Task 1: Software Product Line

a) A software product line is a system intended to produces customizable software products with the help of modularizing their features and choosing the appropriate features for the build prior to deployment.

b) When the software can be tailor-made for the needs of the customer. That means no one-size-fits-all product. A product line software has customized features and no undesired features that overloaded the user interface.

c) Office365 is not a software product line. It provides always the full package of features, even tough they are not necessary ever used by the user. The design decision to do so is most likely that it is intended to run on workstations and not on embedded systems, hence the 'waste' of recourses is not as crucial.

d) Energy constraint or recourse constrained systems, including embedded systems, mobile devices and smart-sensors.

e) BusyBox: Regarding the description I would definitely consider it a software product line. There is a base functionality which can be customized during compile time to exclude or include various features. "It is also extremely modular ... " (https://busybox.net/about.html; opened 13.04.2019)

Gimp: Is no software product line. All features are always included in every distribution. Especially for beginners the amount of features can be overwhelming.

Android: Since every system-image needs to be downloaded by itself (see Android-Studio) I would consider that at least the build is individual for one or a few devices. The Kernel it is based on is definitely a software product line. So Android should be a software product line.

Task 2: Chat Product Line

a)

Task 3: Software Product Line Development

a) Both views look at the development part. During the domain implementation the intention is to implement various reusable artifacts that are features that were previously found during the domain analysis. The product deviation on the other hand deploys the software by building software in the best case automatically and with the existing artifacts. Based on the requirement analysis of a specific customer the desired software product is build, sometimes implementing additional features that are not included in the domain implementation.

b)The proactive approach implements a product line from scratch while the extractive approach takes a collection of existing products and refactors them incrementally to end up with a product line. The proactive approach is an idealistic and academic process while the extractive approach is lightweight, low-risk and therefore more commonly used in practice. (Apel et al. 2013)