

# SEAT:CODE

Hello

This is our code test. Here you will find the description of the problem, what is required from you, and an explanation of any input and output to be produced by the code.

Hopefully the exercise will be intellectually interesting. In your answer we look for clues of how you cope with (perhaps) a new business domain, how well do you follow requirements, how good is the design of the solution, and, very importantly, how do you balance the time that you have to do it versus the quality of the final output. We understand that you cannot spend days solving it. Do your best and spend the time you can on it, and be prepared to defend your choices during the follow up interview.

Good luck. Any questions let us know. One final thing: please make sure that the code compiles and runs, and include any instruction on how to build and run it. No matter how incredible your solution, if it does not run, it does not count.

Thanks

The **SEAT:CODE** team

# SEAT:CODE

## Problem description

The city of London has implemented a robot that moves through a predetermined route every morning, collecting pollution data at fixed intervals, and reporting them.

The goal of this exercise is to implement code that would control such robot, within the following guidelines:

- Every morning an operator feeds the robot with the polyline (taken from Google) with the route of the robot (see polyline below).
- The robot will move along the points of the polyline.
- Acceptable speed for the robot is between 1 and 3 meters per second
- Every 100 meters, the robot “reads” the level of PM2.5 particles in the location.
- Values of PM2.5 will be in the following range:
  - 0 to 50: Good
  - 51 - 100: Moderate
  - 101 - 150: USG (Unhealthy to Sensitive Groups)
  - > 150: Unhealthy
  - Of course this can be randomly generated at each stop
- Every 15 minutes the robot reports the average of all new readings since the last one and reports according to the grouping above. The output should be a json like:

```
{“timestamp”: 1528106219,“location”: {“lat”: 51.23241, “lng”: -0.1223},“level”: “USG”,“source”: “robot”}
```

## Inputs / Outputs

The only input is the polyline, given by an operator as a startup parameter.  
The output is the periodic (15 minutes) report.

# SEAT:CODE

## Bonus section

We give you 2 ideas on how to make things more interesting. Feel free to pick 0, 1, or 2 of them, or even suggest another one.

### Bonus 1

There are 2 monitoring stations along this predefined route:

1. Buckingham Palace: 51.501299, -0.141935
2. Temple Station: 51.510852, -0.114165

As the robot enters a radius of 100 meters of one of these monitoring stations, they should output a message in the same format as above, ensuring that the robot is moving safe and soundly.

```
{"timestamp": 1528106219, "location": {"lat": 51.23241, "lng": -0.1223}, "level": "USG", "source": "station_name"}
```

### Bonus 2

Implement an API for an operator to interact with the robot, with the following "commands"

1. start/stop
2. re-route (a new polyline would be sent)
3. report: to generate an ad-hoc report with the same json format as above.

## Polyline

For the sake of this exercise, this is the polyline to use. It covers a route between West London and the City.

```
mpjyHx`i@VjAVKnAh@BHHX@LZR@Bj@Ml@WWc@]w@bAyAfBmCb@o@pLeQfCsDVa@@@ODQR}AJ{A?{BGU  
AD_@FKb@MTUX]Le@^kBVcAVo@Ta@|EaFh@m@FWaA{DCo@q@mCm@cC{A_GWeA}@sGSeAcA_EOSMa  
@}A_GsAwFkAiEoAaFaBoEGo@]_AIWW{AQyAUyBQqAI_BfEd@aHZcDIaYJLaBPqDDeD?mBEiA}@F]yKWqGSKl  
CmCleZluZi@_Sw@{WgAoXS{DOcAWq@KQGIFQDGn@Y`@MJEFiHyAVQVOJGHgFRJBCCSKBcAKoACyA?m^y  
VJmLJ{FGGWq@e@eBle@Ei@?q@Bk@Hs@Le@Rk@gCulkJcZsDwLd@g@Oe@o@mB{BgHQYq@qBQYOMSM  
GBUBGCYc@E_@H]DWJST?JFFHBDNBJ?LED?LBv@WfAc@@EDGNK|@e@hAa@`Bk@b@OEK@Go@leACoA@  
a@PyB`@yDDc@e@K{Bi@oA_@w@]m@_@]QkBoAwC{BmAeAo@s@uAoB_AaBmAwCa@mAo@iCgAwFg@iD  
q@}G[uEU_GBUP@clCmA?el?qCB{FBkCl}BOyCMiAGcAC{AN{YFqD^}FR}CNU@JcAHu@b@_E`@}DVsb^mBTsAQ  
KkCmAg@[YQOIovAi@[m@e@s@g@GKCKAEJIn@g@GYGlc@ScBoAf@{A`@uAlBfAG`@
```