Lab #3

CS-2050

February 8, 2023

1 Requirements

In this lab, you will cover pointer arithmetic and the casting of pointers. Be careful to note which type of pointer you are working with, and remember that different types may have different sizes. Also remember that you can treat characters like integers (IE: you can print them with %d to see their integer value).

1.1 strAlloc

```
char * strAlloc(int size)
```

Info: This function takes an integer representing the length of a string, and allocates a character array with the given size. The size of the array should be stored before the start of the array as an **int**. It returns a pointer to the array on success, or NULL on failure.

1.2 strLen

```
int strLen(char *str)
```

Info: This function takes a character array that was allocated using *strAlloc*, and returns the size which is stored before the array.

1.3 strCpy

```
void strCpy(char *source, char *dest)
```

Info: This function takes a source array, and a destination array, both of which were allocated using *strAlloc*. It will copy the contents of the source array into the destination array. It will assume that the destination array is ***at least as large*** as the source array. For example:

```
// before
source = { 'H', 'E', 'L', 'L', 'O' };
dest =   { ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ' };
// after
dest =   { 'H', 'E', 'L', 'L', 'O', ' ', ' ', ' ' };
```

1.4 strCmp

```
int strCmp(char *source, char *dest)
```

Info: This function takes a source array, and a destination array, both of which were allocated using *strAlloc*. It will compare the contents of the source array with the destination array, and return **1** if they are the same string, or **0** if they are not. It will assume that the destination array is ***the same size*** as the source array. For example:

```
// returns 1
source = { 'H', 'E', 'L', 'L', 'O' };
dest =   { 'H', 'E', 'L', 'L', 'O' };
// returns 0
source = { 'H', 'E', 'L', 'L', 'O' };
dest =   { 'H', 'e', 'l', 'l', 'o' };
```

1.5 strFree

```
void strFree(char *str)
```

Info: This function takes a character array that was allocated using *strAlloc*, and frees the memory allocated to the array.

Grading: 17 points

1. Write required *strAlloc* function
   * 4 points
2. Write required *strLen* function
   * 2 points
3. Write required *strCpy* function
   * 3 points
4. Write required *strCmp* function
   * 6 points
5. Write required *strFree* function
   * 2 points

Notice:

1. All of your lab submissions **must** include documentation to receive full points.
2. All of your lab submissions must compile under GCC using the $-Wall$ and $-Werror$ flags to be considered for a grade.
3. You are expected to provide proper documentation in every lab submission, in the form of code comments. For an example of proper lab documentation and a clear description of our expectations, see the lab policy document.