

# Software Systems Lab: OutLab

**L<sup>A</sup>T<sub>E</sub>X**

Name: Shubham Hazra

Roll no: 210100143

August 30, 2022

# Lab 3 - SED and AWK

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Inlab 3 - Q2</b>	<b>5</b>
2.1	Problem Statement . . . . .	5
2.2	Algorithm . . . . .	6
2.3	Code . . . . .	7
2.4	Results . . . . .	8
<b>3</b>	<b>Outlab 3 - Q4</b>	<b>9</b>
3.1	Problem Statement . . . . .	9
3.2	Algorithm . . . . .	9
3.3	Code . . . . .	11
3.4	Results . . . . .	13
<b>4</b>	<b>References</b>	<b>14</b>

# 1 Introduction

## SED

Sed is a stream editor. A stream editor is used to perform basic text transformations on an input stream (a file or input from a pipeline). While in some ways similar to an editor which permits scripted edits (such as ed), sed works by making only one pass over the input(s), and is consequently more efficient. But it is sed's ability to filter text in a pipeline which particularly distinguishes it from other types of editors.

The basic syntax is:

*sed 'sed\_command' filename*

For example:

To substitute a certain pattern1 with another pattern2 in a file:

*sed 's/pattern1/pattern2/' filename*

To print k lines of a file:

*sed -n 'kp' filename*

To delete k lines of a file:

*sed 'kd' filename*

For inplace substitutions:

*sed -i 's/pattern1/pattern2/' filename* [3]

## AWK

AWK (awk) is a domain-specific language designed for text processing and typically used as a data extraction and reporting tool. Like sed and grep, it is a filter, and is a standard feature of most Unix-like operating systems. The AWK language is a data-driven scripting language consisting of a set of actions to be taken against streams of textual data – either run directly on files or used as part of a pipeline – for purposes of extracting or transforming text, such as producing formatted reports. The language extensively uses the string datatype, associative arrays (that is, arrays indexed by key strings), and regular expressions. While AWK has a limited intended application domain and was especially designed to support one-liner programs, the language is Turing- complete.

The basic syntax is:

*awk 'BEGIN{} {} END{}' filename*

For example:

To print contents of a file:

*awk '{print}' filename* [2]

## SED vs AWK

SED	AWK
A command line utility that parses and transforms text, using a simple , compact programming language	A command line utility designed for text processing that allows effective programs in the form of statements
Less powerful	More powerful and robust
Simple and limited	Complex and versatile

In this lab I was required to write sed commands and awk scripts on bash for the various inlab and outlab problems. There were 3 inlab problems and 5 outlab problems. However I am only going to cover one problem from the inlab and one from the outlab in this report.

The following are the methods I used to solve the problems:

- InLab

1. Q1\_a :- awk
2. Q1\_b :- awk
3. Q2 :- awk
4. Q3 :- sed

- OutLab

1. Q1 :- sed
2. Q2 :- sed
3. Q3 :- awk
4. Q4 :- awk
5. Q5 :- awk and sed

## 2 Inlab 3 - Q2

### 2.1 Problem Statement

Create a function which takes in input a set of numbers from a text file in base  $8+2*(\text{line\_number}\%3)$  and returns the same number in base 10. Note that the digits will be space separated on each line. For digits greater than 9, lowercase alphabets will be used.

```
251-TA/sed-awk/inlab$ cat sampleq2_input.txt
1 2 3 4
2 a
1 0
2 5
b 7
```

Sample Input

```
251-TA/sed-awk/inlab$ ./Q2_awk.sh sampleq2_input.txt
1234
34
8
25
139
```

Sample Output

## 2.2 Algorithm

The algorithm that my code follows is:

