

FOSS LAB

Experiments 11

Shell Scripting using PERL and Awk

Submitted By:

Mohammed Rabeeh

Roll No: 35

TVE18CS036

Contents

1	Experiment 11 - A: PERL Scripting	2
1.1	Word processing	2
1.1.1	Aim	2
1.1.2	Program Code	2
1.1.3	Input/Output	4
1.1.4	Result	5
2	Experiment 11 - B: Awk Scripting	6
2.1	Date fomattting	6
2.1.1	Aim	6
2.1.2	Program Code	6
2.1.3	Input/Output	7
2.1.4	Result	7
2.2	Delete duplicate lines	8
2.2.1	Aim	8
2.2.2	Program Code	8
2.2.3	Input/Output	9
2.2.4	Result	9
2.3	Calculate total sales	10
2.3.1	Aim	10
2.3.2	Program Code	10
2.3.3	Input/Output	11
2.3.4	Result	11
2.4	Gross Salary	12
2.4.1	Aim	12
2.4.2	Program Code	12
2.4.3	Input/Output	13
2.4.4	Result	13

1 Experiment 11 - A: PERL Scripting

1.1 Word processing

Date: 6th February 2020

1.1.1 Aim

Create a text file and answer the following queries:

- a) Search for the pattern 'apple' in the file and display the number of occurrences.
- b) Count the number of words that ends with 'e'.
- c) Count the number of words that starts with 'ap'.
- d) Search for words containing 'a' or 's'.
- e) Search for words containing zero or more occurrence of 'e'.
- f) Search for words containing one or more occurrence of 'e'.
- g) Search for words containing the letters 'l' and 'm', with any number of characters in between.

1.1.2 Program Code

```
#!/usr/bin/perl

# Author: Mohammed Rabeeh
# Date: 6th February 2020
# OS: macOS Catalina 10.15.2
# Shell: zsh
# Function: Searches the text file for the following queries
# a) Search for the pattern 'apple' in the
# file and display the number of occurrences.
# b) Count the number of words that ends with 'e'
# c) Count the number of words that starts with 'ap'
# d) Search for words containing 'a' or 's'
# e) Search for words containing zero or more occurrence of 'e'
```

```

# f) Search for words containing one or more occurrence of 'e'
# g) Search for words containing the letters 'l' and 'm',
#     with any number of characters in between
# Input: file.txt

```

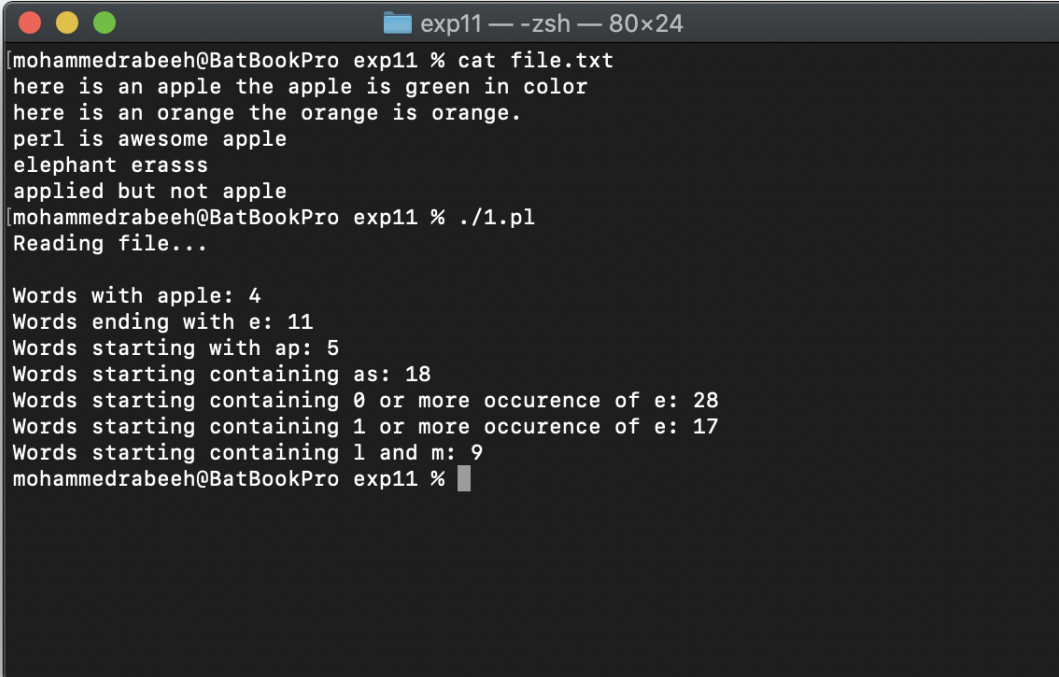
```

open(FH, "file.txt") or die "Couldn't open the file.";
print "Reading file... \n\n";
my $apple_count = 0;
my $e_count = 0;
my $ap_count = 0;
my $as_count = 0;
my $total_count = 0;
my $e_oc_count = 0;
my $lm_count = 0;
while(my $file = <FH>) {
    foreach my $str (split /\s+/, $file) {
        $total_count++;
        if($str =~ /apple/) {
            $apple_count++;
        }
        if($str =~ /e$/) {
            $e_count++;
        }
        if($str =~ /^ap/) {
            $ap_count++;
        }
        if($str =~ /[as]/) {
            $as_count++;
        }
        if($str =~ /e/) {
            $e_oc_count++;
        }
        if($str =~ /[lm]/) {
            $lm_count++;
        }
    }
}

