This exam is **open-book**. You are allowed to consult the `pyspark` documentation, the course materials, and any calculator. However: Every piece of text, and every figure that you write here must be **your own work**, so must not be pasted from anywhere: **you must not paste anything from the Web, nor from your own assignments or project materials, since those were group work!** Any similarity between answers will lead to an investigation of the exams in question by the Examination Board.

Please close all messaging software during the exam.

All your answers go into the **Google Doc** created for each of you; there, you can write plain text, draw tables, or add pictures (for example taken with your phone of something you drew on paper). Show that you know your stuff! Each question is marked with a number of percentage points (for example, 10 %), and they add up to 90 % (10 % is given by default). All questions require you to explain your answer. The explanation must be correct and complete in order to get all the points for that question.

Question 1 *(15 %)*

Say that your company has a large computing cluster with enough disk space and memory to store big data. You are given three sources of big data below. For each, give your opinion:

How would you store that data on the cluster? In what format? Why is your solution good?

Give as many details as you know. You need to store all the data safely.

(a) Weather data from all over the globe. Each data record contains at least the following information: `timestamp, geographical coordinates, wind speed, amount of rain` in the last 5 minutes. (The data is processed every hour, and weather statistics will be made for each country.)

(b) All the sales made by a company. Say, the company makes consumer electronics, and sells them in thousands of shops in a large country. Each data point is a sale, and contains: the type of the item, the timestamp for the sale, and the sale price. (This data will be processed every day: for each type of item, the company will count how many copies were sold on that day.)

(c) All the pictures taken by satellites of the earth surface every day. Each picture is between 20 and 120 MB in size, and is annotated with the time when it was taken, and the geographical coordinates of the top-left corner of the picture.

Question 2 *(10 %)*

Now you have a computing cluster running the Google File System. The cluster has 1000 chunk servers plus one master server. Each of the servers has a disk space of 500 GB, has 10 GB of RAM available, and a chunk size of 128 MB, with the standard replication factor of 3. About 10 KB of metadata needs to be stored per chunk handler. You want to store a very big file on this cluster.

What is the maximum file size that is possible to store on this cluster, and why?

Question 3 *(14 %)*

In a Spark program for processing big data:

What type (actions or transformations) are the following Spark methods over an RDD?

- `mapValues(f)`,
- `coalesce(numPartitions)`,
- `isEmpty()`,
- `getNumPartitions()`,
- `keys()`,
- `top(num)`,
- `sampleByKey(withReplacement, fraction)`.

(You can find them in the PySpark documentation:

`http://spark.apache.org/docs/2.2.0/api/python/pyspark.html#pyspark.RDD`.)

Explain why, for each. Because this is an open-book exam, your own explanation (and how clear it is) matters much more than choosing the correct answer.

Question 4 *(16 %)*

You know two software frameworks which can process *streaming* big data. For both these frameworks:

Explain, in your own words, what type of *code or configuration* will cause data to be sent (transmitted) from a worker node to another worker node in the cluster. Said differently: what method calls or topologies cause a lot of *network traffic* in your computing cluster, if the cluster runs one of these streaming frameworks, and you programmed and configured the application running there?

(Network traffic can slow down a processing job, because network communication is slower than processing data locally.)

Give some examples of streaming methods or configurations which may cause network traffic. Compare the two systems if you can: are there any similarities between them, in this respect?

Question 5 *(15 %)*

Some cloud service providers, such as Google, are known for collecting a lot of user data, in order to tailor (select the most relevant) advertisements or search results for the user. How would you design a Bigtable database for such user data?

In more details, say that the company collects the following types of user data, at least for the last 12 months:

- the search keywords that the user wrote in their search engine,
- the user's locations, as reported by the user's mobile phone when any location services (such as GPS) were enabled,
- the websites that the user visited using the company's browser.

Sketch the Bigtable schema and explain what is stored in every cell.

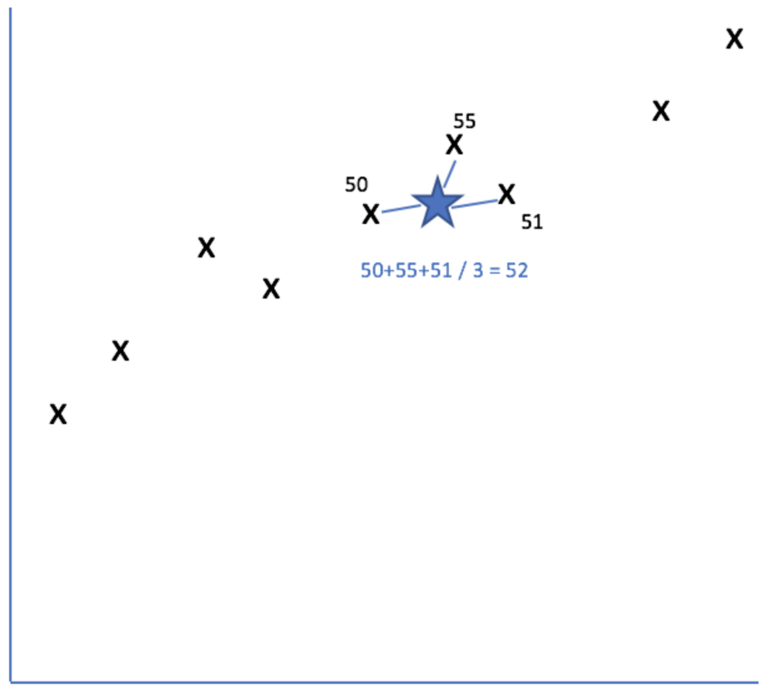
There are more than one good solutions possible!

Question 6 *(20 %)*

Sketch a Spark program which does the following: learns how to regress (in other words, predict the value of) new data points, given a very large dataset of points for which you already know the correct value. This is a simple form of regression over big data.

More technically put: you have a large dataset of N two-dimensional data points of the form `x, y, value`. For each data point, `x` and `y` are two features of that point. Some possible examples of what these data points might mean in practice: the latitude and longitude of a place, or the air temperature and humidity at a given time of any day in any year. `value` is the true value of an important aspect of that data point, for example the *annual amount of precipitation* at that place (say, 200 mm), or the *latitude* of the place where that weather data was taken (say, 45.35 degrees). This dataset is stored for you in a big file in plain text (.csv or .json, as you like), one point per line.

Now, you also get p new data points for which you know `x` and `y`, but you must find out the most likely `value`. The algorithm you should implement is simple (see picture below): the value of the new point (the blue star) is the *mean* value among the values of the k *closest points* in terms of Euclidean distance. k is a relatively small number (say, below 1000) and is given to you.



The only difficulty here is that N is very large. Consider first the case in which $p = 1$, to make it easier. Then, find a more general solution for a larger p (discuss what would change as you increase p; where you stop with increasing p is up to you).

(Points will be given here only if you have a solution that is really feasible and implementable in Spark. You'll also get some points for designing and explaining a feasible solution, even if you don't provide much code.)