---

**Algorithm 1:** InLab - Q2

---

```
while read line do
    num_digits  $\leftarrow$  number of fields(NF);
    for  $i \leftarrow 1$  to num_digits do
        | digits[i - 1]  $\leftarrow$  $i(Value of that field);
        | /* digits gets 10 if value is a, 11 for b and 12 for c          */
    end
    base  $\leftarrow 8 + 2 \cdot \text{NR}(\text{Number of row}) \% 3$ ;
    sum  $\leftarrow 0$ ;
    for  $i \leftarrow 0$  to num_digits - 1 do
        | sum  $\leftarrow$  sum + digits[i]  $\cdot$  base(num_digits-i-1);
    end
    print sum;
end
```

---

## 2.3 Code

The code that I wrote is:

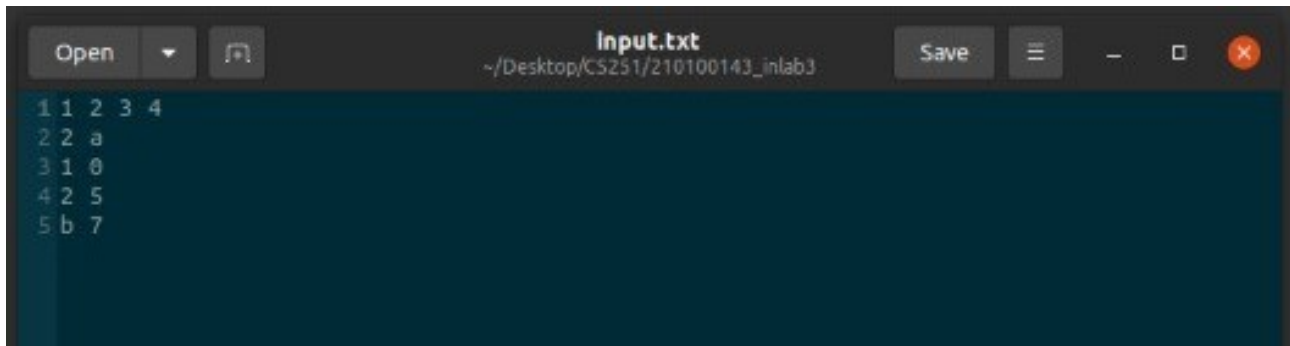
References from : [\[4\]](#) [\[5\]](#) [\[1\]](#)

InLab - Q2

```
1  #!/bin/bash
2
3  awk '
4  {
5      num_digits=NF;
6      for(i=1;i<=NF;i++)
7      {
8          if($i=="a")
9              {digits[i-1]=10}
10             else if($i=="b")
11                 {digits[i-1]=11}
12             else if($i=="c")
13                 {digits[i-1]=12}
14             else {digits[i-1]=$i}
15         }
16         base=8+2*(NR%3)
17         sum=0;
18         for(i=0;i<num_digits;i++)
19         {
20             mult=1;
21             for(j=0; j<num_digits-i-1; j++)
22             {
23                 mult=mult*base;
24             }
25             sum=sum+digits[i]*mult;
26         }
27         print sum
28     }, $1
```


## 2.4 Results

The **input**, the **command** I ran and the **output** I got are:

A screenshot of a text editor window titled 'input.txt' with a dark theme. The window has 'Open', 'Save', and window control buttons at the top. The file path is '~/.Desktop/CS251/210100143\_inlab3'. The content of the file is:

```
1 1 2 3 4
2 2 a
3 1 0
4 2 5
5 b 7
```

Input

A screenshot of a terminal window with a dark theme. The window title is 'shubham@shubham-IdeaPad-L340-15IRH-Gaming: ~/.Desktop...'. The prompt is 'shubham@bash:~/.Desktop/CS251/210100143\_inlab3\$'. The command entered is './Q2\_awk.sh input.txt'.

```
shubham@bash:~/.Desktop/CS251/210100143_inlab3$ ./Q2_awk.sh input.txt
```

Command

A screenshot of a terminal window with a dark theme. The window title is 'shubham@shubham-IdeaPad-L340-15IRH-Gaming: ~/.Desktop...'. The prompt is 'shubham@bash:~/.Desktop/CS251/210100143\_inlab3\$'. The command entered is './Q2\_awk.sh input.txt'. The output is:

```
1234
34
8
25
139
shubham@bash:~/.Desktop/CS251/210100143_inlab3$
```

Output



## 3 Outlab 3 - Q4

### 3.1 Problem Statement

Your task for this question is to perform addition of two numbers and write the result to file for each test case.

