Task 1:

Part1:
I have been running the ipython notebook from a VM in the smog cloud.

Part2:
See the appended pdf file. It was not possible to set the password and username for smog through sourcing the ACC file. Because of this problem I have stored the login info in clear text in the source code and I'm using python to set the environmental variables. I changed the password in the file after running to xxx so it's not visible in the pdf.

Task 2:

Question 1:
What does it mean that the object store has a global, flat namespace? What is the practical consequence for you when using it?

It mean that each object have a unique name that works as a identifier. This identifier makes it possible to find the object in the namespace without looking deep into any hiarchy structure.

Question 2:
The above excersise showed a low-level way of 'contextualization' using user data 'cloud-init'. Do some reserach online and discuss alternative tools and techniques for contextualization of your VMs. Discuss the difference between instance meta-data and user-data. Some links to get you started:

http://docs.openstack.org/user-guide/cli_provide_user_data_to_instances.html
http://cernvm.cern.ch/portal/contextualisation https://cloudinit.readthedocs.org/en/latest/

Aim for the equivalent of ~1/2 page of an A4 paper, 12pt font, 2cm margins.

The main difference between instance metadata and user-data is that the metadata is not in the control of the user. The meta data is provided by the cloud infrastructure and contains things that's necessary for the cloud to handle the Virtual machines and other things around that. The user-data is provided by the user and in the user's control. The

user data can be used to personalize or specialize the virtual machine, for example by installing necessary software on initialisation of the virtual machine.
Because of this difference the way how the data is accessed differs as well. The meta data is provided by the cloud typically through http access while the user data is provided via the user at initialisation.
The user data is normally provided as a batch script but there might be alternative ways of doing this. As the website https://cloudinit.readthedocs.org/en/latest/ suggest there might be different methods to run script at startup and in different phases of startup. For example their cloud boothook that runs in the earliest phase possible.
More special versions of contextualisation that's mentioned in readthedocs is the gzip compressed content that works like including a compressed file with gzip. In contextualization the file is opened and run as it wasn't compressed from the start. Reducing the size of the initialization. Other methods are possible to imagine, for example providing a http link for code or files to use for the initialization, much like the meta data already present in the cloud. This could make it possible for the user to change many instances initialization process at one place, something that could ease management for the user.

Part 3:
Explain your findings in part 3. Are VMs slower than the physical machine? If yes, explain the reason. Are there alternative to VMs? How do you compare them with VMs? Keep the answer fairly short, limit it to a few sentences, max ¼ of a page.

The VM performed the calculation in 35.009 seconds (real) which is slightly slower than the physical machine (about 8% difference). It could be because the VM I'm testing with doesn't have access to the same amount of cores but I suspect the Linux command to not being concurrent.
I suspect the difference might come from overhead between the virtual hardware and the physical hardware. Another possibility might be that the IO load differs and that the CPU in my VM was occupied with other jobs. The hypervisor might be involved in this as well.