Tartalomjegyzék

C string

Study this code a bit:

```
#include <stdio.h>

int main() {
    char str[] = "puppy";
    printf("The word is %s.\n", str);

    return 0;
}
```

Try to print the elements of the str array one by one as if they were integers (use %d in printf). The str array has 6 elements actually (even though it only stores 5 characters).

You can also read strings with scanf, but for this we'll need an appropriately large char array:

```
char str[100];
scanf("%s", str);
```

So strings are actually just character arrays with a "http://wiki.math.bme.huterminal null"http://wiki.math.bme.hu character at the end. The character code of this is '\0'. Example of a **for** cycle to go through a string:

```
for(i = 0; str[i] != '\0'; i++) {
    printf("%c", str[i]);
}
```

Exercises

Open a new project for each exercise or a new file if you're in the command line.

Don't use any string or other libraries in the coming exercises. It would defeat their purpose. Only use stdio.h.

Character count

Write a function that takes a C string and a character (char type). It returns how many times the given character appears in the string.

Most common character

Write a function that takes a C string and returns its most common character.

Centroid

Write a C program that read an arbitrary number of 3 dimensional points (but at most 10). It then calculates and prints the centroid of the points (the coordinates' average). The input stops if it receives the (0, 0, 0) point. (This (0, 0, 0) won't be part of the calculation.)

Character removal

Write a C function that takes 2 C strings. It removes all characters from the first string that appear in the second. As an example, with the input "http://wiki.math.bme.hupuppydog"http://wiki.math.bme.hu, "http://wiki.math.bme.hupog"http://wiki.math.bme.hu it changes the first string to "http://wiki.math.bme.huuyd"http://wiki.math.bme.hu.

Parenthesis

Write a C function that takes a C string and a positive integer **n**. It removes the paranthesis up to **n** depth. We can assume that there aren't parenthesis without a pair and that they don't intersect. Embedded parenthesis are allowed though. An example:

```
((There was) once) a (puppy)dog.
```

Using 1 depth removal here would result in:

```
(There was) once a puppydog.
```

2 depth:

```
There was once a puppydog.
```

Search

Write a C function that takes 2 C strings and checks whether the second can be found in the first. (It returns 0 if it can't be found and 1 if it can.)

If you've completed that, then modify it so that the **.** (dot) character is a wildcard. It should match everything. For example "http://wiki.math.bme.hupuppydog"http://wiki.math.bme.hu, "http://wiki.math.bme.hup.pp.d.g"http://wiki.math.bme.hu would return 1.