COMP10002 Foundations of Algorithms

Workshop Week7

Wenbin Cao

September 12, 2019

GitHub Repo: https://github.com/AlanChaw/COMP10002-FoA

Outline

Strings and Arrays

Question Discussion

Assignment 1

Array of Strings

Example:

```
char *words[10] = {"Algorithms", "are", "fun"};
int i;
for (i = 0; words[i] != NULL; i++) {
    printf("%s\n", words[i]);
}
```

Output:

```
Algorithms
are
fun
```

Array of Strings

Example program of reading input strings into a 2D-array

Inner Functions for Strings

Function	Purpose
`int strlen(char *s)`	Returns the number of characters in `s`, not including the null byte.
`char *strcpy(char *dest, char *src)`	Copies `src`, including the null byte, into the array space indicated by `dest`. No bounds checking is done, and `dest` must be large enough to accommodate `src`, plus a null byte. This makes it risky. Be careful!
`char *strncpy(char *dest, char *src, int n)`	As for `strcpy`, except that `n` indicates the number of characters to be copied. Less risky than `strcpy`, but note that `dest` must be (at least) `n` bytes long, even if the string in `src` is shorter than `n`.
`int strcmp(char *s1, char *s2)`	Compares `s1` and `s2`, and returns negative, zero, or positive when `s1` is less than, equal to, or greater than, `s2`. The ASCII ordering shown on page 60 is used, with the uppercase letters preceding lowercase.
`int strncmp(char *s1, char *s2, int n)`	As for `strcmp`, except that `n` supplies a limit on the number of characters examined.
`int strcasecmp(char *s1, char *s2)`	As for `strcmp`, except that upper and lower case alphabetic characters are considered to be equal.
`char *strcat(char *dest, char *src)`	Appends a copy of `src` to the string `dest` by overwriting its null byte, copying characters, and then writing a new null byte. The string `dest` must be large enough to accommodate the extended string. Risky.
`char *strncat(char *dest, char *src, int n)`	As for `strcat`, except that `n` provides a limit on the number of characters of the second argument that may be copied. Safer than `strcat`.
`int atoi(char *s)`	Returns the integer value represented by the characters of `s`.
`double atof(char *s)`	Returns the `double` value represented by the characters of `s`.

Discussion

Exercise 1

Write a function `is_subsequence(char *s1, char *s2)` that returns `1` if the characters in `s1` appear within `s2` in the same order as they appear in `s1`. For example, `is_subsequence("bee", "abbreviate")` should be `1`, whereas `is_subsequence("bee", "acerbate")` should be `0`.

Discussion

Exercise 2

Ditto arguments, but determining whether every occurrence of a character in `s1` also appears in `s2`, and `0` otherwise. For example, `is_subset("bee", "rebel")` should be `1`, whereas `is_subset("bee", "brake")` should be `0`.

Discussion

Exercise 3

Write a function `is_anagram(char *s1, char *s2)` that returns `1` if the two strings contain the same letters, possibly in a different order, and `0` otherwise, ignoring whitespace characters, and ignoring case. For example, `is_anagram("Algorithms", "Glamor Hits")` should return `1`.

Assignment 1

- Assignment FAQ page
- Assignment Specification
- Marking Rubric
- Submission Instruction

Due on 10am Monday, 23 September