THE UNIVERSITY OF SYDNEY

Due: Monday Week 2 by 11AM

COMP2121 Assignment 0: Introduction

The goal of this assignment is to help you familiarise yourself with the new version of the PASTA *system and understand the context of this year's new programming assignments. This year, the assignments of COMP2121 have been fully redesigned: Instead of implementing an SMTP server and a P2P Twitter application, you will implement a blockchain. Since it is a very advanced computer system, we introduce the blockchain context in the remainder of this assignment. In particular, at the end you should have an overview of what a blockchain is, how many features are expected, and how to properly test and submit your work.*

1 What is a blockchain?

You may not have heard of blockchain before, but I bet you must be surrounded with Bitcoin news these days. Bitcoin, is a *distributed system* offering a digital currency, called the *bitcoins*, created used and verified independently of any central bank. It is guarded by cryptographic primitives and it recently gained momentum for no single person or government can fully control it. Thanks to being distributed, it has no single point of failure, can even tolerate attacks from multiple hackers. It offers *pseudonymity* in that users do not need to reveal their identity. The exchange rate of this promising currency has increased from 1300 AUD/coin to 3300 AUD/coin this year and many people around the world are currently providing this peer-to-peer service by mining bitcoins. Yet, bitcoin suffers limitations making blockchain an interesting area of research. As a little hacker like you, you might be interested in knowing how Bitcoin works, and what system underneath supports the operation of Bitcoin. The answer is "blockchain".

Traditionally, the blockchain implements a distributed ledger as a chain of blocks, each block storing a series of transactions just like the page of a ledger. Recently, blockchain has found broader applications. People and companies start using blockchain for other purposes, like storing business contracts, recording patients medical history; some people even declare love and marriage on it. No matter how fancy its usage is, the goal of all blockchains remains the same—to maintain the integrity of some information potentially indefinitely.

From a developer's perspective, the components of blockchain are nothing special. The data structure underneath is simply a linked list (that you might have seen or will learn about it rapidly in your data structure course) that represents the chain of blocks. The distributed system is built upon a peer-to-peer (P2P) network (that you will learn in this course). To make records consistent on each participant's computer, you will learn about consensus algorithms. Some security mechanisms like encryption and digital signature will be used to provide confidentiality and authentication. By combining these four building blocks, you will be able to create a blockchain yourself!

2 How many features are expected to implement during the semester?

The blockchain core data structure (Linked list + P2P network) is required. While a fully functional blockchain typically requires various additional features to ensure security, consistency, etc., in this project, we will just implement one simple security feature to make the blockchain tamper-proof. A tamper-proof blockchain is achieved using hash pointers between blocks as links of the linked list.

2.1 Hash function

Before talking about how hash pointers are used in the linked list, let me give you a quick introduction of hash functions. A hash function is a special class of function that given an arbitrary object as input, produces the output which is a fixed length of bits. Those output bits are called the hash value. A good cryptographic hash function has three important properties:

2.1.1 Pre-image resistance

Given a hash value h = hash(m), it should be difficult to find the original message m.

2.1.2 Second pre-image resistance

Given an input m1, it should be hard to find another input m2 (not equal to m1) that generates the same hash value hash(m1) = hash(m2).

2.1.3 Collision resistant

It is hard to find two distinct messages m1, m2 that hash to the same output hash(m1) = hash(m2).

Cryptographic hash functions are commonly used in various computer systems and protocols. They provide a strong guarantee that if an attacker somehow changes the message, then the new hash value of the tampered message will be different to the original hash value. The victim could easily validate the message by checking if two hash values match each other.

