CSE2312 (Fall 2022)
Homework #5

Notes:
With this homework, we continue writing assembly functions for the RPi 3b/3b+/4b.

All numbers are in base-10 unless otherwise noted.

If part of a problem is not solvable, explain why in the answer area.

The target date to complete this homework set is November 8, 2022.

This homework set will not be graded, but please solve all of the problems to prepare for the quizzes and exams.

1. Suppose that BUSINESS2 structure is defined as:

```
typedef struct _BUSINESS2
{
    uint32_t taxId;
    char name[27];
    char direction;
    char street[35];
    uint32_t addNo;
    char city[30];
    char state[3];
    uint32_t zip;
} BUSINESS2;
```

Show the relative offset of each field in the structure from the beginning of the structure for the unpacked (default alignment) case:

Show the relative offset of each field in the structure from the beginning of the structure for the packed case:

2. Write assembly functions that implement the following C functions:

a. int32_t sumS32(const int32_t x[], uint32_t count)
    // returns sum of the values in the array (x) containing count entries.

b. int32_t dotpS32(const int32_t x[], const int32_t y[], uint32_t count)
    // returns the dot product of the values in the arrays (x and y)
    containing count entries.

c. uint32_t countAboveLimit(const int32_t x[], int32_t limit, uint32_t count)
    // returns number of values in the array (x) containing count entries
    that are > limit

d. int32_t findCityAligned (const char city[], const BUSINESS2 business[], uint32_t count)
    // returns the index of the first entry in the array (business) containing
    count entries which matches the requested city. If the city is not
    found, return a value of -1. You can assume that C default alignment is
    used for this problem.

e. int32_t findCityPacked (const char city[], const BUSINESS2 business[], uint32_t count)
    // returns the index of the first entry in the array (business) containing
    count entries which matches the requested city. If the city is not
    found, return a value of -1. You can assume that C packing is used for
    this problem.

3. Encode the following numbers as single-precision floating point numbers:

a. 0.125

      s = ___, e = _____, m = _____

      32b hex value = _____

b. -15.5625

      s = ___, e = _____, m = _____

      32b hex value = _____

c. 0.33…

      s = ___, e = _____, m = _____

      32b hex value = _____

d. 251.2578125

      s = ___, e = _____, m = _____

      32b hex value = _____

e. 2048.0000001

      s = ___, e = _____, m = _____

      32b hex value = _____

4. Assume float x = 1048576.

a. Calculate the smallest positive number that can be added to x that will not be lost in the mantissa.

b. In general, what is the ratio of the large to the smallest single-precision floating point number that can be added together without a loss of accuracy? Consider the cases you add a number to x = 1048576 or x = 2097151.