```

```
print("Words with apple: $apple_count\n");
print("Words ending with e: $e_count\n");
print("Words starting with ap: $ap_count\n");
print("Words starting containing as: $as_count\n");
print("Words starting containing 0 or more occurrences of e: $total_count\n");
print("Words starting containing 1 or more occurrences of e: $e_oc_count\n");
print("Words starting containing l and m: $lm_count\n");
close;
```

1.1.3 Input/Output



```
exp11 — zsh — 80x24
[mohammedrabeeh@BatBookPro exp11 % cat file.txt
here is an apple the apple is green in color
here is an orange the orange is orange.
perl is awesome apple
elephant erasss
applied but not apple
[mohammedrabeeh@BatBookPro exp11 % ./1.pl
Reading file...

Words with apple: 4
Words ending with e: 11
Words starting with ap: 5
Words starting containing as: 18
Words starting containing 0 or more occurrence of e: 28
Words starting containing 1 or more occurrence of e: 17
Words starting containing l and m: 9
mohammedrabeeh@BatBookPro exp11 %
```

1.1.4 Result

A PERL script was created as per the requirements mentioned in the problem. A file called "file.txt" was created and processed using the script and the output was verified. The script was executed on macOS Catalina 10.15.2 in zsh shell version 5.7.1.

2 Experiment 11 - B: Awk Scripting

2.1 Date fomatting

Date: 6th February 2020

2.1.1 Aim

To write a awk script that accepts date argument in the form of mm-dd-yy and displays it in the following format. The script should check the validity of the argument and in the case of error, display a suitable message. Sample Input : 12-10-2008. Sample Output : The day is 10 The month is OCT The year is 2008.

2.1.2 Program Code

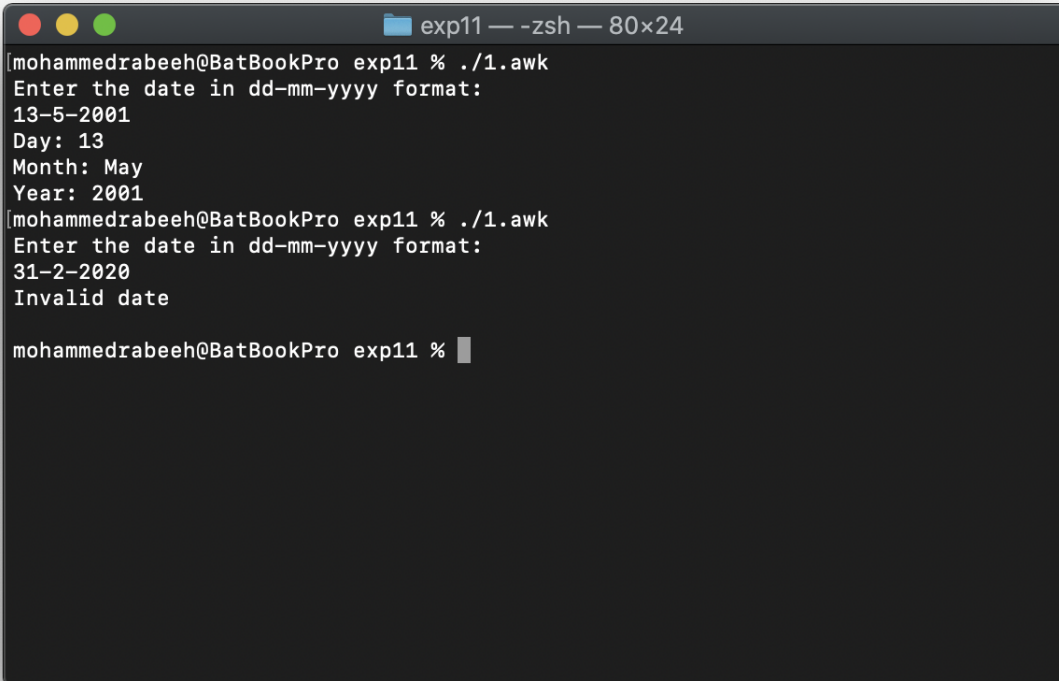
```
#!/usr/bin/awk -f

# Author: Mohammed Rabeeh
# Date: 6th February 2020
# OS: macOS Catalina 10.15.2
# Shell: zsh
# Function: Formats the date from dd-mm-yyyy format to Day, Month and Year
# Input: Date in dd-mm-yyyy format
BEGIN{
    FS="-"
    print "Enter the date in dd-mm-yyyy format: "
    getline < "/dev/tty"

    if(((($3%4!=0) && ($2==2) && ($1>28)) ||
        (($3%4==0) && ($2==2) && ($1>29)) ||
        $2 > 12)
        print "Invalid date\n"
    else
    {
        split("January February March April
        May June July August September October
        November December", month, " ")
        print "Day: " $1
```

```
        print "Month: " month[$2]
        print "Year: " $3
    }
}
```

