**Homework**

**Beet Seed**

**1-** If I need to test a coffee mug as shown on the link : <https://www.amazon.com/YETI-Rambler-Insulated-Stainless-MagSlider/dp/B09RLQCSKB/ref=sr_1_1_sspa?keywords=Thermos%2BMug&qid=1706280887&sr=8-1-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9hdGY&th=1> ;

* First i would start my testing with static testing so like checking the user manual to find out if there any defects in the user manuel.
* As a second step i would proceed with non-functional testing such as checking ; if the color of the mug is really red as specified in the requirements, if the trademark/logo is located exactly at the same place located as shown in the requirements and also would check the size/height/width of the characters as per the requirements. At this stage i would also test the shape/design of the mug if it aligns with the specified requirements.
* Later on i would start functional testings on the mug ; for ex by filling it with water ( cold & hot subsequently ) at different temperatures to test if maintains the temperature at indicated degrees as per the requirements. I would also test the weight and the capacity of it if they match with the requirements.
* As validation ; i would also test and assess the mug from user/consumer perspective ; so i would test if i can comfortably hold it with 1 hand, also if the coldness or warmness of the mug depending on the water in it has any negative impact on me or it is isolated well enough.

**2-** Validation is checking whether the software meets the user`s expectations. In other words here the tester tests the software from user perspective (user acceptance testing ) to ensure if it satisfies the end user`s business needs. For ex : a button color on a mobile device could be specified as `` Pink`` in the requirements, but through validation the tester needs to check if this color satisfies the user`s expectations.

Verification is checking whether a software meets the specified requirements through for ex. unit testing, so basically verification checks if the software meets the written criterias/specifications provided by the customer before testing starts/or during the testing if any changes were made.

3- In some countries like Cyprus, UK etc. traffic keeps left so cars have the steering wheel on the right. So as a tourist if i would like to visit those countries and rent a car to drive ; i believe this could be an example of successful verification ( as long as the steering wheels were located as per the requirements of the country ) however it does not meet my personal expectations as a tourist who got used to have steering wheel on the opposite site, so i believe this could be an example of failed validation.

traffic keeps left and cars usually have the steering wheel on the right (RHD: right-hand drive)

**Beet Sprout**

company

1.Product

pros :

* You will have clear vision for your work and will be able to see the impact of your work.
* You will have more opportunities to learn.

cons :

* Long working hours with little recognition.
* The company`s success is dependent on the product`s success. if the product is unsuccessful, there is a risk to loose your job.
* Your skills may become stale if you are not working on new products

2.Startup

pros:

* More opportunities to learn, flexible hours, unique experience, increased job satisfaction, minimal supervision, opportunities for innovation.

cons :

* uncertain job security, heavy workload, long hours, constant change, limited resources, less pay

3.Outsource

Pros

* Have the opportunity to work with international clients
* competitive salary packages
* provide necessary tools to employees
* good onboarding process for new hires
* you can start with no prior experience

cons

* communication issues
* less contribution to decision making process

**Mighty Beet:**

Early testing saves time and money ; this principle is important because if we test the software in the early stages of SDLC we can detect the defects and fix them as early as possible, they will no longer cause any same failures in the later stages of the SDLC. This will let us save software quality costs as long as less failures will occur in the development process later on.

Exhaustive testing is impossible ; this principle is important because even if we find lots of defects in a software, there still remain some undiscovered bugs ( defects ) which might be/or not identified by the end-users later on after deployment. Therefore instead of trying to test everything/every scenario which is impossible in most cases, we need to focus on narrowing the testing scenarios by using test techniques and risk based analysis. Also by the help of proper test techniques we can increase the coverage and our testing would be more effective. In my current job as a process QA, i think i somehow use this technique while auditing agent interactions with customers; usually it is not possible to listen whole long lasting phone calls, all read all the phone/chat transcripts, i focus on critical lines in such communications where most errors occur like the starting few minutes of a phone call where most of the errors occur.