Partition Testing and Musa OP

- Choose one of the bottom-up or top-down model construction procedures for PT (partition test) to perform one of the following:
    1. Starting with a checklist, construct a corresponding set of partitions for PT.
    2. Starting with an overall input domain definition, partition the input domain for PT.
- Enhance your testing model for the previous problem to build a Musa OP to support UBST. In particular, pay attention to and describe the information sources, participants and additional data collection needed, granularity and practicality of the OP constructed.

Partition Testing:

A two-factor authentication software where only numbers can be entered as input. The valid entries will be 6 digits in length. From this we can form the following equivalence classes for a bottom-up construction for PT:
Partition 1: Number of digits <= 5
Partition 2: Number of digits = 6
Partition 3: Number of digits >= 7

Musa OP:

Customers of the software:
Business, Education, Government

Customer Type	Weight
Business	0.45
Education	0.10
Government	0.40
Other	0.05

Users of the software:
Human Users, Programmers, Third Party

User Type	User Profile by Customer Type				Overall User Profile
	Business	Education	Government	Other	
Human Users	0.80	0.80	0.80	0.15	0.7675
Programmers	0.15	0.20	0.20	0.10	0.1725
Third Party	0.05	0.00	0.00	0.75	0.06

System Modes:
Operational, Maintenance, Admin

System Mode	Weight
Operational	0.90
Maintenance	0.08
Admin	0.02

Functions:

Code Generation, Identifying user, Accepting the code, Identifying application, Verify code correctness.

Occurrence rate:

Function	**Occurrence Rate**	**Usage**
Code Generation	Once every 30 seconds	0.05
Accepting Code	On every login (once every 60 days)	0.40
Verify Code Correctness	On every login (once every 60 days)	0.40
Identifying User	On every login (once every 60 days)	0.05
Identifying Application	On every login (once every 60 days)	0.10

This data has been gathered from the usage details from log file of the software. We need to conduct user surveys to ensure that function usage is determined more accurately.

We can see that the Code Generation function has a high occurrence rate, but the usage is very low. However, the Accepting Code and Verify Code Correctness functions have higher usage with lower occurrence rate. This implies that both Accepting Code and Verify Code Correctness are functions that need to be tested very rigorously since the software is usage is high. Furthermore, Application Identification function needs to be tested the second most, followed by Code Generation and User Identification functions.

We can also see that our usage will be highest for Human Users for all four customer types. This implies that we need to test extensively to ensure that human user interaction with the software meets the expectations and is free of defects.