Pseudo Code Assignment

The handout can be found here.

Preface

To keep it simple we will take a look at a *single thread solution*. Then we will try to see what can be done to optimize the execution using multiple threads. For brevity, we will just list some possibilities.

Single thread

Given a growing list of mails (listOfMails):

```
 recruitment = 0
spam = 0
sales = 0
reception = 0

intervalInMin = INPUT("Interval in minutes?")

WHILE liveSorting
    get listOfMails
    listOfMails -> stream
        if listOfMails != null
            FOREACH mail of listOfMails
                IF contains "CV" (~ title / body / attachment ???)
                    forward to "recruitment@parkshark.com"
                    recruitment++
                    delete mail (* in mailbox)
                ELSE IF contains "Promo" OR contains "advertising" (~)
                    forward to "spam@parkshark.com"
                    spam++
                    delete mail (*)
                ELSE IF contains "sales" (~)
                    forward to "sales@parkshark.com"
                    sales++
                    delete mail (*)
                ELSE
                    // devide work among 2 people (can use a modulus and switch case for more people)
                    if reception is even
                        forward to "reception@parkshark.com"
                        reception++
                        delete mail (*)
                    else
                        forward to "jobstudent@parkshark.com"
                        reception++
                        delete mail (*)
        else
            liveSorting = false     // Stop program when there are no more new mails
            // Execute some extra code before shut down
    print # of mails
    sleep (intervalInMin toMillis)

getListOfMails
    listOfMails = (List) GET someApi
    return listOfMails

deleteMail
    DELETE someApi

print # of mails
    total = recruitment + spam + sales + reception
    print: total, recruitment, spam, sales, reception
    // Can print to csv file to save data
    // This csv can be consumed by some other code and do more calc by adding collumns
    // For example (current - previous) for data on 1 loop
```

Multiple threads

Things we could try to optimize: - Use a **parallelStream** instead - List of mails has to be big enough to improve execution speed - Introduce a **thread pool** and pass what comes after FOREACH to executors - Have **different threads each take a list**, then filter for containing a string before FOREACH - Some lists could be made to have more / less importance

Questions for 'Park Shark'

- Where does the list of mails come from?
- What is the average amount of mails received in the mailbox?
- What are the preferences regarding the interval to check the mailbox?

    - What should be the frequency?
    - Should it be changeable?
- Where will this program be running?
    - Should this program have an own interface?
    - Or will it be integrated and run with command line arguments?
- ...

-- Steven D'Hondt