This exam is **open-book**. Study materials and your own notes are allowed, but **no code** or any other group work. Pocket calculators are allowed.
No other electronics allowed except for the Chromebook!

Write your **name and student number** on the first page (below this box), and in the header of each sheet of paper.

Give your answers **on this paper** in the space provided. Design your answer on **scratch paper before** you start writing here, or you may run out of space! The explanation must be **correct and complete** in order to get all the points for that question. Partial points are possible. The points add up to 90 % (10 % is given for free).

Name and student number: ____________________

Question 1 *(10 %)*

Say 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 1 TB, has 12 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 2 *(16 %)*

For each of the following Spark methods over an RDD, say **which type** they are, namely:

- **actions or transformations**, and
- with **narrow or wide dependencies**.

Explain why, for each. Your explanation matters more than stating the correct answer.

1. `flatMap(f, preservesPartitioning=False)`,
2. `map(f, preservesPartitioning=True)`,
3. `filter(f)`,
4. `coalesce(numPartitions)`,
5. `max()`,
6. `getNumPartitions()`,
7. `top(num)`,
8. `collect()`.

Question 3 *(16 %)*

Take the following Spark program:

```
rdd = sc.wholeTextFiles("/data/Gutenberg-EBooks")
book_index = rdd \
    .map(lambda (file, contents): (file, set(contents.split()))) \
    .flatMap(lambda (file, words): [(w, set([file])) for w in words]) \
    .reduceByKey(lambda files1, files2: files1 | files2) \
    .filter(lambda (word, files): len(files) == 100) \
    .map(lambda (word, files): word) \
    .collect()
```

For each line which creates a new RDD, state the **data type** of the records in that RDD (for example, integer, string). For each line which doesn't create a new RDD but returns a result, again state the type of that result.

Question 4 *(12 %)*

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

How would you store that data on the cluster, using the storage framework(s) you know for big data? Why is your solution good enough?

You need to store all the data safely, and make it possible for it to be processed by the company.

(a) All the sales made by the company. Say, the company sells products in thousands of shops. 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 once a day: for each type of item, the company will count how many copies were sold on that day.)

(b) 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 5 *(10 %)*

Some cloud service providers, such as Google, are known for collecting a lot of user data, in order to tailor advertisements or search results for the user. Sketch a Bigtable database for such user data.

Say that the company collects the following types of user data:

- the search keywords that the user wrote in their search engine,
- the pages that the user visited using the company's browser.

Sketch the Bigtable schema and explain what is stored in every cell. Many good solutions are possible!

Question 6 *(10 %)*

With what you know about processing big streaming data, answer the following, in your own words:

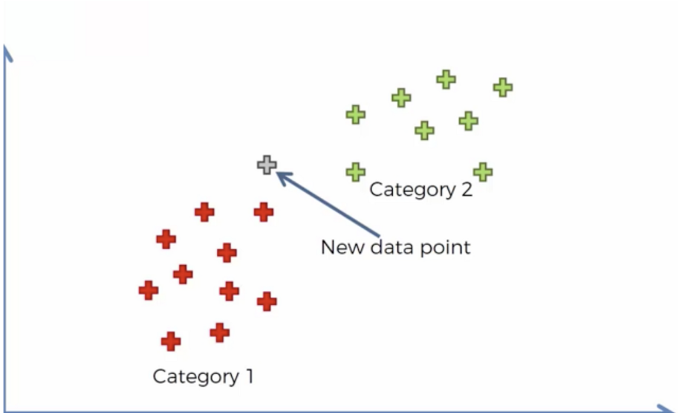
If you already have Spark Streaming available on a computing cluster, why would you also install and use Apache Kafka? In other words, list which functionalities Kafka has, that Spark Streaming does not have.

Question 7 *(16 %)*

Sketch a Spark program which **classifies** data points, as follows.

You have a big dataset of N two-dimensional data points of the form `x, y, class`. For each data point, `x` and `y` are two features of that point. These data points might be: the weight and height of a human being, or the air temperature and humidity at a given time of any day in any year. `class` is the true class (or "category") of that point, for example the *gender* of the person, or the *season* of the year (summer, etc.) when that data was taken. You can have any number of classes in the dataset (say, 3 gender identifiers, or 4 seasons). The dataset is stored in a big file in plain text (say, .csv), one point per line.

You also get p new data points for which you know `x` and `y`, but you must find out the most likely `class`. The algorithm you should implement is simple (see picture below): the class of the new point is the *majority class* among the k *closest points* in terms of Euclidean distance. (You can also use other distances if you have ideas.) k is a relatively small number (say, $k = 100$) which is given to you. You can choose how to resolve ties (if a new data point has 50 neighbours of one class, and 50 of another class).



Consider first the case in which $p = 1$, to make it easier. Then, try to find a more general solution for a larger p. (Points will be given here if you have a solution that is really feasible on big data. You'll also get some points for explaining a solution, even if you don't provide much Spark code.)