Ex. No.: 4a)

Date:

EMPLOYEE AVERAGE PAY

Aim:

To find out the average pay of all employees whose salary is more than 6000 and no. of days worked is more than 4.

Algorithm:

1. Create a flat file emp.dat for employees with their name, salary per day and number of days worked and save it.
2. Create an awk script emp.awk
3. For each employee record do
 a. If Salary is greater than 6000 and number of days worked is more than 4, then print name and salary earned
 b. Compute total pay of employee
4. Print the total number of employees satisfying the criteria and their average pay.

Program Code:

```
BEGIN {
    total_pay = 0
    count = 0
    print "EMPLOYEES DETAILS"
}

{
    if ($2 > 6000 && $3 > 4) {
        employee_pay = $2 * $3
        printf "%s %d\n", $1, employee_pay
        total_pay += employee_pay
        count++
    }
}

END {
    if (count > 0) {
        printf "no of employees are= %d\n", count
        printf "total pay= %d\n", total_pay
        printf "average pay= %.1f\n", total_pay/count
    } else {
        print "No employees meet the criteria"
    }
}
```

Sample Input:

//emp.dat – Col1 is name, Col2 is Salary Per Day and Col3 is //no. of days worked

JOE 8000 5
RAM 6000 5
TIM 5000 6
BEN 7000 7
AMY 6500 6

Output:

Run the program using the below commands

[student@localhost ~]$ vi emp.dat
[student@localhost ~]$ vi emp.awk
[student@localhost ~]$ gawk -f emp.awk emp.dat.

EMPLOYEES DETAILS
JOE 40000
BEN 49000
AMY 39000
no of employees are= 3
total pay= 128000
average pay= 42666.7
[student@localhost ~]$

Result:

The Awk script for calculating employee average pay has been successfully executed, accurately processing employee data and providing the average salary.

Ex. No.: 4b)

Date:

RESULTS OF EXAMINATION

Aim:

To print the pass/fail status of a student in a class.

Algorithm:

1. Read the data from file
2. Get a data from each column
3. Compare the all subject marks column
    a. If marks less than 45 then print Fail
    b. else print Pass

Program Code:

//marks.awk

```
BEGIN {
    # Print header
    printf "%-10s %-6s %-6s %-6s %-6s %-6s %-6s %-10s\n",
           "NAME", "SUB-1", "SUB-2", "SUB-3", "SUB-4", "SUB-5", "SUB-6", "STATUS"
    print "-------------------------------------------------------------------"
}

{
    status = "PASS"  # Default status

    # Check each subject mark (columns 2-7)
    for (i = 2; i <= 7; i++) {
        if ($i < 45) {
            status = "FAIL"
            break  # No need to check further if one subject fails
        }
    }

    # Print student record with status
    printf "%-10s %-6d %-6d %-6d %-6d %-6d %-6d %-10s\n",
           $1, $2, $3, $4, $5, $6, $7, status
}
```

Input:

//marks.dat
//Col1- name, Col 2 to Col7 – marks in various subjects
BEN 40 55 66 77 55 77
TOM 60 67 84 92 90 60
RAM 90 95 84 87 56 70
JIM 60 70 65 78 90 87

Output:
Run the program using the below command
[root@localhost student]# gawk -f marks.awk marks.dat

NAME SUB-1 SUB-2 SUB-3 SUB-4 SUB-5 SUB-6 STATUS

BEN 40 55 66 77 55 77 FAIL TOM 60 67 84 92 90 60 PASS RAM 90 95 84 87 56 70 PASS JIM 60 70 65 78 90 87 PASS

Result:

The Awk script for processing examination results has been successfully executed, extracting and displaying relevant information like pass/fail status and scores