2.1.3 Input/Output

A terminal window titled 'exp11 — zsh — 80x24' showing the execution of an awk script. The user enters a valid date '13-5-2001' and an invalid date '31-2-2020'. The script outputs the day, month, and year for the valid date and an error message for the invalid date.

```
exp11 — zsh — 80x24
[mohammedrabeeh@BatBookPro exp11 % ./1.awk
Enter the date in dd-mm-yyyy format:
13-5-2001
Day: 13
Month: May
Year: 2001
[mohammedrabeeh@BatBookPro exp11 % ./1.awk
Enter the date in dd-mm-yyyy format:
31-2-2020
Invalid date

mohammedrabeeh@BatBookPro exp11 %
```

2.1.4 Result

Date formatting using awk was scripted and the output was verified. The script was executed on macOS Catalina 10.15.2 in zsh shell version 5.7.1.

2.2 Delete duplicate lines

Date: 6th February 2020

2.2.1 Aim

To write an awk script to delete duplicated line from a text file. The order of the original lines must remain unchanged.

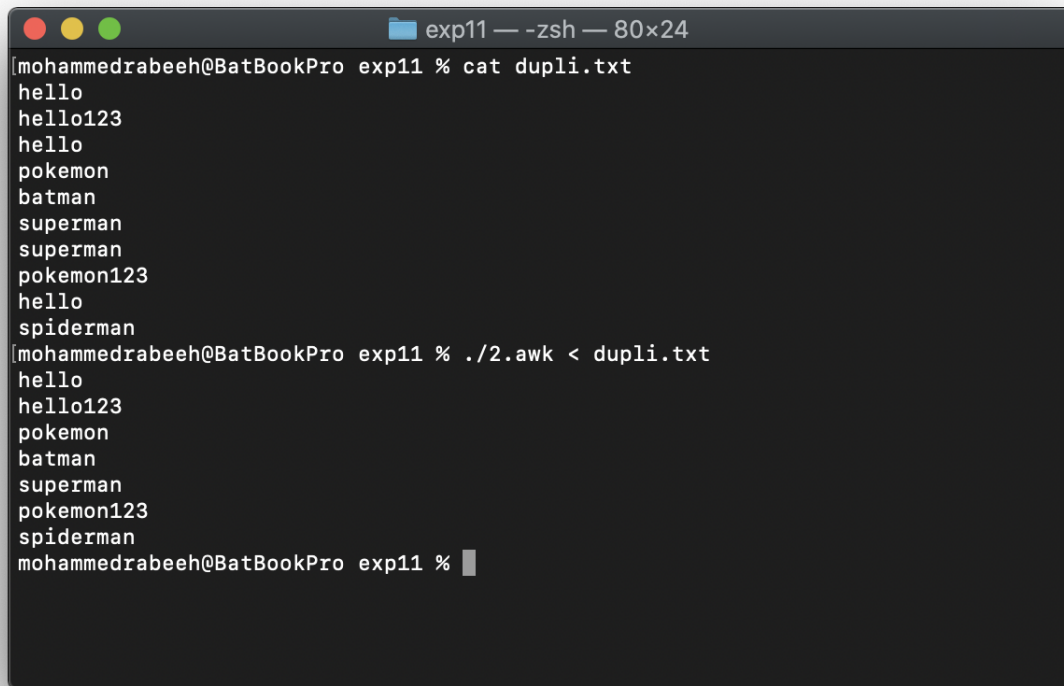
2.2.2 Program Code

```
#!/usr/bin/awk -f

# Author: Mohammed Rabeeh
# Date: 6th February 2020
# OS: macOS Catalina 10.15.2
# Shell: zsh
# Function: Remove duplicate lines from text and prints it.
# Syntax: ./2.awk filename

!already_present[$0]++ {print $0}
```

