ESERCIZIO 1

Si scriva uno script Perl dal nome `totobet.pl` in grado di verificare la correttezza delle giocate di una o più *schedine*. Ogni schedina è composta da un gruppo di scommesse calcistiche. In particolare lo script dovrà funzionare nel seguente modo: una volta immessi come argomenti iniziali i nomi di 2 file (rispettivamente il nome del file delle **scommesse** e il nome del file dei risultati delle **partite** giocate) lo script verificherà i risultati delle schedine giocate; calcolerà il valore del **moltiplicatore** e della **possibile vincita** e infine verificherà se ogni scommessa è stata **vinta** o meno.

Il valore del **moltiplicatore** si ottiene moltiplicando, per ogni schedina, il valore di tutte le quote relative ai match sui quali si sta scommettendo secondo la formula:

```
moltiplicatore = quota_1 * quota_2 * ... * quota_n
```

Il valore della **possibile vincita** è ottenuto moltiplicando l'importo scommesso per il valore del moltiplicatore secondo la formula:

```
possibileVincita = moltiplicatore * importoScommesso
```

Un file delle scommesse è così formato:

```
            #Schedina 1#
Roma-Torino             X       1.5
                ...
Spal-Bologna            X       2.0
    #Importo Scommesso      10
            #Schedina 2#
                ...
                ..
            #Schedina n#
Juventus-Chievo         1       1.5
                ...
 Frosinone-Atalanta       2    2.5
    #Importo Scommesso      5
```

Ogni schedina è composta da più giocate. La prima colonna indica univocamente il match sul quale si sta scommettendo; la seconda colonna mostra il pronostico del risultato dell'incontro ("**1**" → vittoria della squadra in casa, "**2**" → vittoria della squadra ospite, "**X**" → pareggio) mentre la terza e ultima colonna identifica la **quota** della singola giocata.

La squadra di casa è convenzionalmente la prima elencata nel match, mentre la squadra ospite è la seconda squadra. Ad esempio, in Juventus-Chievo, "Juventus" è la squadra di casa e "Chievo" è la squadra ospite.

Il file contenente i risultati finali delle partite/match è così composto:

```
Roma-Torino             3-2
Udinese-Parma           1-2
Inter-Sassuolo          0-0
Frosinone-Atalanta      0-5
            ...
Torino-Udinese          1-0
Sassuolo-Juventus       0-3
Milan-Cagliari          3-0
```

La prima colonna indica le squadre in competizione mentre la seconda ed ultima colonna indica il risultato ottenuto.

Una volta terminata la procedura di verifica, lo script stamperà su un nuovo file dal nome "`check`" i risultati ottenuti come nell'esempio di cui sotto.

NOTA BENE: è possibile scaricare il materiale supplementare a questo link.

Esempio file check:

#Schedina 1#

```
Roma-Torino             X       1.5 --> NO
Udinese-Parma           1       2.0 --> NO
Inter-Sassuolo          X       3.0 --> OK
Frosinone-Atalanta      2       1.5 --> OK
Spal-Bologna            X       2.0 --> OK
          #Importo Scommesso: 10
           #Moltiplicatore: 27
         #Possibile Vincita: 270
              #Vincita: NO
```

#Schedina 2#

```
Juventus-Chievo         1       1.5 --> OK
Genoa-Milan             2       2.0 --> OK
Napoli-Lazio            1       1.5 --> OK
Frosinone-Atalanta      2       2.5 --> OK
          #Importo Scommesso: 5
         #Moltiplicatore: 11.25
        #Possibile Vincita: 56.25
              #Vincita: SI
```

ESERCIZIO 2

Il comando `du` è un comando Unix/Linux standard, utilizzato per controllare le informazioni sull'utilizzo del disco di file e directory su una macchina. Se eseguito con i parametri `-ka path/to/folder` mostra lo spazio occupato da tutti i file e le cartelle (in **Kbyte**) che si trovano all'interno del path specificato. Un esempio di output del `du -ka path/to/folder` è mostrato di seguito.

```
francesco@Francesco:/mnt/c/Users/franc/Desktop$ du -ka SettembreTest/
4       SettembreTest/diskUsage.pl
8       SettembreTest/esempio.txt
0       SettembreTest/main.pdf
0       SettembreTest/output.a
124     SettembreTest/Sotto cartella/file1.pdf
128     SettembreTest/Sotto cartella/file2.pdf
128     SettembreTest/Sotto cartella/file3.pdf
380     SettembreTest/Sotto cartella
16      SettembreTest/test - Copia.docx
12      SettembreTest/test.docx
0       SettembreTest/txt.ciao
420     SettembreTest/
```

La prima colonna esprime lo spazio occupato su disco in Kbyte, mentre la seconda indica il nome del file la cui dimensione si riferisce. Ad esempio, il file `esempio.txt` occupa esattamente 8Kb di memoria su disco.

Si scriva uno script Perl dal nome `diskUsage.pl` capace di filtrare determinate tipologie di file specificate dall'utente e di produrre in output un riassunto dettagliato dello spazio occupato su disco da quelle specifiche tipologie di file.
Lo script riceve in input i seguenti argomenti:

1. **(OBBLIGATORIO)** una lista di possibili formati di file (e.g., **pdf**, **txt**, **docx**, ...) e
2. **(OPZIONALE)** il path ad una cartella di sistema (e.g., **/home/esame/Scrivania/NomeCognomeMatricola**). Se l'utente non inserisce alcun path, di default, lo script effettuerà la ricerca all'interno della cartella corrente.

Lo script dovrà essere eseguito, quindi, con la seguente sintassi:

```
./diskUsage.pl --formats={FORMATS} [path/to/directory]
```

Una volta eseguito, lo script dovrà controllare se, all'interno del path specificato (o nella cartella corrente), sono presenti file con le estensioni specificate tramite il parametro `--formats={FORMATS}`. Quindi, dovrà aggregare per estensione (ovviamente solo quelle specificate) i file presenti e sommare il valore complessivo di Kb occupati da ogni formato di file.

Prima di terminare, lo script stamperà su `STDOUT` l'elenco dei formati dei file filtrati, ciascuno dei quali associato ai rispettivi valori di uso dello spazio sul disco, e ordinati in ordine decrescente di valore. A parità di valore è necessario ordinare anche lessicograficamente per chiave.
Infine, si stamperà su un `FILE` dal nome `du.out` la somma totale dello spazio su disco utilizzato dai file precedentemente filtrati.

ESEMPIO:

1. Contenuto della cartella SettembreTest:

```
SettembreTest/              6824
├── Sotto cartella          6784
│   ├── file1.pdf           5436
│   ├── file2.pdf           728
│   └── file3.pdf           620
├── diskUsage.pl                4
├── esempio.txt             8
├── main.pdf                0
├── output.a                0
├── test - Copia.docx       16
├── test.docx               12
└── txt.ciao                0
```

2. Esecuzione Script:

```
./diskUsage.pl --format=pdf,txt,docx ./SettembreTest
```

3. Azioni effettuate internamente dallo script

Dopo aver eseguito il comando **du -ka ./SettembreTest**, lo script terrà in considerazione solo i file le cui estensioni sono **pdf**, **txt** oppure **docx**. Effettuerà per ognuna di queste estensioni la somma dei Kb occupati su disco e, infine, procederà con le stampe.

4.1. Output (STDOUT):

```
     Estensione: pdf      6784Kb
     Estensione: docx     28Kb
     Estensione: txt      8Kb
```

4.2 Output (du.out):

```
     Totale occupazione disco della cartella
./SettembreTest:    6820 Kb
```