#### Sample Input:

```
1
10 2
9 9 9
1 0 0
```

#### Input description:

First line of the file indicates the number of test cases. Second Line gives the base of the input and output separated by spaces. The next two lines of each test case have two numbers. The digits of the numbers are space separated. The base is guaranteed to be less than or equal 16.

#### Expected output for the above input:

```
1 0 0 0 1 0 0 1 0 1 1
```

#### Output description:

The digits in each result should be space separated. Each line should contain only one result.

Write the bash script q4.sh that takes two text files, one containing input and other to write the output. Make use of **awk** (mandatory!) to solve the task.

### 3.2 Algorithm

The following algorithm was written by using the algorithm2e package which is able to make the most good looking algorithms, however this package does not allow page breaks and the following algorithm is too big to fit in one page with the section heading so I removed the page break from the above section and added this algorithm section to the same page so the algorithm is able to render fully in the page below.

The algorithm that my code follows is:

---

**Algorithm 2:** OutLab - Q4

---

```
while read line do
  if NR==1 then
    | testcases=$1;
  end
  if NR%3==2 then
    | input_base=$1;
    | output_base=$2;
  end
  if NR%3==0 then
    | num_digits1=NF;
    | for i ← 1 to num_digits1 do
    |   | digits1[i] ← $i(Value of that field);
    |   | /* digits gets 10 if value is a, 11 for b and so on till 15 for
    |   | f                                                                */
    | end
    | num1 ← 0;
    | for i ← 0 to num_digits1 - 1 do
    |   | num1 ← num1 + digits[i] · input_base(num_digits1-i-1);
    | end
  end
  if NR%3==0 and NR!=1 then
    | num_digits2=NF;
    | for i ← 1 to num_digits2 do
    |   | digits2[i] ← $i(Value of that field);
    |   | /* digits gets 10 if value is a, 11 for b and so on till 15 for
    |   | f                                                                */
    | end
    | num2 ← 0;
    | for i ← 0 to num_digits2 - 1 do
    |   | num2 ← num2 + digits[i] · input_base(num_digits2-i-1);
    | end
    | sum ← num1 + num2;
    | num_digits=0;
    | i=0;
    | while sum > 0 do
    |   | digits[i]=int(sum%output_base);
    |   | sum=int(sum/output_base);
    |   | num_digits++;
    |   | i++;
    | end
    | for i ← num_digits - 1 to 0 do
    |   | print digits[i]
    | end
    | print new line
  end
end
```

---

### 3.3 Code

The code that I wrote is:

References from : [\[4\]](#) [\[5\]](#) [\[1\]](#) [\[6\]](#)

OutLab - Q4

```
1  #!/bin/bash
2
3  awk '{
4      if (NR==1)
5      {
6          testcases=$1
7      }
8      if (NR%3==2)
9      {
10         input_base=$1;
11         output_base=$2;
12     }
13     if (NR%3==0)
14     {
15         num_digits1=NF;
16         for(i=1;i<=NF;i++)
17         {
18             if($i>=0 && $i<=9)
19             {digits1[i-1]=$i}
20             else if($i=="a")
21             {digits1[i-1]=10}
22             else if($i=="b")
23             {digits1[i-1]=11}
24             else if($i=="c")
25             {digits1[i-1]=12}
26             else if($i=="d")
27             {digits1[i-1]=13}
28             else if($i=="e")
29             {digits1[i-1]=14}
30             else if($i=="f")
31             {digits1[i-1]=15}
32             else
33             {digits1[i-1]=$i}
34         }
35         num1=0;
36         for(i=0;i<num_digits1;i++)
37         {
38             mult=1;
39             for(j=0; j<num_digits1-i-1; j++){
40                 mult=mult*input_base;
41             }
42             num1=num1+digits1[i]*mult;
43         }
44     }
```

