String and Character Manipulation

It is often a good exercise to implement your own versions of some standard library functions. In this task, you need to implement some standard library functions related to character and string manipulations.

The functions you need to implement are listed below.

```c
int hw4_islower(int ch);
int hw4_isupper(int ch);
int hw4_isalpha(int ch);
int hw4_isdigit(int ch);
int hw4_tolower(int ch);
int hw4_toupper(int ch);

size_t hw4_strlen(const char *str);
char *hw4_strchr(const char *str, int ch);
char *hw4_strcpy(char *dest, const char *src);
char *hw4_strcat(char *dest, const char *src);
int hw4_strcmp(const char *lhs, const char *rhs);
```

Your implementations should conform to the C standard. The standard documentations and examples on these functions can be found at https://en.cppreference.com/w/c/string/byte. **Do not trust materials from any other sources like Wikipedia, Baidu Zhidao, CSDN, Zhihu, etc.**

Note that

- All functions' return types, parameter declarations and behaviors should meet the requirements in the cppreference pages (accessible via the link above). The function names are prefixed `hw_` to avoid name collisions.
- Be careful: The cppreference website contains both C and C++ documentations. Refer to the C documentation pages instead of the C++ ones. C and C++ have slight differences and are not fully compatible.
- You may ignore the `restrict` qualifier on the pointer parameters of `strlen`, `strchr`, `strcpy`, `strcat` and `strcmp`.
- Due to backward compatibility issues, the C standard requires `strchr` to return `char *` instead of `const char *`. To disable the warning about "discarded qualifiers", just use an explicit cast like `char *pos = (char *)str;` or `return (char *)str;`.
- The behaviors of some functions are subject to locale settings, as is mentioned by the standard documentations. **You only need to work under the default locale (`"C"`).**
- It is not allowed to use any standard library functions. You must implement everything on your own.
- It is guaranteed that the tests do not lead to undefined behaviors.

Moreover, you need to make your implementation **as simplified as possible**. In details:

- For `hw4_islower`, `hw4_isupper`, `hw4_isalpha`, `hw4_isdigit`, `hw4_tolower` and `hw4_toupper`, the function body should contain **only one statement**, which is a `return` statement.
- For `hw4_strlen`, `hw4_strchr`, `hw4_strcpy`, `hw4_strcat` and `hw4_strcmp`, these functions should be no more complex than a single pass of the given strings. You can't scan the strings over and over again, or go back and forth between some positions. **For example, if your `hw4_strlen`**

traverses the whole string to count the number of characters, you can't call it in `strcmp` to simply obtain the length of the strings, because you still need to traverse the strings to do the comparison.
- You may define helper functions, but make sure they are necessary and meaningful.

Submission and grading

On the OJ, submit your code containing the definitions of the functions. Do not include the `main` function in your submission. The OJ will check whether your implementations are correct.

We will have an off-line check after the deadline. You will need to explain your implementations to your TAs. The grade for this problem is the combination of the results on OJ and that of the off-line check.

Violation of certain rules will not be detected on OJ. For example, you may get full score on OJ if you call the standard library versions of these functions directly, but this will result in zero points in the end.

Samples

No samples will be provided here, because the cppreference pages already have many.