# Overview

The goal of this is to write a simple command-line program that will print out a list of food trucks, given a source of food truck data from the San Francisco government’s API. Along with your code, readme, and installation instructions, we will also ask you to provide a brief write up discussing your thought process.

# The Task

## Data

The San Francisco government’s website has a public data source of food trucks

(https://data.sfgov.org/Economy-and-Community/Mobile-Food-Schedule/jjew-r69b). The data can be accessed in a number of forms, including JSON, CSV, and XML. How you access the data is up to you, but you can find some useful information about making an API request to this data source here (<https://dev.socrata.com/foundry/data.sfgov.org/jjew-r69b> [)](https://dev.socrata.com/foundry/data.sfgov.org/jjew-r69b) .

## The Problem

Write a command line program that prints out a list of food trucks that are open at the current date and current time, when the program is being run. So if I run the program at noon on a Friday, I should see a list of all the food trucks that are open then.

## Criteria

We will primarily evaluate programs on code quality and output correctness.

For quality, we expect code to be easy to read and maintain, performant, testable and reliable. You should submit code that you are proud to have written. However do not include tests in your submission.

Please display results in pages of 10 trucks. That is: if there are more than 10 food trucks open, the program should display the first 10, then wait for input from the user before displaying the next 10 (or fewer if there are fewer than 10 remaining), and so on until there are no more food trucks to display. Display the name and address of the trucks and sort the output alphabetically by name.

## Example

$ java -cp foo:bar FoodTruckFinder

NAME ADDRESS

Mang Hang Catering 1 Thomas More

Way Steve’s Mobile Deli 145 King Street

We expect you to write the solution in Java

## Things we are looking for

Aside from the guidelines listed in the problem and the criteria, the grader(s) of your program will also be assessing the following:

* Food Trucks are sorted alphabetically
* There are clear comments or self-documenting code
* Clear naming conventions
* Concise and readable code
* Code is broken up into multiple classes and methods based on responsibility
* Clean output from program
* Error cases addressed and properly handled
* Code is testable
* The Socrata API is properly used
* Business Logic is not included in the DTOs
* Think about a solution that can be scaled for bigger application

You are free to use any external libraries if your design calls for it.

# Submitting your work

Please email us a zipped folder containing your work. Your submission should include:

1. Your code.
2. A README file that contains instructions on how to do the following on Linux or Mac OSX (if you need an exception to this, let us know):
   1. ○ install dependencies.
   2. ○ build your program.
   3. ○ run your program, with example commands on how to run it if necessary.

Your submission should not include any binary files or any executable files other than your code, such as .jar files.

(**Note** Oracle email server doesn’t allow **.zip** extension files, so we suggest you upload the zip file to a shared cloud storage and provide us the secure download link)

# Submission Notes

We recommend you complete and return this project within 24 hours.

We don’t expect your write-up to take more than thirty minutes to complete. Therefore, keep your response high-level and not more than two paragraphs. Do spend time polishing and packaging up your submission so it is easy to install, run, and review.

Example stub

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

public class FoodTruckFinder {

public static void main(String[] args) {

try {

StringBuilder result = new StringBuilder();

URL url = new URL("http://data.sfgov.org/resource/bbb8-hzi6.xml");

HttpURLConnection conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("GET");

BufferedReader rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));

String line;

while ((line = rd.readLine()) != null) {

result.append(line);

}

rd.close();

System.out.println(result.toString());

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

}