2.2.3 Input/Output



```
exp11 — -zsh — 80x24
[mohammedrabeeh@BatBookPro exp11 % cat dupli.txt
hello
hello123
hello
pokemon
batman
superman
superman
pokemon123
hello
spiderman
[mohammedrabeeh@BatBookPro exp11 % ./2.awk < dupli.txt
hello
hello123
pokemon
batman
superman
pokemon123
spiderman
mohammedrabeeh@BatBookPro exp11 %
```

2.2.4 Result

Deleting of duplicate lines using awk was scripted and the output was verified. A sample file called "dupli.txt" was created to verify the output. The script was executed on macOS Catalina 10.15.2 in zsh shell version 5.7.1.

2.3 Calculate total sales

Date: 6th February 2020

2.3.1 Aim

To write an awk script to find out total number of books sold in each discipline as well as total book sold based on the given table
electrical 34
mechanical 67
electrical 80
computers 43
mechanical 65
civil 198
computers 64

2.3.2 Program Code

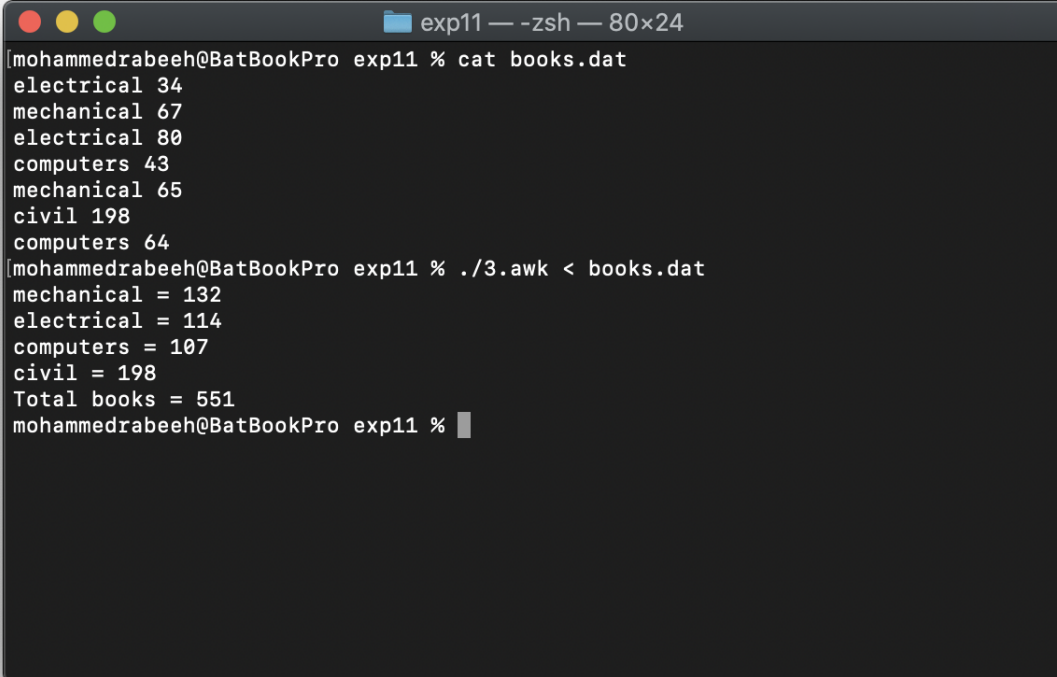
```
#!/usr/bin/awk -f

# Author: Mohammed Rabeeh
# Date: 6th February 2020
# OS: macOS Catalina 10.15.2
# Shell: zsh
# Function: Calculates no. of books sold in each discipline and in total.
# Syntax: ./3.awk file_input

{
    arr[$1]+=$2;
    total+=$2;
}

END {
    for (i in arr){
        print i " = " arr[i];
    }
    print "Total books = " total
}
```

2.3.3 Input/Output



```
exp11 — zsh — 80x24
[mohammedrabeeh@BatBookPro exp11 % cat books.dat
electrical 34
mechanical 67
electrical 80
computers 43
mechanical 65
civil 198
computers 64
[mohammedrabeeh@BatBookPro exp11 % ./3.awk < books.dat
mechanical = 132
electrical = 114
computers = 107
civil = 198
Total books = 551
mohammedrabeeh@BatBookPro exp11 %
```

2.3.4 Result

The sales in each discipline as well as the total sales were calculated using an awk script. The input table was given to the program through a file called "books.dat". The script was successfully executed on macOS Catalina 10.15.2 in zsh shell version 5.7.1 and the output was verified.

