

**Recep Tayyip Erdogan University**

**Faculty of Engineering and Architecture**

**Computer Engineering**

CE103- Algorithms and Programming - I

**Homework-4 (Week-10)**

**Fall Semester, 2021-2022**

| Instructor | Asst. Prof. Dr. Uğur CORUH |
| --- | --- |
| Contact Information | [ugur.coruh@erdogan.edu.tr](mailto:ugur.coruh@erdogan.edu.tr) |
| Google Classroom Code | **3ipdtws** |
| Publish Date | **15.12.2021** |
| Due Date | **24.12.2021 23:59** |

**Complete the following homework requirements, prepare them in the format given in the description below until the deadline and time, and upload them to the classroom's related assignment.**

**Grades:**

| Problem-1 | 100 points |
| --- | --- |
| **Total** | **100** points |

**Problem-1 (100 points):**

In this problem, you will develop C DLL and functions. Also, this C DLL functions equivalent in Csharp DLL, and you will test with a unit test. Also, there will be a middle layer as Cpp DLL.

For basic example;

you will develop a basic function in C DLL named

char\* sayHelloToC(char\* name, int count);

This function will concatenate the string with a newline character and return it as an output.

This function will be called by a C++ DLL named

char\* sayHelloToCpp(char\* name, int count);

In Addition, this function will be called by a C# DLL named

String sayHelloToCs(string name,int count); also there will be

And finally, this function will be called via a unit test in Csharp and sayHelloToC equivalent function developed in csharp sayHelloToCsLocal(string name,int count);

Assert.Equal(“Computer\n Computer\n Computer\n”, sayHelloToCs(“Computer”,3));

Assert.Equal(sayHelloToCsLocal (“Computer”,3), sayHelloToCs(“Computer”,3));

You will make this operation for hw2 functions as below; this function will be developed in C DLL, and their equivalent will also be developed with Csharp, and there will be a middle layer as Cpp DLL.

/\*\*

\*

\*   @name   fibonacciNumber (ce103\_fibonacciNumber)

\*

\*   @brief Fibonacci Number Calculator

\*

\*   Calculates the Fibonacci number in the given index and return as output

\*

\*   @param  [in] fiIndex [\b int]  index of fibonacci number in the serie

\*

\*   @retval [\b int] calculated fibonacci number

\*\*/

int ce103\_fibonacciNumber(int fiIndex);

/\*\*

@name   strrev (ce103\_strrev)

@brief \b Reverse String

Reverse given string

@param [in] fiStr [\b char\*] The given string needed to be reversed.

@retval [\b char\*] This function returns the string after reversing the given string

\*\*/

char\* ce103\_strrev(char\* fiStr);

/\*\*

@name   strlen (ce103\_strlen)

@brief \b Get string length

Returns the length of the C string str.

The length of a C string is determined by the terminating null-character:

A C string is as long as the number of characters between the beginning of

the string and the terminating null character

(without including the terminating null character itself).

see more <a href="https://en.cppreference.com/w/c/string/byte/strlen">strlen reference 1</a>

see more <a href="https://www.programiz.com/c-programming/library-function/string.h/strlen">strlen reference 2</a>

see more <a href="https://www.cplusplus.com/reference/cstring/strlen/">strlen reference 3</a>

@param [in] fiStr [\b const char\*] pointer to the null-terminated byte string to be examined

@retval [\b int] The length of the null-terminated byte string str.

\*\*/

int ce103\_strlen(const char\* fiStr);

/\*\*

@name   strcat (ce103\_strcat)

@brief \b Concatenate strings

Appends a copy of the null-terminated byte string pointed to by src to the end of the null-terminated byte string pointed to by dest

The character src[0] replaces the null terminator at the end of dest. The resulting byte string is null-terminated.

The behavior is undefined if the destination array is not large enough for the contents of

both src and dest and the terminating null character. The behavior is undefined if the strings overlap.

The behavior is undefined if either dest or src is not a pointer to a null-terminated byte string.

see more <a href="https://en.cppreference.com/w/c/string/byte/strcat">strcat reference</a>

see more <a href="https://www.cplusplus.com/reference/cstring/strcat/">strcat reference</a>

@param  [in] fiDest [\b char\*] pointer to the null-terminated byte string to append to

@param  [in] fiSrc  [\b char\*] pointer to the null-terminated byte string to copy from

@retval [\b char\*] returns a copy of dest

\*\*/

char\* ce103\_strcat(char\* fiDest, char\* fiSrc);

/\*\*

@name   strcmp (ce103\_strcmp)

@brief  \b Compare two strings

Compares two null-terminated byte strings lexicographically.

