHAR;

   return(result_copy);
}
```

This makes no sense. Why would you define a macro for this character then not use it everywhere you needed that character?

```c
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C / C++ Programming I : Fundamental Programming Concepts
// 146359 Raymond L. Mitchell Jr.
// 05 / 08 / 2020
// C1A6E3_main.c
// Win10
// Visual C++ 19.0
//
// This program prompts the user to enter a string, then extracts characters
// identifed by the user via an index and character count
//

#include <stdio.h>
#include <string.h>

#define LENGTH  257

char *GetSubstring(const char source[], int start, int count, char result[]);

int main(void)
{
   char source[LENGTH], result[LENGTH];
   int start, count;

   // get the users strings
   printf("Enter a sequence of 0 or more arbitrary printable characters: ");
   fgets(source, LENGTH, stdin);

   printf("Enter a space-separated start index and character count: ");
   scanf("%d%d", &start, &count);

   // remove newline character from string
   source[strcspn(source, "\n")] = '\0';

   // display results
   printf(
      "\"%s\", %d, %d, extracts \"%s\"\n",
      source, start, count, GetSubstring(source, start, count, result));

   return 0;
}
```

```
********** C1 ASSIGNMENT 6 EXERCISE 3 AUTOMATIC PROGRAM RUN RESULTS **********

************* THE RESULTS BELOW HAVE BEEN PARTIALLY CHECKED AND **************
*************   NO ERRORS WERE FOUND.  HOWEVER, THIS DOES NOT   **************
*************  NECESSARILY MEAN THAT THERE ARE NO ERRORS.  THE  **************
*************  INSTRUCTOR WILL DO A MORE THOROUGH CHECK DURING  **************
*************                  MANUAL GRADING.                  **************


------------------------------ START OF 1ST RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: This is really fun
Enter a space-separated start index and character count: 2 25
"This is really fun", 2, 25, extracts "is is really fun"

------------------------------- END OF 1ST RUN -------------------------------

------------------------------ START OF 2ND RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: This is really fun
Enter a space-separated start index and character count: 27 9
"This is really fun", 27, 9, extracts ""

------------------------------- END OF 2ND RUN -------------------------------

------------------------------ START OF 3RD RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: This is really fun
Enter a space-separated start index and character count: 0 12
"This is really fun", 0, 12, extracts "This is real"

------------------------------- END OF 3RD RUN -------------------------------

------------------------------ START OF 4TH RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: one two three
Enter a space-separated start index and character count: 5 35
"one two three", 5, 35, extracts "wo three"

------------------------------- END OF 4TH RUN -------------------------------

------------------------------ START OF 5TH RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: one two three
Enter a space-separated start index and character count: 18 7
"one two three", 18, 7, extracts ""

------------------------------- END OF 5TH RUN -------------------------------

------------------------------ START OF 6TH RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: one two three
Enter a space-separated start index and character count: 6 5
"one two three", 6, 5, extracts "o thr"

------------------------------- END OF 6TH RUN -------------------------------

------------------------------ START OF 7TH RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters: one two three
Enter a space-separated start index and character count: 0 3
"one two three", 0, 3, extracts "one"

------------------------------- END OF 7TH RUN -------------------------------

------------------------------ START OF 8TH RUN ------------------------------

Enter a sequence of 0 or more arbitrary printable characters:
Enter a space-separated start index and character count: 3 18
"", 3, 18, extracts ""

------------------------------- END OF 8TH RUN -------------------------------
```