How to build a good cryptographic hash function and show its mathematical proof is beyond this course. But with the help of java security library. It is simple to calculate the hash value with standard hash functions, like the SHA-256 hashing algorithm used in this course. Below is the code to calculate an SHA-256 hash of message **12345** and print the hash value encoded in base64 encoding.

```
1 import java.io.ByteArrayOutputStream;
2 import java.io.DataOutputStream;
3 import java.io.IOException;
4 import java.security.MessageDigest;
```

```
5  import java.security.NoSuchAlgorithmException;
6  import java.util.Base64;
7
8  public class Assignment0 {
9
10     public static void main(String []args) {
11         System.out.println(hashMessage("12345"));
12     }
13
14     public static String hashMessage(String message) {
15         String hashString = "";
16         try {
17             MessageDigest digest = MessageDigest.getInstance("SHA-256");
18
19             ByteArrayOutputStream baos = new ByteArrayOutputStream();
20             DataOutputStream dos = new DataOutputStream(baos);
21
22             dos.writeUTF(message);
23             byte[] bytes = baos.toByteArray();
24             byte[] hash = digest.digest(bytes);
25             hashString = Base64.getEncoder().encodeToString(hash);
26         } catch (NoSuchAlgorithmException e) {
27             e.printStackTrace();
28         } catch (IOException e) {
29             e.printStackTrace();
30         }
31         return hashString;
32     }
33
34
35 }
```

Before going on, let me give you your first assignment (Assignment 0) to test your understanding so far.

Task 1 hash an array of messages

Using the skeleton code below, try to compute the hash of an array of messages. Hint: You should not concatenate messages into a single string, and then write the string to **`DataOutputStream`**. Instead, you should use a for loop to write messages inside the array to **`DataOutputStream`** one by one. The order to write messages into **`DataOutputStream`** should be the same order as messages stored in the array.

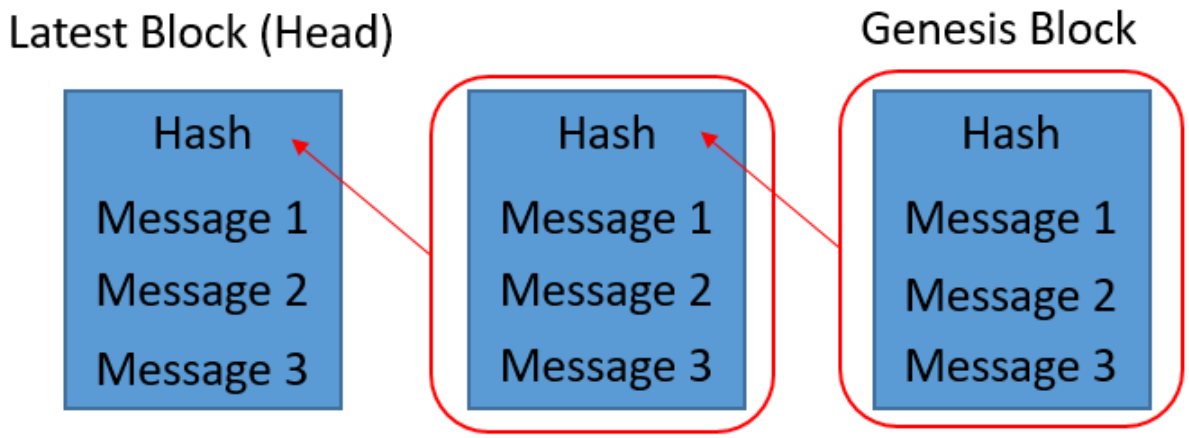
Assignment 0 counts for 0 mark. However, please put all source code under **`src`** folder, and zip the **`src`** folder to submit your solution on PASTA (http://soit-app-pro-12.ucc.usyd.edu.au:8080/PASTA/). The archived file should be named **`src.zip`**. Please remove all test cases you have written before submitting. This is the default way for submitting your assignment.

```
1  import java.io.ByteArrayOutputStream;
```

```
2  import java.io.DataOutputStream;
3  import java.io.IOException;
4  import java.security.MessageDigest;
5  import java.security.NoSuchAlgorithmException;
6  import java.util.Base64;
7  import java.util.ArraytList;
8
9  public class Assignment0 {
10
11     public static void main(String []args) {
12         ArrayList<String> messages = new ArrayList<>();
13         messages.add("12345");
14         messages.add("12345");
15         System.out.println(hashMessages(messages));
16         // you should see the hash value calculated is
17         // 7ccQDEgOM+OWvega620WYCXJHs1w53wSumP329xFirw=
18     }
19
20     public static String hashMessages(ArrayList<String> messages) {
21         String hashString = "";
22         try {
23             MessageDigest digest = MessageDigest.getInstance("SHA-256");
24
25             ByteArrayOutputStream baos = new ByteArrayOutputStream();
26             DataOutputStream dos = new DataOutputStream(baos);
27
28             // write you code here
29
30             byte[] bytes = baos.toByteArray();
31             byte[] hash = digest.digest(bytes);
32             hashString = Base64.getEncoder().encodeToString(hash);
33         } catch (NoSuchAlgorithmException e) {
34             e.printStackTrace();
35         } catch (IOException e) {
36             e.printStackTrace();
37         }
38         return hashString;
39     }
40
41
42 }
```

