**Mini-Project**

**Khasa Gillani**

1. **Self-Learning :**

Resources from where I have been studying about smart contract and solidity language:

* <https://cryptozombies.io/>
* <https://www.tutorialspoint.com/solidity/solidity_constructors.htm>
* <https://www.trufflesuite.com/docs/truffle/overview>

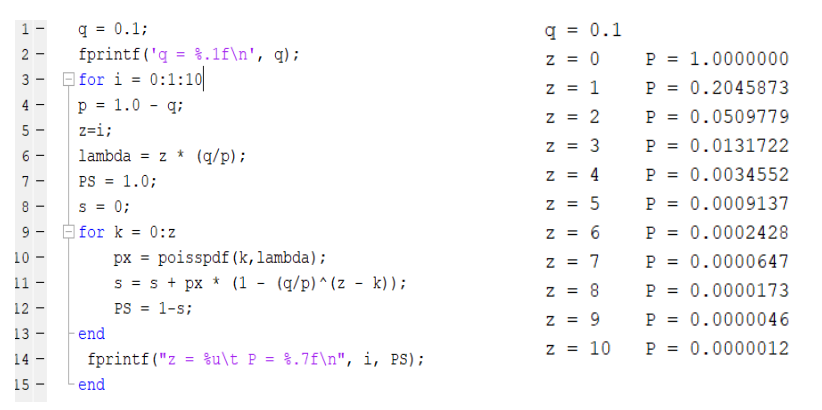
**What I have learnt so far:**

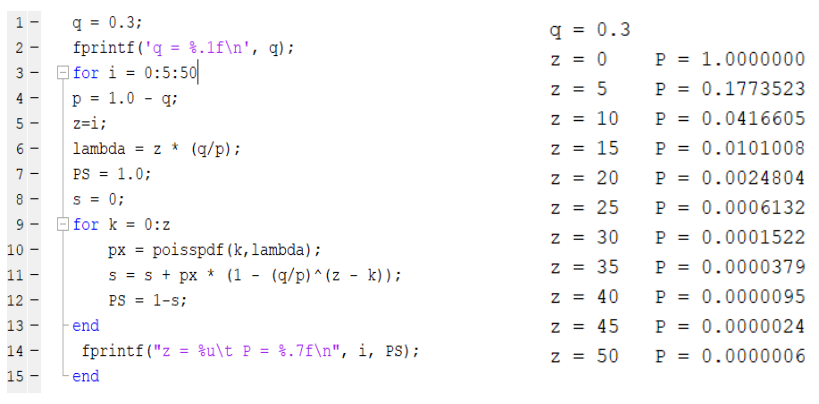
* The fundamentals of solidity language
* How to develop and deploy smart contracts
* Getting hand on experience by going through zombie factory smart contract and other small unit tests

Apart from that I have been doing certification of Blockchain on <https://www.coursera.org/> and enrolled in different courses:

* Blockchain Basics
* IBM Blockchain Foundation for Developers

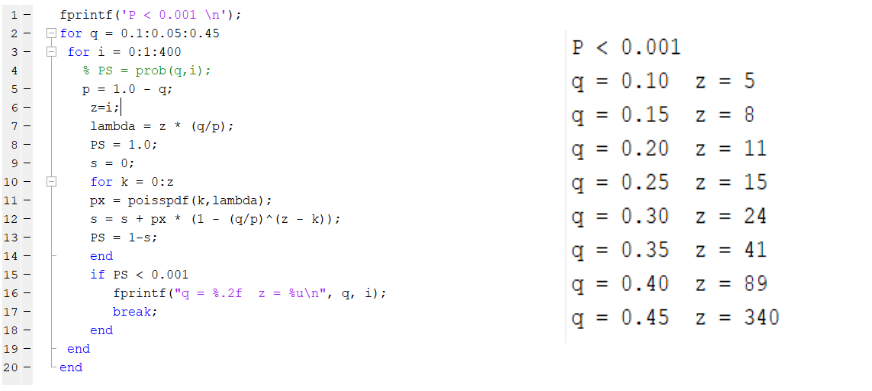
Reason of doing these certifications is to get more expertise and developing understanding of Blockchain and how it works.For getting more insight of foundation of Blockchain I have studied academic literature and implemented the concept of Bitcoin white paper <https://bitcoin.org/bitcoin.pdf> in MATLAB. Below I have attached the sample code:





probability drop off exponentially with z.

**Solving for P < 0.001:**



After learning through these platforms, I am confident enough to work on blockchain projects because now I can say I have essential knowledge for getting started. However still need long way to get more expertise.

1. **Token Exchange Overview**
2. **ERC20**

This smart contract comprises of functions for exchange of ERC tokens between various accounts. It includes the following

* ***Functions***:
  + ***Total Supply***: Return the total number of tokens from associated account. Such as, the subject who has deployed the account “CryptoPayment” is the owner of account.
  + SetToken: Set the number of tokens and owner account address value
  + ***transfer***: Transfer given token from owner ‘account to recipient’s account
  + TransferFrom: Transfer given token from mentioned sender account to recipient account
  + Allowance: check transfer permission of tokens from one account to another
  + Approve: set the number of token allowed for transfer from account A to account B
  + balanceOf: Return balance in form of token in given account
* ***Events***:
* Two events ***Transfer*** and ***Approve*** are declared which get to be triggered upon transfer of token and approval of transaction respectively.
* SafeMath is used to handle integer related errors

1. **Library: SafeMath**

Safemath library provides the safe use signed/unsigned numbers in solidity such as overflow in addition, underflow in subtraction and multiplication and division.

