CSE2312-002, 003 (Spring 2019)
Homework #3

Notes:

- All numbers are in base-10 unless otherwise noted.
- If part of a problem is not solvable, explain why in the answer area.
- Print out the form and handwrite your answers in the spaces below.
- Place the hw3.s file and the scanned answers to problem 1 in a single zip file with name lastname_hw3.zip, where lastname is your last name as listed in MyMav.
- Submit the single zip file to Canvas before 11:59:00pm on April 11, 2020.
- Make sure that the code follows the procedure call standards for ARM architecture (see IHI0042F section 5.1), with emphasis on this requirement: “A subroutine must preserve the contents of the registers r4-r8, r10, r11 and SP (and r9 in PCS variants that designate r9 as v6).” (in other words, push and pop R4-11 if you need to use them, as shown in the vector.s examples in class)

1. Suppose that BUSINESS4 structure is defined as:

```
typedef struct _BUSINESS4
{
    uint32_t taxId;
    char name[22];
    char street[30];
    char direction;
    uint32_t addNo;
    char city[35];
    char state[3];
    uint32_t zip;
} BUSINESS3;
```

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. bool isStrEqual(const char* str1, const char* str2) // returns 1 the strings match, 0 otherwise

b. void strCopy(char* strTo, const char* strFrom) // copies strFrom to strTo
// note: strTo must have enough space to hold strFrom

c. int32_t sumS16(const int16_t x[], int32_t count)
// returns sum of the values in the array (x) containing count entries.

d. uint32_t sumU16(const uint16_t x[], uint32_t count)
// returns sum of the values in the array (x) containing count entries.

e. uint32_t countZeros(const int32_t x[], uint32_t count)
// returns number of zero values in the array (x) containing count entries

f. void rightStringFull(char strOut[], const char strIn[], uint32_t length)
// input: array (strIn) containing the input string, and the number of characters to extract (length)
// output: array (strOut) containing length number of strIn characters from the end of the array or an empty string if the length is too large
// strIn = ‘abcdef’, length = 5 → returns strOut = ‘bcdef’
// strIn = ‘abcdef’, length = 7 → returns strOut = ‘’

g. void rightStringTrunc(char strOut[], const char strIn[], uint32_t length)
// input: array (strIn) containing the input string, and the number of characters to extract (length)
// output: array (strOut) containing up to, but not exceeding length number of strIn characters from the end of the array
// strIn = ‘abcdef’, length = 5 → returns strOut = ‘bcdef’
// strIn = ‘abcdef’, length = 7 → returns strOut = ‘abcdef’

h. uint32_t countMatches(const char strIn[], const char strMatch[])
// input: array (strMatch) containing the string to match in the array (strIn)
// output: returns the number of occurences of strMatch in strIn

i. int32_t find2ndMatch(const char strIn[], const char strMatch[])
// input: array (strMatch) containing the string to find in the array (strIn)
// output: returns the offset within str of the 2nd occurrence of strMatch or -1 if not found

j. void sortAscendingInPlace (uint32_t x[], uint32_t count)
// input: array (x) containing count entries

// output: array (x), with the values sorted in ascending order

k. uint8_t decimalToUint8(const char str[])
   // convert the null-terminated string (str) to an unsigned 8-bit integer
   // treat the string as representing a decimal number

l. int8_t decimalToInt8(const char str[])
   // convert the null-terminated string (str) to a signed 8-bit integer
   // treat the string as representing a decimal number
   // if a character other than 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, or – is present, return 0

m. uint16_t hexStringToUint16(const char str[])
   // convert the null-terminated string (str) to an unsigned 16-bit integer
   // treat the string as representing a hexadecimal number
   // if a character other than 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, or F is present, return 0

n. void uint8ToBinaryString (char str[], uint8_t x)
   // convert the unsigned integer (x) to a null-terminated string (str) representing a binary number

o. int32_t findStreet (char street[], const BUSINESS3 business[], uint32_t count)
   // returns the index of the first entry in the array (business) containing count entries which matches the requested street. If the name is not found, return a value of -1.

Write the solution of each of these functions in a single file hw3.s with functions being callable from a C program. You do not need to send the .c file.