2.2 Linked list

Now, it is time to create our first linked list using hash pointers. Like typical linked lists, we have nodes and pointers. Each block (node) in the blockchain has usually two types of fields. The first are the messages stored in the block. The second is the block pointer. Blockchain utilizes the hash value of the previous block as the pointer. Its structure is as follows.



A blockchain uses the linked list shown above to store committed messages. A committed message is a message that has been permanently stored in the blockchain, and there is (presumably) no way to change it in the future.[1] Suppose an attacker would like to change one of the messages in an intermediate block. In order to make sure the chain is correctly linked, the attacker also needs to update the hash value stored in the next block, and the hash value stored in the next next block, and so on until the latest block. This indicates that every change made on the old messages will have an impact on the latest hash value. Just by checking the latest hash value, you can confidently tell if any tampering happened before. Uncommitted messages are stored in another pool. Once the pool meets certain criteria, messages in the pool get committed, and a new block is formed in the blockchain.

Once you have mastered how to construct the blockchain using hash pointers, you are ready to start implementing the next core building block, that is the P2P network. But, no need to hurry! We are going to teach you how to do this step by step during the semester.

3 How to properly test and submit my work?

To test your code, you are encouraged to write your test cases and thoroughly test them before submitting your code. There are three reasons for that. First of all, as a developer, you should be responsible for everything you wrote, and testing improves your confidence. Secondly, since the test is designed by yourself, it is a white-box testing rather than a black-box testing PASTA provides you. You can fix your bugs quickly. Thirdly, there will be a lot of

[1] You will see later in this course under what assumptions this can be ensured.

students testing their code on PASTA when approaching the deadline and the waiting queue could be long in case the server becomes overloaded. You may have to wait a long time before your code gets tested, it could translate in a waste of your time if you realise afterwards that there was a small typo in your submission...

If you need some extra help on the assignments or lecture materials, the discussion forum is the place to ask questions. Everyone is welcomed to post questions on the forum. We do not expect you to send emails directly to the lecturer or tutors, because your questions may also be valuable to other students.

Typically we are getting tons of questions during the submission deadline. To cope with this, we subdivided assignments into tasks. Assignments have hard due dates, which means every submission made after the assignment due date will have a penalty. Tasks, in contrast, have a soft due date. You are encouraged to finish it before the soft due date. But you could catch up later in case you are busy that week (no later than assignment due date). This helps in two ways. First, it helps you organize your time and distribute your load. Second, since this is the first time this assignment is applied, there may exist some ambiguity in the assignment specification. (If this happens please inform the TA via emails as soon as possible.) *The due date will not be postponed.* Once again, please organize your time wisely. Finishing assignments in one night before the deadline is infeasible.

The submission system we use this semester, as usual, is PASTA. PASTA experience a major update for this semester. We are welcoming any suggestions to the new PASTA system, tasks and assignments through PASTA itself. If you have any feedback regarding the PASTA system, please email the TA. Assignment 0 is precisely to test that you could properly use PASTA. After Assignment 0, we will not accept excuses that a student is not familiar with PASTA.