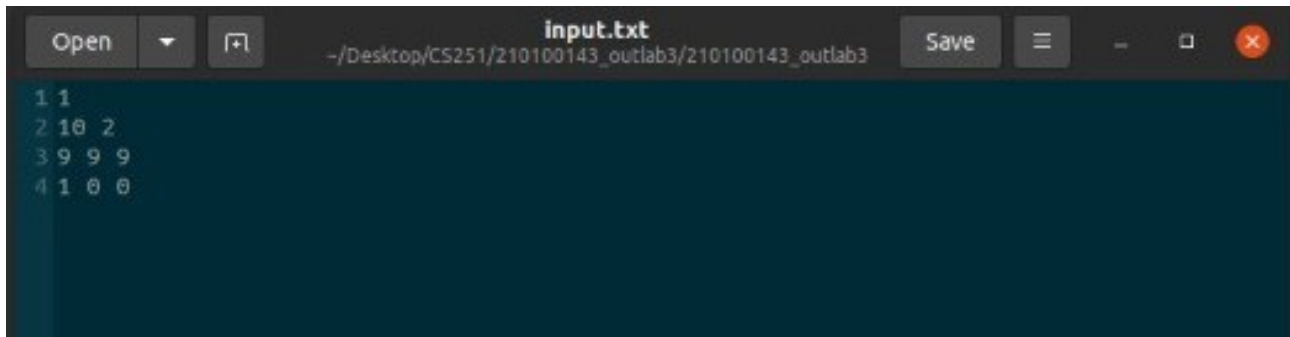
```

45         if(NR%3==1 && NR!=1)
46         {
47             num_digits2=NF;
48             for(i=1;i<=NF;i++)
49             {
50                 if($i>=0 && $i<=9)
51                 {digits2[i-1]=$i}
52                 else if($i=="a")
53                 {digits2[i-1]=10}
54                 else if($i=="b")
55                 {digits2[i-1]=11}
56                 else if($i=="c")
57                 {digits2[i-1]=12}
58                 else if($i=="d")
59                 {digits2[i-1]=13}
60                 else if($i=="e")
61                 {digits2[i-1]=14}
62                 else if($i=="f")
63                 {digits2[i-1]=15}
64                 else
65                 {digits2[i-1]=$i}
66             }
67             num2=0;
68             for(i=0;i<num_digits2;i++)
69             {
70                 mult=1;
71                 for(j=0; j<num_digits2-i-1; j++){
72                     mult=mult*input_base;
73                 }
74                 num2=num2+digits2[i]*mult;
75             }
76             sum = num1+num2;
77             num_digits=0
78             i=0
79             while(sum>0)
80             {
81                 digits[i]=int(sum%output_base)
82                 sum=int(sum/output_base)
83                 num_digits++
84                 i++
85             }
86             for(i=num_digits-1;i>=0;i--)
87             {
88                 printf "%d ",digits[i]
89             }
90             printf "\n"
91         }
92     }
93 } ' $1 > $2

