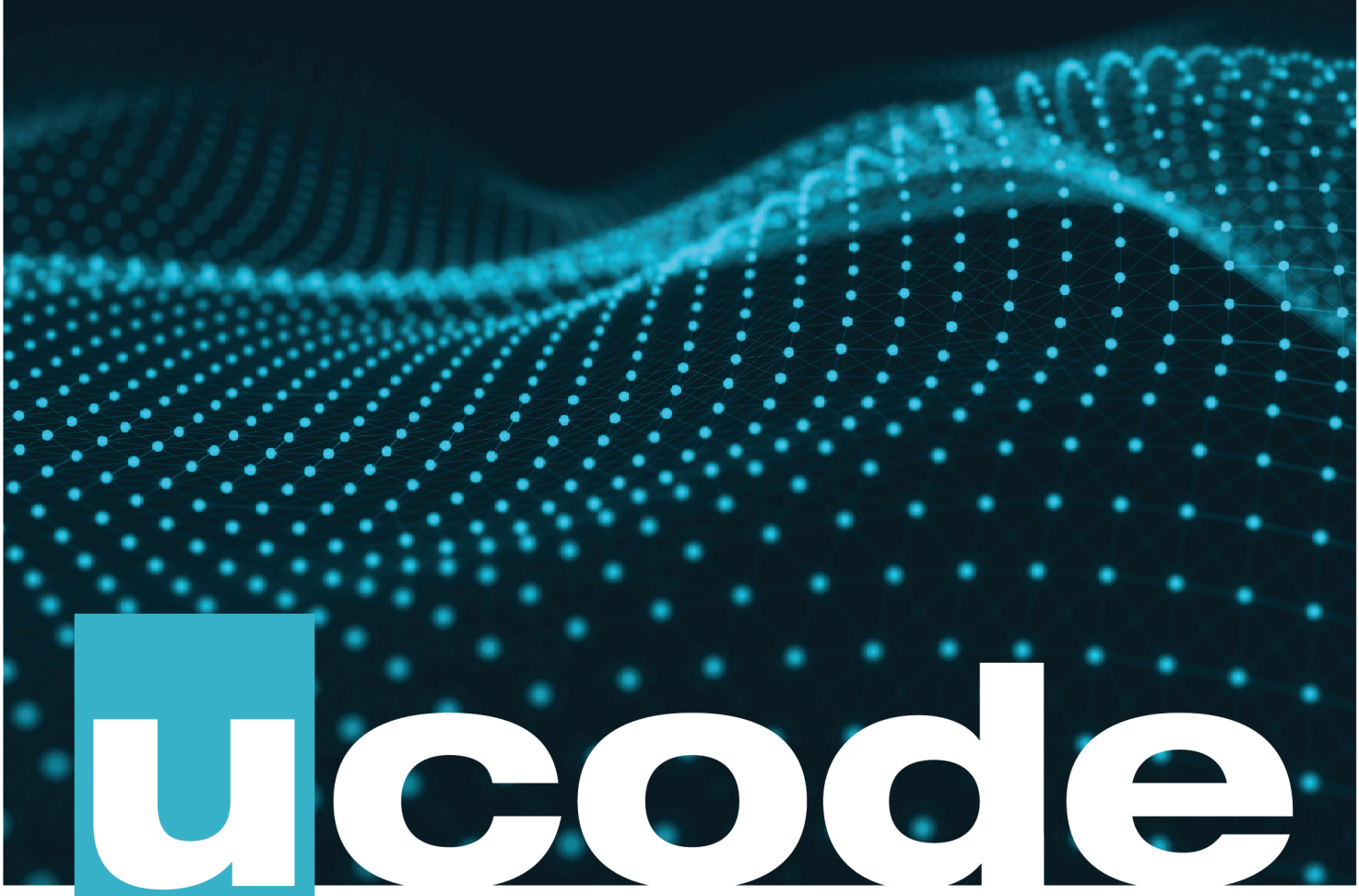
Marathon C

Sprint 07

June 7, 2019



ucode

Contents

ucode

Challenge Based Learning

1. Be attentive to all statements of the story. Examine the given examples carefully. They may contain details that are not mentioned in the task.
2. Perform only those tasks that are given in the story.
3. You should submit only the specified files in the required directory and nothing else. In case you are allowed to submit any files to complete the task you should submit only useful files. Garbage shall not pass.
4. You should compile C files with clang compiler and use these flags: `-std=c11 -Wall -Wextra -Werror -Wpedantic`.
5. Your program must manage memory allocations correctly. Memory which is no longer needed must be released otherwise the task is considered as incomplete.
6. You should use only functions which allowed in a certain task.
7. Usage of forbidden functions is considered as cheat and your challenge will be failed.
8. You must complete tasks according to the rules specified in `the Auditor`.
9. Your exercises will be checked and graded by students. The same as you. `Peer-to-Peer (P2P) learning`.
10. Also, your exercises will pass automatic evaluation which is called `Oracle`.
11. Got a question or you do not understand something? Ask the students or just Google that.
12. Use your brain and follow the white rabbit to prove that you are the Chosen one!!!

Task 00

NAME

New string

DIRECTORY

`t00/`

SUBMIT

`mx_strnew.c`

ALLOWED FUNCTIONS

`malloc`

DESCRIPTION

Create a function that:

- allocates memory for a string of a specific `size` and one additional byte for terminating `'\0'`;
- initializes each character with `'\0'`.

Your program must manage memory allocations correctly. Memory which is no longer needed must be released otherwise the task is considered as incomplete.

RETURN VALUES

- string of a specific size and terminated by `'\0'`;
- `NULL` if the creation fails.

SYNOPSIS

```
char *mx_strnew(const int size);
```

EXAMPLE

```
mx_strnew(10); //returns string with size 10 and terminated by '\0'
mx_strnew(-1); //returns NULL
```

SEE ALSO

`man 3 malloc`

Task 01

NAME

Duplicate string

DIRECTORY

`t01/`

