Tu Lam

CS 362 / Instructor Vijay Tadimeti

February 17th, 2021

Homework #5

1. *Date Field*

i) *What will be your equivalence partitions for these specifications? Write its valid and invalid equivalence partitions.*

Answer: From the date field information that was given, we can determine the **valid equivalence** is to be **1-31**, and the **invalid equivalence** partition to be **less than 1 & greater than 31.**

The valid equivalence is 1-31 is due to that this is the valid input option for the date field as the information stated that “accepts value between 1 and 31”. This shows that the date field needed to be one of these numbers in the range for it to be accept.

Since the valid equivalence is only from 1-31, then the invalid equivalence will be from less than 1 and greater than 31. We choose this because since the input only accept number from 1-31, any number above it (e.g., 32, 33…) and the number below it (e.g., 0, -1…) will not fit in with the date field system.

ii) *How would you do the same using the BVA method? Apply BVA for the same specification above.*

Answer: Using the BVA method, which describe as focusing on the input values at boundaries, and in here we can use the BVA method to divide into two boundaries. One boundary is the **below boundary value** which **will consist of 0 and 30**, while the **above boundary value** will contain **2 and 32.**

2. *Username*

i) *What will be your equivalence partitions for these specifications? Write its valid and invalid equivalence partitions.*

Answer: From the username information that was given, we can determine the **valid equivalence** is to be **7-10 characters long**, and the **invalid equivalence** partition to be **less than 7 & greater than 10.**

The valid equivalence is 7-10 characters long is due to that this is the valid input option for the username as the information stated that “A program accepts a username that is 7-10 characters long”. This shows that the username program needed to be one of these numbers in the range for it to be accept.

Since the valid equivalence is only from 7-10, then the invalid equivalence will be from less than 7 and greater than 10. We choose this because since the input only accept number from 7-10, any number above it (e.g., 11, 12…) and the number below it (e.g., 6, 5…) will not fit in with the username system.

ii) *How would you do the same using the BVA method? Apply BVA for the same specification above.*

Answer: Using the BVA method, which describe as focusing on the input values at boundaries, and in here we can use the BVA method to divide into two boundaries. One boundary is the **below boundary value** which will consist of **6 and 9**, while the **above boundary value** will contain **8 and 11**.

3. *Ages*

i) *What will be your equivalence partitions for these specifications? Write its valid and invalid equivalence partitions.*

Answer: From the ages information that was given, we can determine the **valid equivalence** is to be **16-59 & 66-70** years old, and the **invalid equivalence partition** to be **less than 16**, **greater than 70**, and in **between 60-65**.

The valid equivalence is 16-59 and 66-70 years old is due to that this is the valid input option for the username as the information stated that “A program accepts an age as input but the ages it can accept are 16-70 except 60-65”. This shows that the username program needed to be one of these numbers in the range for it to be accept.

Since the valid equivalence is only from 16-59 & 66-70, then the invalid equivalence will be from less than 16, between 60-65, and greater than 70. We choose this because since the input only accept number from 16-59 & 66-70, any number above it (e.g., 71, 72…) and the number below it (e.g., 15, 14…) will not fit in with the ages system. Also, they also exempt the range of 60-65 to also be invalid in the system.

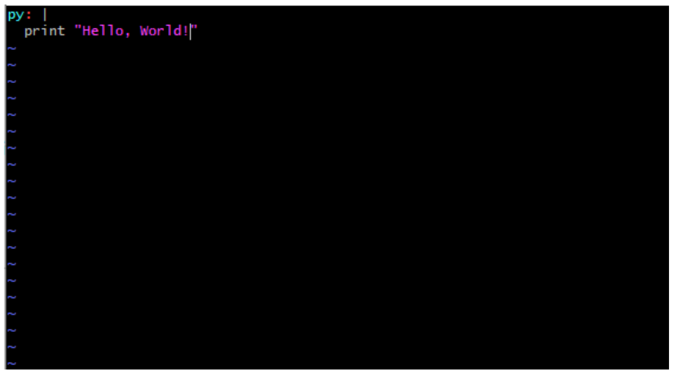
ii) *How would you do the same using the BVA method? Apply BVA for the same specification above.*

Answer: Using the BVA method, which describe as focusing on the input values at boundaries, and in here we can use the BVA method to divide into two boundaries. One boundary is the **below boundary value** which will consist of 15, 58, 65 and 69, while the **above boundary value** will contain 17, 60, 67 and 71.

