Hey! We are really glad you have reached this point in the recruitment process.

Your main job as a Scripting Software Developer at lastminute.com will be to deliver “Wow!” and hack clever tools to solve data-centered problems. Please complete the following tasks to prove your scripting abilities and problem solving skills. Take your time – you have 72 hours.

You are free to use python, perl, ruby and bash. Submit your answer as a zip file containing two subdirectories (one for each task) containing source codes and additional files if needed.

We are well aware that “recruitment tasks” take time to solve, but use this as an opportunity to shine.

Good luck!

Task 1

The main job of the Group Performance Marketing team is to measure how well online campaigns attract customers. This is usually done through tracking tools that identify individuals, detect which **channel** brought them onto our site and whether they made a purchase or not. Tracking information is then combined with purchase logs to **attribute** value to a particular channel by summing up all values for orders that were made during a visit that began through a given channel.

For the course of this exercise let’s assume our company operates in 4 key markets: United Kingdom (UK), France (FR), Italy (IT) and Spain (ES); sells 5 product types: Hotel, Flights, Holidays, Car rentals and Restaurant dinners. The customer is free to make payments in 3 currencies: GBP, EUR and USD. Furthermore, each market uses separate software and localized date/currency formats.

The “task1” directory contains two CSV files:

1. orders.txt – contains a purchase log with the following columns:
    - OrderDate – an exact date when a purchase was made.
    - Country – a two letter country code.
    - OrderId – a numeric order id.
    - Value – total order value expressed in the currency used to make the payment.
    - Currency – currency code for the Value column.
    - ProductCategory – a product category name indicating what was purchased.
2. visitors.txt – contains data from tracking systems.
   When a customer enters our website, she receives a tracking cookie with a unique user id. Any user can have multiple **“visits”** - an unbroken sequence of activities with the time gap between on-site actions (clicks) not exceeding 30 minutes. If the user is idle for more than 30 minutes, subsequent actions (clicks) are recorded as a next visit for the same user id. For each visit, we also track the channel which brought the customer to our site. Possible channels include:
    - **Direct** – the user enters our site by typing our address or has it bookmarked (technicaly, there is no referrer in his http request).

- **SEO** (Search Engine Optimization) – organic links in Google, Bing or Yahoo. We define two **subchannels**: SEO_Brand and SEO_NonBrand where:
    - **SEO_Brand** means that the customer **used** a keyword with a variation of our brand name "lastminute.com" (including misspellings or skipping the ".com" bit)
    - **SEO_NonBrand** means that the customer used a generic search term that **does not contain** a variation of our brand name "lastminute.com".
- **PPC** (Pay-per-Click) – paid advertising links (like Google Adwords). Again, we define two **subchannels**: PPC_Brand and PPC_NonBrand where:
    - **PPC_Brand** means that the customer **used** a keyword with a variation of our brand name "lastminute.com" (like for SEO_Brand)
    - **PPC_NonBrand** means that the customer used a generic search term that **does not contain** a variation of our brand name "lastminute.com"

The visitors.txt file contains following columns:

- VisitDate – date when the visit has started (date of first click in that visit)
- UserID – a unique user identifier that was contained in the cookie dropped by our website.
- VisitID – a sequence number indicating the visit.
- Source – an encoded channel/campaign indicator. Depending on the channel it contains the following values:
    - **for Direct** – empty.
    - **for SEO** – a string following the pattern "seo :: [search engine] :: [keywords]". The **SEO_Brand** and **SEO_NonBrand** channels need to be distinguished by analyzing the contents of the [keywords] part.
    - **for PPC** - a string following the pattern "ppc_[country]_[keywords]". The **PPC_Brand** and **PPC_NonBrand** channels need to be distinguished by analyzing the contents of the [keywords] part.
- OrderID – a numeric order id. Only present if a purchase was made during the visit.

Your task is to create a script that is launched with two command line parameters: a date in the YYYY-MM-DD format and a 3 letter currency code (GBP/EUR/USD). The script should return:

1. The number of purchases made during the supplied date and total purchase value (sum of Value) expressed in the provided currency for all combinations of country, product category and channels/subchannels.
2. The number of unique user ids and the total number of visits that were recorded for the supplied date broken down by channel.

Example call:

```
python task1_myscript.py 2010-03-02 USD
```

Should return something like (the output format is flexible):

```
Orders:
UK – Hotels – Direct = 32 orders, 31233 USD
UK – Hotels – SEO_Brand = 1 orders, 423 USD
UK – Hotels – SEO_NonBrand = 212 orders, 54323444 USD
…

Visits:
Direct – 32311 Users, 54442 Visits
SEO_Brand – 1234 Users, 4422 Visits
SEO_NonBrand – 45522 Users, 98384 Visits
PPC_Brand – 2 Users, 2 Visits
PPC_NonBrand – 12454 Users, 44222 Visits
```

When performing currency conversions, please use FX rates downloaded automatically from the web (any website will be ok) for the date when the script is being run.

Task 2

Please write a script that takes a music band name as a command line parameter and returns its musical genre and discography (note that this information must be downloaded in real-time from the internet – you cannot use a static database to store it).

Example call:

```
python task2_myscript.py Queen
```

Should return something like (output format is flexible):

```
Queen
Genre: Rock
…

* A Kind of Magic
* The Miracle
* Innuendo
* Made in Heaven
```