SUBMIT

`mx_strdup.c, mx_strnew.c, mx_strlen.c, mx_strcpy.c`

ALLOWED FUNCTIONS

`malloc`

DESCRIPTION

Create a function which has the same behaviour as standard libc function `strdup`.

SYNOPSIS

```
char *mx_strdup(const char *str);
```

SEE ALSO

`man 3 strdup`

Task 02

NAME

Join strings

DIRECTORY

`t02/`

SUBMIT

`mx_strjoin.c, mx_strnew.c, mx_strlen.c, mx_strdup.c, mx_strcpy.c, mx_strcat.c`

ALLOWED FUNCTIONS

`malloc`

DESCRIPTION

Create a function that:

- concatenates strings `s1` and `s2` into new string;
- terminates new string with `'\0'`.

RETURN VALUES

- string as result of concatenation of `s1` and `s2`;
- new copy of `non-NULL` parameter if one and only one of the parameters is `NULL`;
- `NULL` if the join fails.

SYNOPSIS

```
char *mx_strjoin(char const *s1, char const *s2);
```

EXAMPLE

```
str1 = "this";
str2 = "dodge ";
str3 = NULL;
mx_strjoin(str2, str1); //returns "dodge this"
mx_strjoin(str1, str3); //returns "this"
mx_strjoin(str3, str3); //returns NULL
```

Task 03

NAME

Copy array of integers

DIRECTORY

`t03/`

SUBMIT

`mx_copy_int_arr.c`

ALLOWED FUNCTIONS

`malloc`

DESCRIPTION

Create a function that copies array of integers to a new array.

RETURN VALUES

- `pointer` to the first element;
- `NULL` if array `src` does not exist or copy fails.

SYNOPSIS

```
int *mx_copy_int_arr(const int *src, int size);
```

EXAMPLE

```
arr1 = {1, 2, 3};
arr2 = NULL;
mx_copy_int_arr(arr1, 3); //returns array [1, 2, 3]
mx_copy_int_arr(arr2, 3); //returns NULL
```

Task 04

NAME

Delete string

DIRECTORY

`t04/`

SUBMIT

`mx_strdel.c`

ALLOWED FUNCTIONS

`free`

DESCRIPTION

Create a function that:

- takes a pointer to string;
- frees the string memory with `free`;
- sets string to `NULL`.

SYNOPSIS

```
void mx_strdel(char **str);
```

SEE ALSO

`man 3 malloc`

Task 05

NAME

Concatenate words

DIRECTORY

`t05/`

SUBMIT

`mx_concat_words.c, mx_strdel.c, mx_strjoin.c, mx_strnew.c, mx_strlen.c, mx_strdup.c, mx_strcpy.c, mx_strcat.c`

ALLOWED FUNCTIONS

`malloc, free`

DESCRIPTION

Create a function that:

- concatenate the `NULL` terminated array of words into sentence, where words are separated by a single space character;
- frees all unused memory.