The sign of the result is the sign of the difference between

the values of the first pair of characters (both interpreted as unsigned char)

that differ in the strings being compared.The behavior is undefined

if lhs or rhs are not pointers to null-terminated byte strings.

see more <a href="https://en.cppreference.com/w/c/string/byte/strcmp">strcmp reference</a>

see more <a href="https://www.cplusplus.com/reference/cstring/strcmp/">strcmp reference</a>

@param  [in] fiLhs [\b const char\*] pointers to the null-terminated byte strings to compare

@param  [in] fiRhs [\b const char\*] pointers to the null-terminated byte strings to compare

@retval [\b int] Negative value if lhs appears before rhs in lexicographical order.

Zero if lhs and rhs compare equal.

A positive value if lhs appears after rhs in lexicographical order.

\*\*/

int ce103\_strcmp(const char\* fiLhs, const char\* fiRhs);

/\*\*

\*

@name  strcpy (ce103\_strcpy)

@brief \b Copy string

Copies the C string pointed by source into the array pointed by destination,

including the terminating null character (and stopping at that point).

To avoid overflows, the size of the array pointed by destination shall be long enough to contain

the same C string as source (including the terminating null character),

and should not overlap in memory with the source.

see more <a href="https://en.cppreference.com/w/c/string/byte/strcpy">strcpy reference 1</a>

see more <a href="https://www.cplusplus.com/reference/cstring/strcpy/">strcpy reference 2</a>

@param [out] foDestination [\b char\*] Pointer to the destination array where the content is to be copied.

@param [in]  fiSource [\b const char\*] C string to be copied.

@retval returns a copy of dest

\*\*/

char\* ce103\_strcpy(char\* foDestination, const char\* fiSource);

/\*\*

\* @name    hex2bin (ce103\_hex2bin)

\* @brief   \b Hexadecimal to Binary (BCD)  Conversion

\*

\* Hexadecimal to Binary (BCD)  Conversion

\* Packs hexadecimal string to packed binary array, Example: "AB1234" => 0xAB 0x12 0x34

\* If length of the input string is less than the fiHexLen,remaining bytes is not filled.

\* If odd number of characters processed, last nibble is padded with 0

\*

\* @param   [in]  fiHex    [\b unsigned char\*] Ascii hex string.

\* @param   [in]  fiHexLen [\b int]     Ascii data length.

\* @param  [out]  foBin    [\b char\*]   Convertion result as binary.

\*/

void ce103\_hex2bin(char\* fiHex, int fiHexLen, unsigned char\* foBin);

/\*\*

\* @name    bin2hex (ce103\_bin2hex)

\* @brief   \b Binary (BCD) to Hexadecimal Conversion

\*

\* Unpacks alpha numeric value, Example: 0x12 0x34 = "1234".

\*

\* @param [in]  fiBin      [\b unsigned char\*]    Binary data to be converted.

\* @param [in]  fiBinLen   [\b int] Binary data length.

\* @param [out] foHex      [\b char\*] Convertion result as ascii. Doubles the binary length.

\*

\*/

void ce103\_bin2hex(unsigned char\* fiBin, int fiBinLen, char\* foHex);

Problem restrictions

* There will be a single solution, and under this, you will develop all of your projects
* Solution file name (ce103-hw4-name-surname)
* C DLL project name (ce103-hw4-c-dll)
* Cpp DLL project name (ce103-hw4-cpp-dll)
* Csharp DLL project name (ce103-hw4-cs-dll)
* Csharp Unit Test project name (ce103-hw4-test)
* There should be a well-written report. (ce103-hw4-name-surname.docx)
* Github repo name should be ce103-hw4-name-surname
* Provide Doxygen generated PDF file only as ce103-hw4-name-surname-dox.pdf

***Problem Grading Criteria***

1. *You need to share your code in Github with a private repository that we can see only.*
2. *You need to share your code in Classroom without binaries, delete unnecessary files (binaries will give you penalty points)*
3. *Explain what you did in your homework with comments and explanations; please do not send it if you didn’t understand.*
4. *Beautify your code with Indentation; your codes should look good.*
5. *Project and File Types should be correct*
6. *Do not send not working code or project with bugs (Not Running or Complaining)*
7. *Your project test results should be correct*
8. *If there is an algorithm solution, then explain your methods one-by-one*
9. *Correct naming for source code ce103-hw4-name-surname.rar (this will include visual studio solution that you cloned from GitHub)*
10. Do not copy your friend’s code; copy detect software for similarity; make it own. There will be plagiarism checking. You have already created these source codes. Now you will make them your source codes.
11. You should only share with us Doxygen-generated pdf files. Do not send HTML or another part of auto-generated documentation files.

Good Luck!