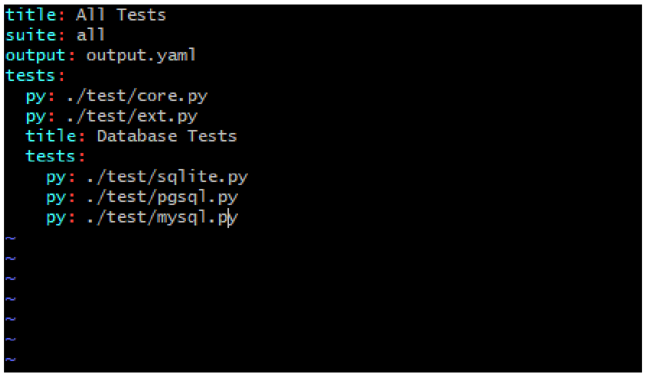
4. *pbbt*

i) *Read the documentation and provide evidence of you trying out the examples or the snippets provided on the documentation page. (Can include screenshots or other examples that you have tried out on your own or Github code).*

Answer: Below is two examples I tried to use the pbbt, one is printing "Hello World!" and the other is using the test cases for my 2 examples.

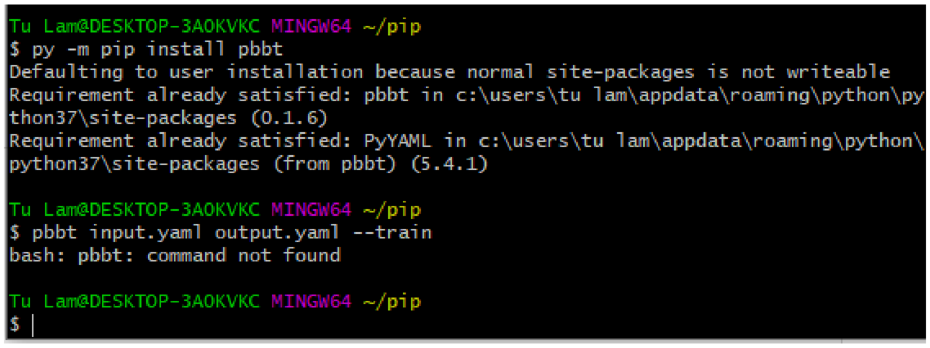


```
py: |
  print "Hello, World!"
```



```
title: All Tests
suite: all
output: output.yaml
tests:
  py: ./test/core.py
  py: ./test/ext.py
  title: Database Tests
  tests:
    py: ./test/sqlite.py
    py: ./test/pgsql.py
    py: ./test/mysql.py
```

But I could not get them to run, so this was my best attempt to use the pbbt.



```
Tu Lam@DESKTOP-3AOKVKC MINGW64 ~/pip
$ py -m pip install pbbt
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pbbt in c:\users\tu lam\appdata\roaming\python\py
thon37\site-packages (0.1.6)
Requirement already satisfied: PyYAML in c:\users\tu lam\appdata\roaming\python\
python37\site-packages (from pbbt) (5.4.1)

Tu Lam@DESKTOP-3AOKVKC MINGW64 ~/pip
$ pbbt input.yaml output.yaml --train
bash: pbbt: command not found

Tu Lam@DESKTOP-3AOKVKC MINGW64 ~/pip
$ |
```

ii) *What did you learn about pbbt? Did you encounter any difficulties?*

Answer: I learn the pbbt is used to test cases using the black-box method by having a nice input and output interface. We just need to test different boundary cases that we can think to test out the program seeing if the expected output matches with the intentional input. I have problem when trying to run pbbt, as first, I was having a hard time install pbbt and after a minute later, I was able to install it. But more problems occur as I tried many times to run and execute the program. I never got it to work and try to search online on how to do it and never got the answer I need, but this is my best attempt I made as far as pbbt.

5. ***Unit Testing***

i) *Explain whether or not unit testing is considered black-box or white-box testing.*

Answer: Unit testing is considered to be a **white-box testing**. This is true due to black-box testing is only considered to test the functionality requirement and test to see any defects that present in the system without knowing any code implementation behind it. With white-box testing, it is the opposite as you test out one component in the system, but to see where the code fail to meet the validation it need to pass the test. With this, we can categorize unit testing to be a white-box testing.

ii)

1. *Black-box testing that you would perform on your favorite social media system from in-class activity 1.*
   Answer: For the social media Snapchat, for a black-box testing example. We were thinking to test the boundary for the age when you first create a snapchat account. Test to see if the functionality work if someone enter in their birthdate and it is below 13 years old, if so then the app will ask user that they must be 13 years old or older to create an account.

2. *Unit-testing that you would perform on your favorite social media system from in-class activity 1.*
   Answer: For a unit-test that we can create using the Snapchat app is to test if the message will pop up if the user upload more than 10 picture/videos saying they have reached a limit to

upload. If they do pop up, we can validate it and if it does not, then there was something wrong behind the scenes with the unit system itself.