RETURN VALUES

- result of concatenation of the `NULL` terminated array into string;
- `NULL` if array of strings `words` does not exist or concatenation fails.

SYNOPSIS

```
char *mx_concat_words(char **words);
```

EXAMPLE

```
words = {"Free", "your", "mind.", NULL};
mx_concat_words(words); //returns "Free your mind."
mx_concat_words(NULL); //returns NULL
```

FOLLOW THE WHITE RABBIT

`http://lmgtfy.com/?q=memory+leaks`

Task 06

NAME

Trim string

DIRECTORY

`t06/`

SUBMIT

`mx_strtrim.c, mx_strdel.c, mx_isspace.c, mx_strnew.c, mx_strlen.c, mx_strncpy.c`

ALLOWED FUNCTIONS

`malloc, free`

DESCRIPTION

Create a function that:

- creates new string without whitespace characters at the beginning and the end of the string;
- frees all unused memory.

RETURN VALUES

- new trim string;
- `NULL` if string `str` does not exist or trim of string fails.

SYNOPSIS

```
char *mx_strtrim(const char *str);
```

EXAMPLE

```
name = "\f  My name... is Neo  \t\n ";
mx_strtrim(name); //returns "My name... is Neo"
```

Task 07

NAME

Clean string

DIRECTORY

`t07/`

SUBMIT

`mx_del_extra_whitespaces.c, mx_strtrim.c, mx_isspace.c, mx_strncpy.c, mx_strnew.c, mx_strdel.c, mx_strlen.c`

ALLOWED FUNCTIONS

`malloc, free`

DESCRIPTION

Create a function that:

- creates new string without whitespace characters in the beginning and at the end of a string;
- puts in the new string exactly one space character between words;
- frees all unused memory.

Word is a sequence of characters separated by whitespaces.

RETURN VALUES

- new created string;
- `NULL` if string `str` does not exist or creation of string fails.

SYNOPSIS

```
char *mx_del_extra_whitespaces(const char *str);
```

EXAMPLE

```
name = "\f  My name...     is  \r Neo  \t\n ";
mx_del_extra_whitespaces(name); //returns "My name... is Neo"
```

Task 08

NAME

Split string

DIRECTORY

`t08/`

SUBMIT

`mx_strsplit.c, mx_strnew.c, mx_strncpy.c, mx_strdel.c, mx_count_words.c`

ALLOWED FUNCTIONS

`malloc, free`

DESCRIPTION

Create a function that:

- converts a string `s` to the `NULL`-terminated array of words;
- frees all unused memory.

Word is a sequence of characters separated by the character `c` as a delimiter.

RETURN VALUES

- `NULL`-terminated array of strings;
- `NULL` if the string `s` does not exist or conversion fails.

SYNOPSIS

```
char **mx_strsplit(char const *s, char c);
```

EXAMPLE

```
s = "**Good bye,**Mr.*Anderson.****";
arr = strsplit(s, '*');  // arr = ["Good bye,", "Mr.", "Anderson."]
s = "     Knock, knock,    Neo.    ";
arr = strsplit(s, ' ');  // arr = ["Knock,", "knock,", "Neo."]
```

Task 09

NAME

Delete duplicates

DIRECTORY

`t09/`

SUBMIT

`mx_del_dup_arr.c, mx_copy_int_arr.c`

ALLOWED FUNCTIONS

`malloc`

DESCRIPTION

Create a function that:

- takes an array of integers `src`, its size `src_size` and the pointer to the size of new array `dst_size`;
- initializes `dst_size` by the size of the array without duplicates.
- creates new array without duplicates;

RETURN VALUES

- new array without duplicates;
- `NULL` if array `src` does not exist or creation fails.

SYNOPSIS

```
int *mx_del_dup_arr(int *src, int src_size, int *dst_size);
```

Task 10

NAME

Delete array of strings

DIRECTORY

`t10/`

SUBMIT

`mx_del_strarr.c, mx_strdel.c`

ALLOWED FUNCTIONS

`free`

DESCRIPTION

Create a function that:

- takes a pointer to a `NULL`-terminated array of strings;
- deletes the content of the array;
- frees array memory with `free`.
- sets pointer to `NULL`.

SYNOPSIS

```
void mx_del_strarr(char ***arr);
```