CS 137: Assignment #9

Due on Friday, Dec 2, 2022, at 11:59 PM

Submit all programs using the Marmoset Submission and Testing Server located at
https://marmoset.student.cs.uwaterloo.ca/

Victoria Sakhnini

Fall 2022

Notes:

- Use the examples to guide your formatting for your output. Remember to terminate your output with a newline character.
- For this assignment, you may use any content covered until the end of Module 13.
- Use Valgrind for testing
- <math.h> is allowed only for problem 2.

Problem 1

Create a C program `stringfuns.c` that consists of the following two functions:

```
void reverseConcatenate (void *lhs, void *rhs);
void concatenate (void *lhs, void *rhs);
```

to be used with `reduce` function to concatenate all strings in an array (of length >=1) in reverse order and the original order, respectively.

You are to submit this file containing only your implemented function (that is, you must delete the test cases portion and the `reduce` function.). However, **you should keep the required included libraries.**

The sample code for testing is below.

```
1. #include <stdio.h>
2. #include <string.h>
3. #include <assert.h>
4.
5. //The function "reduce" as found in Chapter 13 of the course notes.
6. void reduce(void *src, size_t n, size_t src_bytes,
7.              void *dest, void (*f) (void *, void *))
8. {
9.     if (n == 1)
10.      {
11.          f(src, dest);
12.          return;
13.      }
14.      reduce((char *)src+src_bytes, n-1, src_bytes, dest, f);
15.      f(src, dest);
16.  }
17.
18.  void reverseConcatenate (void *lhs, void *rhs)
19.  {
20.
21.  }
22.
23.  void concatenate (void *lhs, void *rhs)
24.  {
25.
26.  }
27.
```

```
28.  int main(void)
29.  {
30.
31.      int n = 10;
32.      char *words[]  = {"The", "Quick", "Brown", "", "Fox", "Jumps",
    "Over", "The", "Lazy", "Dog"};
33.      char result[] = "TheQuickBrownFoxJumpsOverTheLazyDog";
34.      char backwards_result[] = "DogLazyTheOverJumpsFoxBrownQuickThe";
35.
36.      char answer[1000]; // you may assume that the length of the
    concatenated strings will not exceed 1000.
37.      answer[0] = '\0';
38.      reduce(words, n, sizeof(char *), answer, reverseConcatenate);
39.      assert(!strcmp(answer, backwards_result));
40.
41.      answer[0] = '\0';
42.      reduce(words, n, sizeof(char *), answer, concatenate);
43.      assert(!strcmp(answer, result));
44.
45.      return 0;
46.  }
```

Problem 2

Create a C program `normmapreduce.c` that consists of the following two functions:

```
void EuclideanNorm(void *src, void *dest);
void sum(void *src, void *dest);
```

to be used with the provided `reduce` and `map` functions to calculate the norm of vectors and to calculate the sum of all norms. Check the provided program below for an example.

Note: The Euclidean Norm of the vector (x,y,x) is $\sqrt{x^2 + y^2 + z^2}$

You are to submit this file containing only your implemented function (that is, you must delete the test cases portion and the `reduce` & `map` functions.). However, **you should keep the required included libraries, and structure definition.**

Sample code for testing is below.

```c
#include <stdio.h>
#include <math.h>
#include <assert.h>

void map(void *src, size_t n, size_t src_bytes,
            void *dest, size_t dest_bytes,
            void (*f) (void *, void *)) {
    if (n == 0)
        return;
    f(src, dest);
    // f writes to dest itself:
    map(src + src_bytes, --n, src_bytes, dest + dest_bytes, dest_bytes, f);
}

void reduce(void *src, size_t n, size_t src_bytes,
               void *dest, void (*f) (void *, void *)) {
    if (n == 1) {
        f(src, dest);
        return;
    }
    reduce((char *)src + src_bytes, n - 1, src_bytes, dest, f);
    f(src, dest);
}
```

```
24.
25. typedef struct Vector {
26.     double x, y, z;
27. } Vector;
28.
29. void EuclideanNorm(void *src, void *dest);
30.
31. void sum(void *src, void *dest) ;
32.
33. int main(void) {
34.     Vector a[2] = { {1, 0, 1}, {2, 0, 3} };
35.     double b[2] = { 0 };
36.     double res = 0;
37.     map(a, 2, sizeof(Vector), b, sizeof(double), EuclideanNorm);
38.     reduce(b, 2, sizeof(double), &res, sum);
39.     assert(fabs(res - 5.019765)<= 0.000001);
40.
41.     return 0;
42. }
```