My task is to read "The twelve factor app" and answer the following.

- Research Internet sources discussing the recommendations.
- Based on your findings, try to re-order(re-number) the know factors accordingly
- be ready to argument these criteria in class

Research Internet sources discussing the recommendations.

Dependencies

This is a blog, where some cons against dependencies are found:

- `https://techblog.bozho.net/comments-on-the-twelve-factor-app/`

The idea mentioned here, is to avoid big app package, by have the app rely on something. I do think this will also affect the fast startup time covered by the Disposability step. You cannot start up a new instance of the app on a new server, if this server has to have some form of tools on it.

all-round

This is a blog, where the author takes all the step and grade the importance on a scale of 1-5

- `https://gaddings.io/comments-on-the-twelve-factor-app/`

Re-order the know factors

To be honest, i can follow all the steps, and see them as equally important. I think the order is to be seem more of the order you would do all the task, to go from nothing to a fully running app. I do believe of the step can be skipped if not needed, ie if the app does not need a log, this step can be skipped.

What is The twelve-factor app?

The twelve-factor app, is a methodology for building Software-As-A-Service programs.

There are twelve step that has to be followed.

The step are:

Codebase

The code base is the code of the app. There are a few rules about what makes the code base:

- The code base refere to a single repository.
- An app is based on a codebase, the app does not share this code base with any other app.
- If code is to be used between different apps, it has to be taken out of the app, and handled as a libery, and imported through a package manager.
- There can be different versions of the codebase running at the same time, ie in the developer environment the code base version would be hgiher, the that in the production environment.

Dependencies

The step explains about the denpendencies in the app.
There are a few ground rules to follow:

- All dependencies has to be declared.
- The app cannot depend on any system tools (ie curl in linux)

Config

This is about the config the app needs. There are a few ground rules to be followed:

- No config of the app can be in the app's base code. The base code has to be the same everywhere, the config can be different from site to site.

Backing services

This describes the backing services or the resources the app needs. This can be a database or a public api the app needs to run. There are a few ground rules this step has:

- The config of connection to a resource has to be in the config, not in the code base.
- A resource has to be able swaped out if it fails, usely by changing the config to use another resource, without changing the code base.

Build, release,run

This is about the 3 stages of the app. The 3 stages are as follows:

- Build, this is where the code base is compiled, and dependencies are added.
- Release, this is where the build and config are combined.
- Run, this is where the release is installed on a system, and can be executed.

Processes

Explains about the execution environment for the app. There are a few rules to follow here:

- The process has to be stateless and share nothing.
- Data persisted, has to be dealt with in that transaction, because you dont know, if the same server will respond next time.

Port binding

This is about exporting services via a port binding.
There is an important rule to follow here:

- The app has to be self-contained, it cannot rely on anything, so if the app needs to make a webpage, it has to import some form of webserver library.

Concurrency

This is about scaling out via the process model.
there is an rule here:

- The app should not be handeling things the OS should be handeling, ei. it should not be handeling the pid in linux.

Disposability

This is about fast start up and graceful shutdowns.
There are a few rules:

- The app should aim for a fast startup time, so scaling will be fast if(when) needed.
- The app should shutdown gracefully, meaning, stop taking new requests, and finish the ones it are working on.

Dev/prod parity

This is about the difference in the development environment and the production environment.

There are a few guidelines to follow:

- Keep the time between deploys to a minimal.
- Code author and and deployer should be the same.
- Keep the difference between dev and prod to a minimal.

Logs

This is about how the app should treat logs.

The rule is:

- The app should send its log to stdout, and let something else handle the logging.

Admin processes

This is about the ability to run a one-off process.

There is a guideline:

- One-off admin process should be run in an identical environment.