1. **Demo**
2. Let’s assume three accounts in this scenario:
   1. 0x7B3B3B38670A4390B06B569E5a0509a8C16Ab8bb
   2. 0xBe5F751F5b03F5e5Be2e1b00840F705405478280
   3. 0x3d7a6ee3f43C66C398A1216733c5E67edDF4b43D
3. Compile smart contract “MyToken.sol
4. Deploy “MyToken.sol” with name, symbol, decimal point and number of tokens. on account a – [refer to figure 1]

A screenshot of a computer

Description automatically generated

Fig.1. Smart contract deployed

1. Now accounts having following status
   1. {“0x7B3B3B38670A4390B06B569E5a0509a8C16Ab8bb”, balance : “100”
   2. {“0xBe5F751F5b03F5e5Be2e1b00840F705405478280”, balance : “0”}
   3. {“0xBe5F751F5b03F5e5Be2e1b00840F705405478280”, balance : “0”}

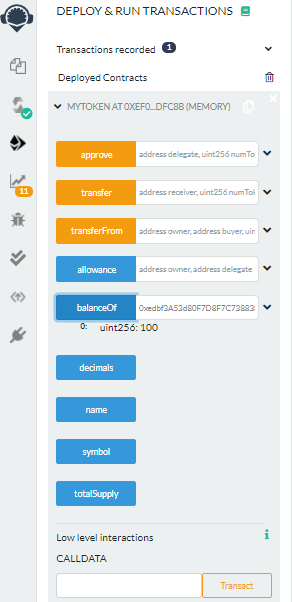


Fig.2. Balance in account a

x

1. Check balance of account a as shown in figure 2.

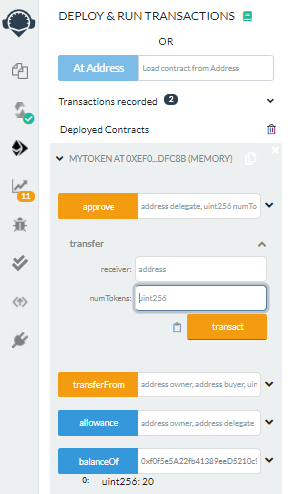
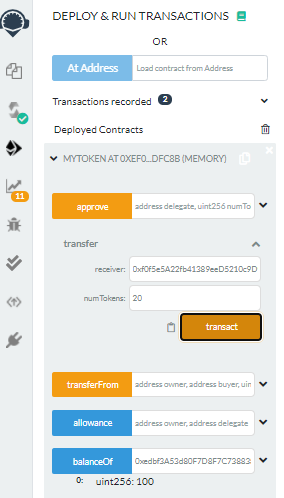


Fig.3. Token transfer from a to b

1. Token Exchange: Case 1
   1. Execute transfer function that result in Transfer of 20 token from a to b as shown in fig.3
      * {“0xedbf3A53d80F7D8F7C738838A01626AABdE2Caa4”, balance: “80”
      * {“0xf0f5e5A22fb41389eeD5210c9D9203Ea65EF1082”, balance: “20”}
      * {“0x444712aDc15E1B54a916741dF032d9380C45b561”, balance: “0”}
2. Token Exchange: Case 2
   1. Select account b from list of account
   2. Execute transfer function that result in Transfer of 10 token from b to c as shown in fig.4
      * {“0xedbf3A53d80F7D8F7C738838A01626AABdE2Caa4”, balance: “80”
      * {“0xf0f5e5A22fb41389eeD5210c9D9203Ea65EF1082”, balance: “10”}
      * {“0x444712aDc15E1B54a916741dF032d9380C45b561”, balance: “10”}
3. Token Exchange: Case 3
   1. Select account c from list of account
   2. Execute transfer function that result in Transfer of 5 token from c to a as shown in fig.5
      * {“0xedbf3A53d80F7D8F7C738838A01626AABdE2Caa4”, balance: “85”
      * {“0xf0f5e5A22fb41389eeD5210c9D9203Ea65EF1082”, balance: “10”}
      * {“0x444712aDc15E1B54a916741dF032d9380C45b561”, balance: “5”}

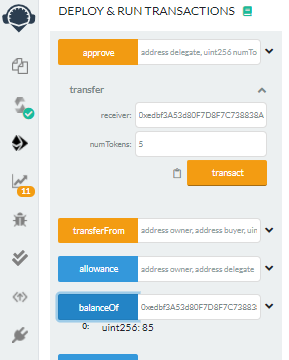


Fig.5. Token transfer from c to a

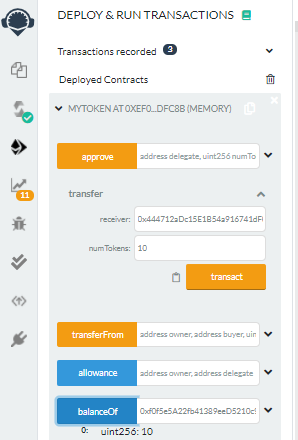


Fig.4. Token transfer from b to c