2.4 Gross Salary

Date: 6th February 2020

2.4.1 Aim

To write an awk script to compute gross salary of an employee accordingly to rule given below : If basic salary less 10000 then DA = 45% of the basic and HRA =15% of basic If basic salary greater than or equal to 10000 then DA =50% of the basic and HRA =20% of basic.

2.4.2 Program Code

```
#!/usr/bin/awk -f

# Author: Mohammed Rabeeh
# Date: 6th February 2020
# OS: macOS Catalina 10.15.2
# Shell: zsh
# Function: Takes basic salary of an employee and calculates the
#           gross salary of the employee.
# Input: basic salary

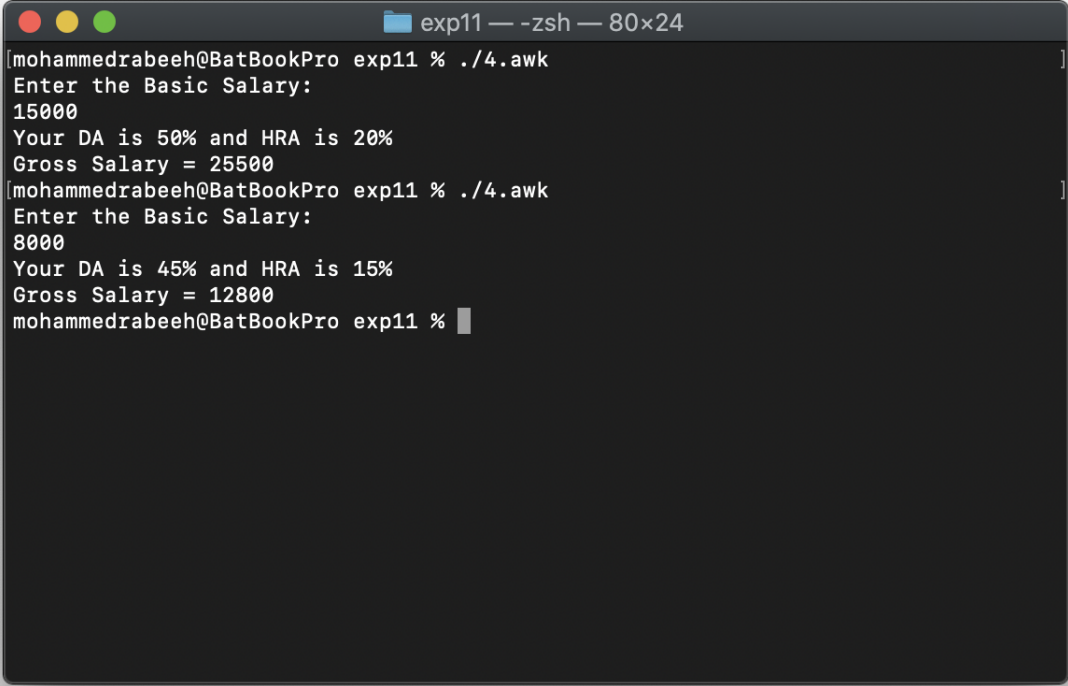
BEGIN{
    print "Enter the Basic Salary: ";
    getline < "/dev/tty";

    if($0<10000){
        DA = 45/100 * $0;
        HRA = 15/100 * $0;
        print "Your DA is 45% and HRA is 15%"
    }
    else{
        DA = 50/100 * $0;
        HRA = 20/100 * $0;
        print "Your DA is 50% and HRA is 20%"
    }

    gross_sal = $0 + DA + HRA;
```

```
    print "Gross Salary = " gross_sal  
}
```

2.4.3 Input/Output



```
exp11 — zsh — 80x24  
[mohammedrabeeh@BatBookPro exp11 % ./4.awk  
Enter the Basic Salary:  
15000  
Your DA is 50% and HRA is 20%  
Gross Salary = 25500  
[mohammedrabeeh@BatBookPro exp11 % ./4.awk  
Enter the Basic Salary:  
8000  
Your DA is 45% and HRA is 15%  
Gross Salary = 12800  
mohammedrabeeh@BatBookPro exp11 % ]
```

2.4.4 Result

The gross salary of an employee based on their basic salary was calculated. Sample salaries of 15000 and 8000 were used. The script was successfully executed on macOS Catalina 10.15.2 in zsh shell version 5.7.1 and the output was verified.