Recep Tayyip Erdogan University

Faculty of Engineering and Architecture

Computer Engineering

CE103- Algorithms and Programming - I

Homework-2 (Week-5)

Fall Semester, 2021-2022

Instructor	**Asst. Prof. Dr. Uğur CORUH**
Contact Information	ugur.coruh@erdogan.edu.tr
Google Classroom Code	**3ipdtws**
Publish Date	**03.11.2021**
Due Date	**13.11.2021 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

Development Environment

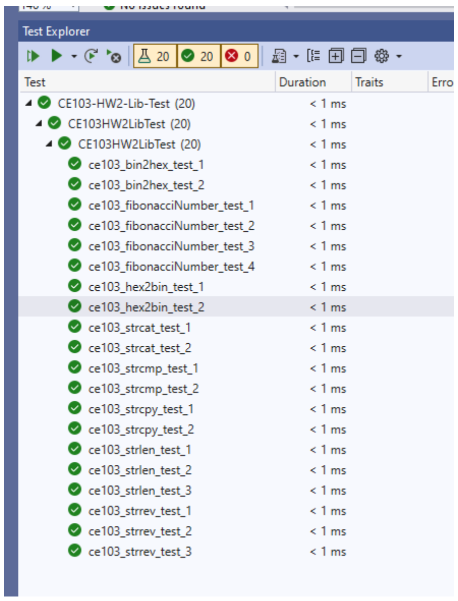
In this problem, you will use the TDD approach. Test-Driven Development. You will have a visual studio solution that has a unit test with empty function bodies. You will implement the correct answer for unit tests.

Download template from

https://github.com/ucoruh/CE103-HW2-No-Code

You will use a template that I shared with you. Do not modify unit tests and function signatures.

You can check that if the unit test is green, then your algorithm can be correct. But not %100 percent.



Test Explorer

20 ✓ 20 ✗ 0

Test	Duration	Traits	Erro
✓ CE103-HW2-Lib-Test (20)	< 1 ms		
✓ CE103HW2LibTest (20)	< 1 ms		
✓ CE103HW2LibTest (20)	< 1 ms		
✓ ce103_bin2hex_test_1	< 1 ms		
✓ ce103_bin2hex_test_2	< 1 ms		
✓ ce103_fibonacciNumber_test_1	< 1 ms		
✓ ce103_fibonacciNumber_test_2	< 1 ms		
✓ ce103_fibonacciNumber_test_3	< 1 ms		
✓ ce103_fibonacciNumber_test_4	< 1 ms		
✓ ce103_hex2bin_test_1	< 1 ms		
✓ ce103_hex2bin_test_2	< 1 ms		
✓ ce103_strcat_test_1	< 1 ms		
✓ ce103_strcat_test_2	< 1 ms		
✓ ce103_strcmp_test_1	< 1 ms		
✓ ce103_strcmp_test_2	< 1 ms		
✓ ce103_strcpy_test_1	< 1 ms		
✓ ce103_strcpy_test_2	< 1 ms		
✓ ce103_strlen_test_1	< 1 ms		
✓ ce103_strlen_test_2	< 1 ms		
✓ ce103_strlen_test_3	< 1 ms		
✓ ce103_strrev_test_1	< 1 ms		
✓ ce103_strrev_test_2	< 1 ms		
✓ ce103_strrev_test_3	< 1 ms		

You will use visual studio community edition 2022 or 2019. The solution is 2022 upgraded.

You will use C for library development, but for testing, there will be C++

You will create Doxygen documentation for your library. There are sample comments already written with rich examples.

Grading Criteria

1. *Github Code Sharing*
2. *Classroom Code and Report Sharing*
3. *Comments and Explanations*
4. *Code Indentation*
5. *Project and File Types*
6. *Code or Project Bugs (Not Running or Complaining)*
7. *Tests and Results*
8. *Algorithm Solution Methods and Explanations*

Homework Outputs

1. *Source Code (.rar with git tracked)*
2. *Source Code (github pushed link)*
3. *Homework Word Template*
4. *Doxygen Output (.rar)*

Problem-1 (100 points):

In this problem, you will develop the required functions listed below. All function descriptions are written, and parameters are defined. Develop functions and do not share any code that you didn't understand. Explain your source code with inline comments.

In the solution file, you will find the following projects

Solution Name

CE103-HW2

Project names

CE103-HW2-Test-App

(for using library functions not required, but if you need you can try functions with this console application)

CE103-HW2-Lib-Test

(include unit test)

CE103-HW2-Lib

(include library functions) -> You will write functions here…

You will find descriptions in the

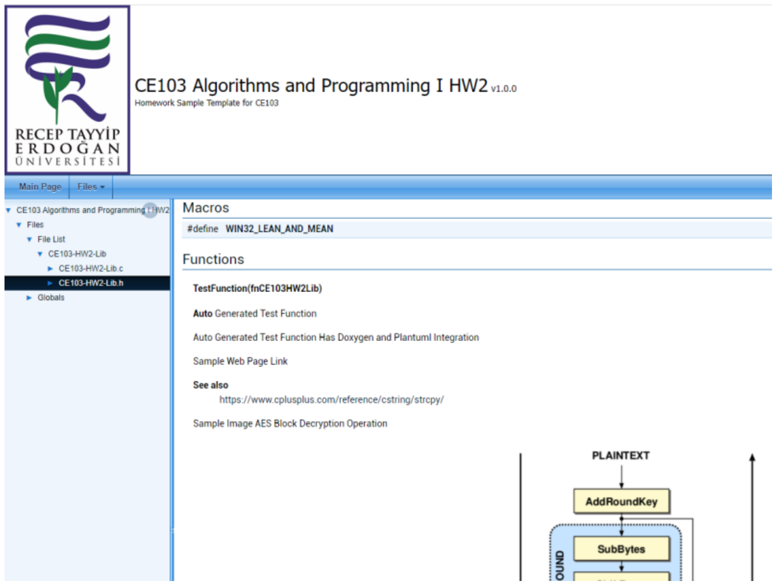
CE103-HW2-Lib.h

```
/**
	*
	*	@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 which is 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.
				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 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);
```

Doxygen usage

In the solution folder, you will find the doxygen folder. In this folder, there is a bat file to generate Doxygen documentation. When you complete installation, you can generate following documentation pages.



CE103 Algorithms and Programming I HW2 v1.0.0
Homework Sample Template for CE103

Macros

#define WIN32_LEAN_AND_MEAN

Functions

TestFunction(fnCE103HW2Lib)

Auto Generated Test Function

Auto Generated Test Function Has Doxygen and Plantuml Integration

Sample Web Page Link

See also
https://www.cplusplus.com/reference/cstring/strcpy/

Sample Image AES Block Decryption Operation

Good Luck!