CMPT125, Fall 2020

Homework Assignment 1

Due date: Wednesday, October 7, 2020, 23:59

You need to implement the functions in ***assignment1.c****.*

Submit only the ***assignment1.c*** file to CourSys.

Solve all 5 problems in the assignment.

The assignment will be graded automatically.
Make sure that your code compiles without warnings/errors, and returns the required output.

Your code MUST compile in CSIL with the Makefile provided.
If the code does not compile in CSIL the grade on the assignment is 0 (zero).
Even if you can’t solve a problem, make sure it compiles

Warning during compilation will reduce points.
More importantly, they indicate that something is probably wrong with the code.

Your code must be readable, and have reasonable documentation, but not too much.
No need to explain i+=2 with // increase i by 2.

An example of a test file is included.
Your code will be tested using the provided tests as well as additional tests.
You are strongly encouraged to write more tests to check your solution is correct, but you don’t have to submit them.

You need to implement the functions in ***assignment1.c****.*

If necessary, you may add helper functions to the assignment1.c file,

but you cannot add add main() to

Submit only the ***assignment1.c*** file to CourSys.

Question 1 [10 points].
Write a function that gets 2 positive ints each between 1 and 999, and outputs the number obtained by their concatenation.

```
int concat_ints(int n1, int n2)
```

For example:
`concat_ints(1, 25)` *should return* `125`.
`concat_ints(123, 456)` *should return* `123456`.
`concat_ints(999, 9)` *should return* `9999`.

Question 2 [10 points].
Implement the function that gets a char and converts it to lower case.

```
void upper2lowercase(char* c)
```

If `c*` *is an uppercase letter, the function changes it to the corresponding lowercase. Otherwise, makes no changes.*

For example:
*if **`c==’D’`, *the function should change it to* `’d’`.
*if **`c==’f’`, *the function should keep it as* `’f’`.
*if **`c==’5’`, *the function should keep it as* `’5’`.
*if **`c==’+’`, *the function should keep it as* `’+’`.

[Hint: you can check if the input is an upper case letter by using its numerical value.
The numerical values of the uppercase letters are consecutive.
For example, we have e.g. ‘A’ + 2 == ‘C’ and ‘D’ < ‘E’]

[Fun fact: a similar function `tolower(char c)` *is implemented in the library <ctype.h>]*
https://www.programiz.com/c-programming/library-function/ctype.h/tolower

Question 3 [25 points].
Write a function that gets a string and a non-negative integer n, and performs left rotation on the string by n symbols.

```
void left_rotate(char* str, unsigned int n)
```

Examples,
if `str` *contains the string* `”abcdef”` *and* `n=2`, *then after the call* `str` *will contain* `”cdefab”`.
if `str` *contains the string* `”abcdeefg123”` *and* `n=8`, *then after the call* `str` *will contain* `”123abcdeefg”`.
if `str` *contains the string* `”abcd”` *and* `n=0`, *then after the call* `str` *will contain* `”abcd”`.
if `str` *contains the string* `”abc”` *and* `n=6`, *then after the call* `str` *will contain* `”abc”`.

Question 4 [25 points].

Write a function that gets a 2-d array of given dimensions of ints.
It returns `true` *if the array contains two columns with exactly the same values in the same order, and returns* `false` *otherwise.*

```
bool contains_equal_columns(int height, int width, const int
ar[height][width])
```

Examples:
On input

```
{{1,2,3,2},
 {2,3,4,3},
 {1,2,3,2}}
```

it returns `true`*. (the column (2,3,2) appears twice)*
On input

```
{{1,2,3,4,5,6},
 {2,3,4,5,1,2},
 {2,3,4,5,0,4},
 {3,4,5,6,1,1}}
```

it returns `false`*.*
On input

```
{{1,1,1,2,4},
 {2,2,2,2,1},
 {3,2,2,2,1},
 {4,4,4,6,3}}
```

it returns `true` *(the column (1,2,2,4) appears twice)*

Question 5 [30 points].

Write a function that gets two strings containing positive integers, and outputs a new string containing their sum.

```
char* str_compute_sum(const char* num1, const char* num2)
```

For example, `str_compute_sum(“123456789”,”987654321”)` *returns* `“1111111110”`*.*

1. You may assume that the input is always legal, i.e., both strings are positive numbers
2. Note that the numbers may be larger than the maximum of `int` *or* `long`*.*
3. Also make sure that the returned string is created on the heap (and not as a local variable).