

# 实验 05 Awk 实验

班级：数据科学与大数据技术 2 班

学号：202026203039

姓名：赖丽婷

用户名：llt

## 一、实验目的

1. 练习使用 `awk` 完成文本数据的处理

## 二、实验要求

1. 填写实验报告，请将关键命令及其结果进行截图(请确保截图中的文字清晰可见)
2. 导出为 pdf 文件，文件名为用户名-姓名-lab05.pdf，在规定截止时间之前上传作业)
3. 以下步骤中所有 **s01** 请换成你自己的用户名。

## 三、实验步骤

1. 学生成绩统计打印

现有学生成绩清单 `score.list` 如下

```
$ cat score.list
```

```
jasper: 80 82 84 84 88 92
```

```
andrea: 85 89 90 90 94 95
```

```
ellis: 89 90 92 96 96 98
```

```
mona: 70 70 77 83 85 89
```

```
john: 78 85 88 91 92 94
```

```
dunce: 60 60 61 62 64 80
```

请编写 `awk` 脚本 **s01**-grade.awk 进行统计：

(1)每位同学的平均分及等级(A,B,C,D,F)

(2)班平均成绩

(3)平均成绩高于或等与班平均的人数

(4)平均成绩低于班平均的人数

(5)每个等级的人数

(请在下面贴出源代码)

```
{
    suma = ($2+$3+$4+$5+$6+$7)/(NF-1)
    all = all+suma
    if(suma>=90){
        a = a+1
        print $1 suma " A"
    }
    else if(suma>=80){
        b = b+1
        print $1 suma " B"
    }
    else if(suma>=70){
        c = c+1
        print $1 suma " C"
    }
    else if(suma>=60){
        d = d+1
        print $1 suma " D"
    }
    else{
        f = f+1
        print $1 suma " F"
    }
}
{
    avgall = all/NR
    if(suma >= avgall)
        counta = counta +1
    else
        countb = countb +1
}

END {print "Class Average: " avgall
      print "At or Above Average: " counta
      print "Below Average: " countb
      if(a>0)
          print "A: " a
      if(b>0)
          print "B: " b
      if(c>0)
          print "C: " c
      if(d>0)
          print "D: " d
      if(f>0)
          print "F: " f
      }
```

(请在下面贴出执行情况截图)

```
llt@llt-virtual-machine:~/5$ awk -f llt-grade.awk score.list
jasper:85 B
andrea:90.5 A
ellis:93.5 A
mona:79 C
john:88 B
dunce:64.5 D
Class Average: 83.4167
At or Above Average: 4
Below Average: 2
A: 2
B: 2
C: 1
D: 1
llt@llt-virtual-machine:~/5$
```

## 2. 金额转中文文本

请编写金额转换脚本 **s01-number2zh**，将低于 100 万元的金额（带固定两位小数）转换成中文大写金额形式。

要求逐行处理输入数据：

- (1) 检查输入是否为非负整数或非负小数，如果不是则打印出错信息；
- (2) 检查输入是否小于 100 万，如果不是则打印出错信息；
- (3) 输入需要首先统一转换为带两位小数的形式；
- (4) 将带两位小数的数值转换为中文大写金额字符串并打印输出。

测试：(要求能正确处理整圆情况和前面要加零的情况以及分为零的情况)

\$ cat amounts.txt

-300  
123456.789  
1234567.89  
+987654.30  
123456.00  
3.4.5  
0  
100301  
100300  
100300.05  
3000  
300001

\$ awk -f a01-num2zh amounts.txt

-300	Warn: Not a nonnegative number
123456.789	壹拾貳萬叁仟肆佰伍拾陸圓柒角玖分
1234567.89	Warn: Not less than 1000000
+987654.30	玖拾捌萬柒仟陸佰伍拾肆圓叁角
123456.00	壹拾貳萬叁仟肆佰伍拾陸圓整
3.4.5	Warn: Not a nonnegative number
0	零圓整
100301	壹拾萬零叁佰零壹圓整
100300	壹拾萬零叁佰圓整
100300.05	壹拾萬零叁佰圓零角伍分
3000	叁仟圓整
300001	叁拾萬零壹圓整

(请在下面贴出源代码)

```
llt@llt-virtual-machine: ~/5
$1>=10000000{
    print $1 "Warn: Not less than 10000000"
next
}
$1<0{
    print $1 "Warn: Not a nonnegative number"
next
}
{
    init()
    if($1<10000000){
        amount=sprintf("%.2f",$1)
        printf("%s%-20s\n",$1,num2str(amount))
    }
}
function num2str(n)
{
    cents=substr(n,length(n)-1,2)
    dols=substr(n,1,length(n)-3)
    return int1str(dols)"圓"int2str(cents)
}
function int1str(n)
{
    if(length(n)>=6)
        return int1str(substr(n,1,1)) "拾" int1str(substr(n,2,length(n)))
}
1,1 顶
```

```

        else
            return int1str(substr(n,1,1)) "佰" int1str(substr(n,2,length(n)))
        if(length(n)>=2)
            if(substr(n,1,1)=="0" && substr(n,2,2)!="0")
                return "零"int1str(substr(n,2,length(n)))
            else if(substr(n,1,1)=="0" && substr(n,2,2)=="0")
                return ""
            else
                return int1str(substr(n,1,1)) "拾" int1str(substr(n,2,length(n)))
        if(length(n)>=1)
            return nums[int(n)+1]
    }
    function int2str(n)
    {
        if(n=="00")
            return "整"
        if(n~/[0-9][0]/)
            return nums[int(substr(n,1,1))+1]"角"
        else
            return nums[int(substr(n,1,1))+1]"角"nums[int(substr(n,2,2))+1]"分"
    }
    function init()
    {
        split("零,壹,贰,叁,肆,伍,陆,柒,捌,玖",nums,",")
    }

```

(请在下面贴出执行情况截图)

```

llt@llt-virtual-machine:~/5$ awk -f llt-number2zh amounts.txt
-300Warn: Not a nonegative number
123456.789壹拾贰万叁仟肆佰伍拾陆圆柒角玖分
1234567.89Warn: Not less than 10000000
+987654.30玖拾捌万柒仟陆佰伍拾肆圆叁角
123456.00壹拾贰万叁仟肆佰伍拾陆圆整
0零圆整
100301壹拾万零叁佰零壹圆整
100300壹拾万零叁佰圆整
100300.05壹拾万零叁佰圆零角伍分
3000叁仟圆整
300001叁拾万零壹圆整

```

### 3. 矩阵行列转置

请编写矩阵转置脚本 **s01-transpos**，对给定的矩阵输出其转置矩阵。

测试：

```

$ cat matrix
11 22 33 44
55 66 77 88
99 58 19 91
$ awk -f s01-transpos matrix
11 55 99
22 66 58
33 77 19
44 88 91

```

(请在下面贴出源代码)

```

{
    for(i=0; i<NF; i++){
        result[i,NR-1] = $(i+1)
    }
}
END {
    for(i=0; i<NF; i++){
        for(j=0; j<NR; j++){
            printf result[i,j]"\t"
        }
        print ""
    }
}

```

(请在下面贴出执行情况截图)

```
llt@llt-virtual-machine:~/5$ vim llt-transpos.awk
llt@llt-virtual-machine:~/5$ awk -f llt-transpos.awk matrix.list
11      55      99
22      66      58
33      77      19
44      88      91
llt@llt-virtual-machine:~/5$
```