```

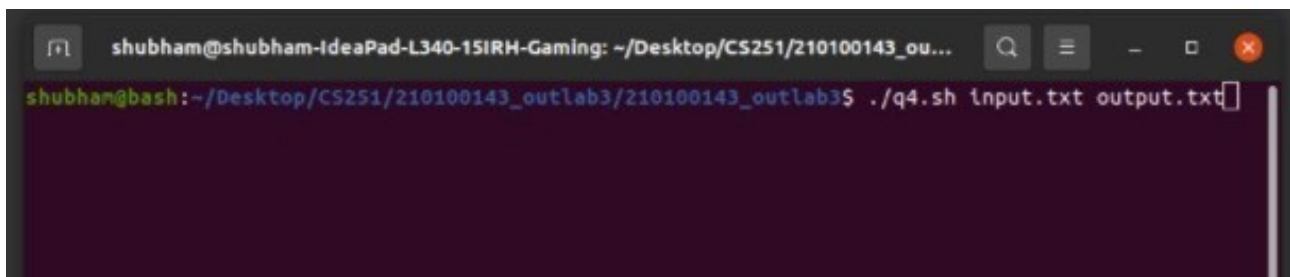
### 3.4 Results

The **input**, the **command** I ran and the **output** I got are:

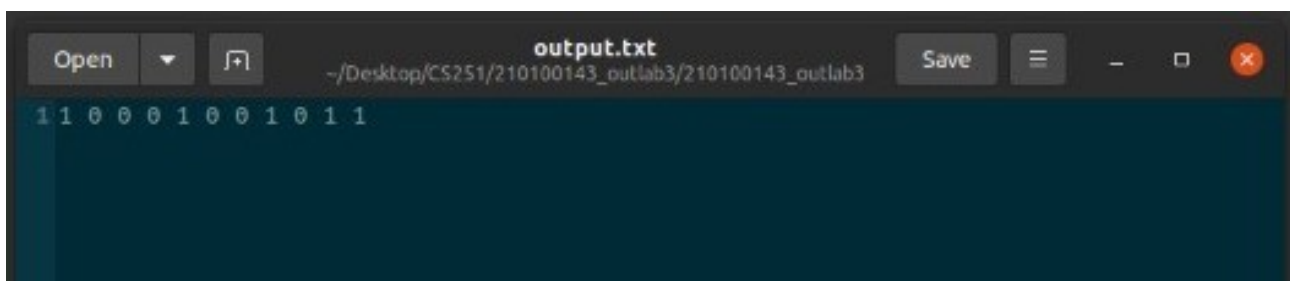
A screenshot of a text editor window titled 'input.txt'. The window has a dark theme and a toolbar with 'Open', 'Save', and other icons. The file path is shown as '~/.Desktop/CS251/210100143\_outlab3/210100143\_outlab3'. The content of the file is as follows:

```
1 1
2 10 2
3 9 9 9
4 1 0 0
```

Input

A screenshot of a terminal window. The window title is 'shubham@shubham-IdeaPad-L340-15IRH-Gaming: ~/.Desktop/CS251/210100143\_ou...'. The prompt is 'shubham@bash:~/.Desktop/CS251/210100143\_outlab3/210100143\_outlab3\$'. The command entered is './q4.sh input.txt output.txt'.

Command

A screenshot of a text editor window titled 'output.txt'. The window has a dark theme and a toolbar with 'Open', 'Save', and other icons. The file path is shown as '~/.Desktop/CS251/210100143\_outlab3/210100143\_outlab3'. The content of the file is as follows:

```
1 1 0 0 0 1 0 0 1 0 1 1
```

Output

## 4 References

- [1] Computerworld. Processing files with awk. <https://www.computerworld.com/article/2794577/processing-files-with-awk.html/>.
- [2] GeeksforGeeks. AWK command in Unix/Linux. <https://www.geeksforgeeks.org/awk-command-unixlinux-examples/>.
- [3] GeeksforGeeks. Sed Command in Linux/Unix. <https://www.geeksforgeeks.org/sed-command-in-linux-unix-with-examples/>.
- [4] Oreilly. Reading Input Files. <https://www.oreilly.com/library/view/effective-awk-programming/9781491904930/ch04.html/>.
- [5] Unix shell programming. Passing value as a command line argument in awk script. <https://www.unix.com/shell-programming-and-scripting/163946-passing-value-command-line-argument-awk-script.html/>.
- [6] Unix-stackexchange. AWK looping through files. <https://unix.stackexchange.com/questions/426335/using-awk-and-looping-through-files-in-a-directory/>.