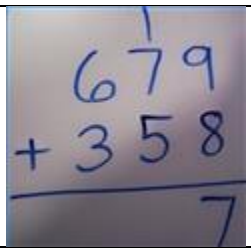

The University of New Mexico

CS 241—Data Organization

Lab 2: Sum of 4 digit numbers on line

The program given in section 1.6 of Kernighan and Ritchie indexes an array by converting a character digit to a numeric integer between 0 and 9.

The program given in 1.5.4 counts words in an input stream by using a flag to keep track of the state of the last character read has having been inside or outside of a word.

In this lab, you will combine these two techniques to write a program that, for each line of input, calculates and displays the sum of all four digit numbers in the line.

Each valid input line will consist of a stream of 4*n* digits with no separation.

The line of input:

```
0001022203330443\n
```

has a length of 16 and is valid with $n = 4$. Your program must identify the numbers 1, 222, 333, and 443, add these numbers and output their sum: 999.

Turning in your assignment

Attach your program file lineSum_*yourname*.c into the Lab 2 assignment in Blackboard Learn.

Suggested keywords and library functions

There exists many very different ways to getting the required functionality with a C program. In my solution, I used only keywords and library functions that are covered in chapter 1 of Kernighan and Ritchie.
The only include I use is **`<stdio.h>`**.
The only keywords: **`void`**, **`int`**, **`char`**, **`while`**, **`if`**, **`else`**, and **`return`**.
The only library functions: **`printf()`**, **`getchar()`**.
The only predefined constant: **`EOF`**.

That is all the blocks you need to build this widget. No need for arrays, pointers, malloc, free or any other more advanced topics.

It is not wrong to use other functions, when you are new to a language and searching for solutions on StackOverflow, it is easy to get lost in lost in the myriad of library functions and advanced features.

Sample Test Cases

Input Line	Output
3153\n	**3153**
00020005\n	**7**
012304560789\n	**1368**
1111222233334444555566667777888899990000\n	**49995**
\n	**0**
0001222233335555\n	**11111**
000122223333555\n	**error**
12121313141415151616171718181919222303 1\n	**17777**

Grading Rubric (total of 20 points)

[1 point]: The program starts with a comment stating the students first and last name and the program is named correctly.

[2 points]: Program compiles without errors or warnings on **moons.cs.unm.edu** using **/usr/bin/gcc** with no options.

[3 points]: Code follows the hallowed CS-241 Coding Standard - including comments and not repeating code.

[14 points]: Correct output for **lineSum.in**. This file includes 14 records. one point is earned for each line of correct output. This will be graded by a diff test with **lineSum.out**. The two text files are posted on the class website.

diff test:

//Copy **lineSum.in** and **lineSum.out** into your current working directory (**.**).
//WARNNING: it is important to copy the file as shown below. If you right-click in the
//browser and download, then **\n** characters might get replaced with **\n\r**, and/or
//other changes that will cause a correct program to fail.

```
% cp ~joel/public_html/cs241/data/lineSum.* .
```

//Compile your program **.**

```
% gcc lineSum.c
```

//Execute (run) your program with **lineSum.in** as input and output to **myOut.txt**.

```
% ./a.out < lineSum.in > myOut.txt
```

//compare your program's output to the expected output

```
% diff myOut.txt lineSum.out
```