

ΑΣΚΗΣΗ 1

LAB21142476

Θέμα 1

Σε αυτήν την άσκηση υλοποιήσαμε ένα πρόγραμμα που υπολογίζει τις παραμέτρους της γραμμικής παλινδρόμησης a, b, c που ελαχιστοποιούν το συνολικό τετραγωνικό σφάλμα err της προσέγγισης $cY=aX+b$. Το πρόγραμμα δέχεται ως είσοδο μια λίστα με αρχεία, τα οποία περιέχουν γραμμές της μορφής : num1:num2 και εκτυπώνει τις παραμέτρους a, b, c και το σφάλμα err .

Συγκεκριμένα για κάθε αρχείο εισόδου, το πρόγραμμα καλεί τη συνάρτηση Calculate με όρισμα το όνομα του αρχείου για να υπολογιστούν οι τιμές των παραμέτρων και να εκτυπωθούν τα αποτελέσματα.

```
while [[ $# -gt 0 ]]; do
    calculate $1
    shift #after the shift $1 = $2, $2 = $3 ...
done
```

Για να μπορεί το πρόγραμμα να δέχεται τυχαίο αριθμό αρχείων, χρησιμοποιήθηκε η εντολή shift του Bash και ένας βρόχος while. Όσο υπάρχουν αρχεία για επεξεργασία ($\$# > 0$) καλούμε την συνάρτηση με όρισμα το όνομα του αρχείου που βρίσκεται στο \$1 και στη συνέχεια κάνουμε shift.

Στη συνέχεια χρησιμοποιούμε την εντολή wc για να βρούμε τον αριθμό των γραμμών του αρχείου εισόδου.

```
length=$(( $(wc -l $1 | awk '{ print $1 }') +0 ))
```

Η έξοδος της εντολής wc είναι της μορφής #lines filename. Οπότε για να αφαιρέσουμε το όνομα του αρχείου χρησιμοποιήθηκε η εντολή awk ώστε να εμφανίσει μόνο τον αριθμό των γραμμών που βρίσκεται στο \$1, το όνομα του αρχείου είναι στο \$2.

Για τον υπολογισμό των αθροισμάτων χρησιμοποιήθηκε η εντολή awk. Χρησιμοποιούμε τον χαρακτήρα “.” για να χωρίσουμε την κάθε γραμμή και να πάρουμε x και y του κάθε σημείου. Μετά τον διαχωρισμό, ο αριστερός αριθμός βρίσκεται στο \$1 και ο δεξιός στο \$2.

```
sum_x=$(awk -F ':' '{s+=$1} END {print s}' $1)
```

Η παραπάνω εντολή υπολογίζει το άθροισμα X. Για κάθε γραμμή εκτελείται το τμήμα `{s+=$1}` όπου αθροίζει τις τιμές του X. Στο τέλος εκτελείται το `END {print s}` και εμφανίζει στην έξοδο το αποτέλεσμα. Όμοια υπολογίζονται και τα υπόλοιπα αθροίσματα.

Έπειτα υπολογίζουμε τα a και b με την εντολή bc και χρησιμοποιούμε την παράμετρο `scale=2` για να κρατήσουμε μόνο δύο δεκαδικά ψηφία. Για να αποφύγουμε τη διαίρεση με το μηδέν, υπολογίζουμε πρώτα τον παρονομαστή του a και αν είναι μηδέν, η συνάρτηση επιστρέφει.

Στη συνέχεια χρησιμοποιούμε την εντολή awk για να υπολογίσουμε το err.

```
err=$(awk -F ':' '{s+=(\ $2-($a * \ $1 + $b))^2 } END {print s}' $1)
err=$( echo "scale=2; $err/1" | bc)
```

Η δεύτερη εντολή χρησιμοποιείται για να κρατήσουμε μόνο δύο δεκαδικά ψηφία.

Τέλος εμφανίζουμε τις παραμέτρους και η συνάρτηση calculate επιστρέφει.

```
echo "FILE: $1, a=$a b=$b c=$c err=$err" | sed 's/=\../=0./g' | sed
's/=-\../=-0./g' | sed 's/\.00//g'
return
```

Η εντολή bc επιστρέφει δεκαδικούς αριθμούς στη μορφή .05. Για το λόγο αυτό χρησιμοποιήθηκε η εντολή sed, ώστε να εντοπίζει τους χαρακτήρες “=” και να τους αντικαθιστά με τους χαρακτήρες “=0.”. Η δεύτερη εντολή sed είναι για την περίπτωση που έχουμε αρνητικούς δεκαδικούς και αντικαθιστά το “=-.” με “=-0.”. Η τελευταία εντολή sed αφαιρεί τα περιττά μηδενικά, δηλαδή το 1.00 το κάνει 1.

