Take Home Assignment

Conference Scheduler

Description

Singapore is hosting the PIOSS © (Proprietary Internal Open Source Software, GPL Strictly Not Welcome) Asia summit 2018. Your proposal to algorithmic-ally generate the entire conference schedule has been accepted. Congratulations!

You will be given a raw list of selected talks. Talks are broadly classified into the below mentioned categories -

- Panel Discussion - Panel discussions with Q&A. 60 Minute
- Lightning Talk - 10 Slides. 10 Minutes
- Regular - 30 Minutes
- Keynote - Themed talks, typically by one of the sponsors. 30 Minutes
- Workshop - Hands-on sessions or Show and Tell workshops. 60 Minutes
- Closing - Daily closing speech at the end of all other sessions. 30 Minutes

Scheduling Constraints are as follows -

- The conference runs for 2 Days - 29th and 30th of June, 2018
- Each day starts with a Keynote speech from one of the sponsors. The Keynote starts at 09:00, daily.
- The conference ends daily at 17:30
    - Closing speech commences at 17:00
- Lunch - The conference will pause for a 60 minute Lunch & Networking session between 12:30 - 13:30 daily.
- Tea - The conference will pause for a 15 minute Tea break.
    - Tea break must start at 15:00
- All talks must be scheduled. The conference organizers *may* choose to run two parallel tracks during a conference day to accommodate all talks.

PROBLEM STATEMENT

In any language of your choice, take the list of shortlisted talks and generate a conference schedule that satisfies the scheduling constraints listed above.

- Please see the attachment titled talks.json that contains the list of all accepted talks that need to be scheduled for the conference.
- Please supply a README describing how we can run the program from source and a high level design document. Please mention any assumptions that you've made in the README.
- The program must print or write the schedule in a simple, readable format.
    - *One* possible solution for the assignment is shown below. Do note that this is just one possible solution and your program is not expected to produce the exact same schedule.

Sample Output

```
> <execute your program>
Day 1 Track 1:
09:00 Design secure CD pipelines using YAN CI KEYNOTE
09:30 What exactly is meltdown? WORKSHOP
10:30 Security Policies in the enterprise REGULAR_TALK
11:00 What is a container orchestration? REGULAR_TALK
11:30 Continuous Deployment in the era of microservices WORKSHOP
12:30 LUNCH
13:30 How to make a bad cocktail with a single voice command
WORKSHOP
14:30 Rise of Cloud REGULAR_TALK
15:00 TEA
15:15 How to Solder LIGHTNING
15:25 Secure Your Hypervisor WORKSHOP
16:25 Blue Green Deploy LIGHTNING
16:35 10 command-line shortcuts you must know LIGHTNING
16:45 Blockchain in 10 minutes LIGHTNING
17:00 Benefits of using ETraining for your employees CLOSING

Day 2 Track 1:
09:00 Chatbot assistant to assist your helpdesk KEYNOTE
09:30 Replicate Databases across AZs WORKSHOP
10:30 Writing a simple classifier WORKSHOP
11:30 Make Data Centers Secure Again REGULAR_TALK
12:00 Defensive programming REGULAR_TALK
12:30 LUNCH
13:30 What Do CDNs really do? REGULAR_TALK
14:00 Testing in the Age of MVP REGULAR_TALK
14:30 What is conversational Web REGULAR_TALK
15:00 TEA
15:15 What is a container? REGULAR_TALK
15:45 How does WGET work LIGHTNING
15:55 What is Map-Reduce REGULAR_TALK
16:25 Blockchain in the enterprise REGULAR_TALK
17:00 Customer feedback for ECloud CLOSING
```

Submission – Please supply the zipped source code only. We will not accept binaries. Create an archive containing the source code and supporting documentation and send it back to us via an e-mail.

NOTES

- Implement a working solution using any language of your choice. Create the necessary data structures to represent the domain model.
- The solution is expected to be written and shared in a manner that is easy to run and extend.
- Use any build tool of your choice. Do note that while you can use common, generic libraries to build your solution, please refrain from using highly specific libraries.
- Please work on the problem individually.
- Writing tests is preferred over extensive documentation.
- Do not assume that our evaluation/test environment has specific libraries and/or tools available/installed. Please state all the dependencies that your solution requires in order to run.
- Your solution will be judged for correctness, design, readability and it must be extensible.
  - While this is a simple problem, we are looking for your solution to be well-designed and not just the algorithm that produces a solution.
- If you are using version control, please include the version history in the archive (e.g. Use the `git bundle` command)