HW1

How to submit --

Send your login id on the department machine to lyang11@email.wm.edu with your name immediately. Otherwise, your homework will not be accepted.

Type the following to submit after you finish all the problems:

~lyang11/bin/submit cs304 hw1 square.c pointers.txt spaces.c square2.c words.c sum.c number.txt

0. Preparation:

Get a CS account: Get a CS account
Learn how to use lab machines if you would like to access the 121 machine remotely

Learn to use the following Linux commands:

>mkdir cs304
This will make a directory named cs304

>cd cs304
This will put you in the directory of cs304

Now you can use your editor (such as emacs) to edit your file, e.g.,
>emacs square.c &

>ls
This will show your files in the current directory

>ls -l
This will show your files in the current directory with more information about each file

>cd ..
This will let you get out of the current directory

Problem 1.

Write a C program to do the following:

Accept an input of integer n;
Print the square of 1, 2, 3, ..., n
Print the sum of the squares computed in the above line.

Save your program in square.c, compile it and run it:

>gcc -o square square.c
>./square

It will output like the following:

Please input a number: 5
The square of 1 is 1

The square of 2 is 4
The square of 3 is 9
The square of 4 is 16
The square of 5 is 25
The sum of the squares is 55

Problem 2. Pointers

Do not run the program, but read the program and figure out the output of the program:

```
#include <stdio.h>
int main() {

  int ar[10]={1,2,3,4,5,6,7,8,9,10}, *p1;
  char s1[10]="abcdefgh", s2[10]="1234567", *p2;

  p1 = ar;
  printf("ar is %p\n", ar);
  printf("The following addresses are %p %p %p\n", p1+15, &ar[14], p1-2);

  p2 = &s1[2];
  printf("s1 is %p\n", s1);
  printf("The following addresses are %p %p %p\n", p2+15, &s1[14], p2-2);

  *p2 = *p2 +1;
  *(s2+4) = 'z';
  printf("The following strings are %s %s %s\n", s1, s2, p2);

  return 0;
}
```

In the above printf statement, %p means "print a value of a pointer" in hexadecimal form; %s means "print a string with some starting address". Be careful that the numbers appeared in the program (15, 14, and -2) are in decimal form.

Suppose the following is the outcome of running the program. Fill in the blanks in the following. Save your answer in pointers.txt

---

ar is 0x7fff98d8a4f0
The following addresses are 0x_____ 0x_____ 0x_____
s1 is 0x7fff98d8a4e0
The following addresses are 0x_____ 0x_____ 0x_____
The following strings are _____

Problem 3.

Below is a program [text file] that reads a single line from the user and displays the number of characters in that line. For example, if the user enters the line "hello world" and then presses enter, the program displays the number 11 (five letters in each word, plus the space). The program uses function int getchar(void), which reads the next available character in the standard input.

```
#include <stdio.h>

int main() {
    int count;
    char c;

    printf("Input: ");
    count = 0;
    c = getchar();
    while (c != '\n') {
        count++;
        c = getchar();
    }
    printf("%d\n", count);
    return 0;
}
```

Here while (c != '\n') {...} means: while c is not equal to '\n', continue run the statement inside the {}.

Your job is to write following different programs based on this one, as described below.
Important: You may not use any built-in C functions (e.g. scanf) except for getchar() and printf().

spaces.c

In a file named spaces.c, write a program to count the number of spaces in the line typed by the user. Thus, for "hello world", the output should be 1, since there is just one space between the two words.

Problem 4: square2.c

In a file named square2.c, write a program that assumes the user types a number below 10000 and displays the square of that number. For example, if the user types 250 and the Enter key, the program should respond with 62500. In this problem, you can assume the number is non-negative.

The hardest part of this is not the squaring, but rather converting the sequence of digits typed by the user into a number (which you can then easily square). You might find the following pseudocode useful for your inspiration:

```
num ← 0
while there are more digits:
    num ← 10 · num + next digit
display num · num
```

You'll need to convert a digit character to its corresponding number to make this happen. You can legally treat a character as a number in an expression, as in "k = c;" where k is an int and c is a char. But its ASCII code will be used, so if c holds the digit '1', k will receive its ASCII value, which is 49. Fortunately, the ASCII code assigns the digits 0 through 9 successive values from 48 ('0') through 57 ('9'), and so you can perform this conversion by simply subtracting 48 from the character value.

Important: You may not use any built-in C functions (e.g. scanf) except for getchar() and printf().

Problem 5: `words.c`

In a file named `words.c`, write a program that counts the number of words in the line typed by the user, defined as the number of sequences of non-space characters — or, equivalently, the number of times a non-space character is preceded either by a space or by the beginning of the line. Here are some sample inputs and the corresponding outputs.

Important: You may not use any built-in C functions (e.g. scanf) except for getchar() and printf().

Input	Output
`"hello world"`	2
`"hello   world"`	2
`"4 words - right?"`	4
`"   word   word   "`	2

Problem 6: `sum.c`

In a file named `sum.c`, write a program that assumes the user types several numbers separated by single spaces and displays the sum of those numbers. For instance, given the input `"54 23 10"`, the output should be 87. You can assume that there are no more than 10 numbers, each of which is non-negative and less than 1,000,000.

Important: You may not use any built-in C functions (e.g. scanf) except for getchar() and printf().

Problem 7: Number representations: number.txt

For this problem let num1 = 0b11101101 and num2 = 0b10000000 be 8-bit numbers. (1) Suppose num1 and num2 are unsigned numbers (more precisely, unsigned char), what are they in decimal representation? (2) Suppose they are signed numbers (more precisely, signed char) in two's complement representation, what are they in decimal representation?

(3) Suppose an integer is represented in 4 bytes. Write the two's complement representation for number 157 and -157 in hexadecimal form..

(5) Convert `10001001` in binary to decimal (under the unsigned number representation) and hexadecimal.

(6) Convert `35` in decimal to binary and hexadecimal.

(7) Again we use 8-bit numbers. Calculate (-122)+(-111), 111+122, and 111-122 by using two's complement representations and addition in binary. Indicate whether overflow occurs.

Save your answer in number.txt