Θέμα 2

Το πρόγραμμα δέχεται ως είσοδο ένα αρχείο που περιέχει γραμμές της μορφής Ομάδα1-Ομάδα2:Σκορ1-Σκορ2 με σκοπό να υπολογίσει τη βαθμολογία της κάθε ομάδας και στη συνέχεια να τις ταξινομήσει σύμφωνα με τη βαθμολογία τους.

Αρχικά για κάθε αγώνα υπολογίζεται η βαθμολογία, τα γκολ που σκόραρε και τα γκολ που δέχτηκε κάθε ομάδα.

```
awk -F"-|:" \
' $3>$4 { score1=3; score2=0}
  $3<$4 { score1=0; score2=3}
  $3==$4 {score1=1; score2=1}
  {print $1 "\t" score1 "\t" $3 "\t" $4 "\n" $2 "\t" score2 "\t" $4
"\t" $3}' $1 `
```

Αυτό γίνεται με την εντολή awk, η οποία χρησιμοποιεί τους χαρακτήρες “-” ή “:” για να διαχωρίσει κάθε γραμμή σε τέσσερις στήλες. Για κάθε γραμμή συγκρίνουμε το σκόρ, \$3 τα γκολ της πρώτης ομάδας και \$4 της δεύτερης και ανάλογα υπολογίζουμε πόσους βαθμούς θα πάρει κάθε ομάδα. Στη συνέχεια εμφανίζουμε για κάθε ομάδα τη βαθμολογία τα γκολ που σκόραρε και δέχτηκε. Δηλαδή για τη γραμμή

ARGENTINA-BOSNIA AND HERZEGOVINA:2-1

Η έξοδος γίνεται :

```
ARGENTINA 3 2 1
BOSNIA AND HERZEGOVINA 0 1 2
```

Στη συνέχεια αθροίζουμε τη βαθμολογία και τα γκολ της κάθε ομάδας από κάθε αγώνα και τα εμφανίζουμε. Αυτό γίνεται με την εντολή awk και με τη χρήση array. Συγκεκριμένα χρησιμοποιήθηκαν τρία array, ένα για τη βαθμολογία ένα για το γκολ που σκόραρε και ένα για τα γκολ που δέχτηκε κάθε ομάδα. Ως κλειδί χρησιμοποιήθηκε το όνομα της κάθε ομάδας και στα τρία array. Για να μπορούν τα ονόματα να έχουν και κενά χρησιμοποιήθηκε το `-F"\t"` ώστε η έξοδος της πρώτης εντολής awk να χωρίζεται σε στήλες χρησιμοποιώντας το χαρακτήρα “\t” και όχι το κενό. Παράλληλα συγκρίνουμε το μήκος κάθε ομάδας, με σκοπό να βρούμε το μέγιστο και να το χρησιμοποιήσουμε ώστε τα αποτελέσματα να είναι πιο ευδιακριτά.

Στο τέλος εκτελείτε ο βρόχος for και περιεχόμενα των τριών array.

```
awk -F"\t" \
'{
    arr1[$1]+=$2;
    arr2[$1]+=$3;
    arr3[$1]+=$4

    if (length($1) > width)
        width = length($1)
}
END {
    for (key in arr1) printf("\t%-*s\t%3s\t%3s -%3s\n",width, key,
arr1[key], arr2[key], arr3[key])
}'
```

Στη συνέχεια ταξινομούμε την έξοδο σύμφωνα με τη βαθμολογία της κάθε ομάδας. Η βαθμολογία της κάθε ομάδας είναι στη δεύτερη στήλη και το όνομα στην πρώτη στήλη. Αρχικά κάνουμε αριθμητική ταξινόμηση σύμφωνα μόνο με την δεύτερη στήλη και επειδή θέλουμε πρώτη την ομάδα με τη μεγαλύτερη βαθμολογία χρησιμοποιούμε το `-k2,2nr`, σε περίπτωση που δύο ομάδες έχουν την ίδια βαθμολογία, τότε τις ταξινομούμε χρησιμοποιώντας μόνο το όνομα τους οπότε προκύπτει το `k1,1`.

```
| sort -k2,2nr -k1,1 \
```

Τέλος προσθέτουμε τον αύξον αριθμό της κάθε ομάδας με την εντολή `nl`.

```
| nl -n ln -w1 -s "."
```

Με το flag `-n ln` έχουμε αριστερή στοίχιση, με το `-s "."` προσθέτουμε το χαρακτήρα "." μετά τον αριθμό και με το `-w1` ορίζουμε πόσα ψηφία έχει κάθε αριθμός ώστε ο χαρακτήρας "." να βρίσκεται ακριβώς μετά τον αριθμό χωρίς κενά.