1. Find a collision in each of the hash functions below:

a. H(x) = x mod 712, where x can be any integer

**Answer:** Let’s consider the collision at 2 then values of x will be 2 and 712 +2

b. H(x) = # of 1-bits in x, where x can be any bit string

**Answer:** The collision will be on the value of 011 and 101.

c. H(x) = the three least significant bits of x, where x can be any bit string

**Answer:** The collision will be on the value of 1**000** and 101**000**.

2. Prove the statement: In a class of 500 students, there must be two students with the same birthday.

**Answer:**

There are 365 days in a year.

So the feasible birthdays all-round the year can be 365.

The class has 500 students.

Using the Pigeon-hole principle, there must be two students with the same birthday

1. Pigeonholes: There can be birthdays from the 365 days [1-365].
2. Pigeon: There are 500 student birthdays [1-500].
3. Collison: There must be two students with the same birthday.

3. Find an x such that H (x ◦ id) ∈ Y where

a. H = SHA-256

b. id = 0xED00AF5F774E4135E7746419FEB65DE8AE17D6950C95CEC3891070FBB5B03C77

c. Y is the set of all 256 bit values that have some byte with the value 0x1D. Assume SHA-256 is puzzle-friendly. Your answer for x must be in hexadecimal.

**Answer:**

As per my understanding I solved the above problem.

Please find the java code snippet for the same.Brute force

|  |
| --- |
| Public static void solve(byte[] id) throws NosuchAlgorithmException, IOException{  Random r = new Random;  MessageDigest digest = MessageDigest.getInstance("SHA-256");  generate: for (long i =0 ; I < 1000000000; i++) {  System.out.println("ATTEMPT: "+ i);  byte[] x = new byte[10];  r.nextBytes(x);  ByteArrayOutputStream outputstream = new ByteArrayOutputStream();  Outputstream.write(x);  Outputstream.write(id);  byte concat[] = outputstream.toByteArray();  byte[] hash = digest.digest(concat);  System.out.println("x: " + DatatypeConverter.printHexBinary(x);  System.out.println("H(x | id): " + DatatypeConverter.printHexBinary(hash);  for (byte b : hash){  if(b == 0x1D){  break generate;  }  } |

The final output will give the hash hex number which will be the same hash as the concatenated value of x and id.

4. Alice and Bob want to play a game over text where Alice chooses a number between 1 and 10 in her head, and then Bob tries to guess that number. If Bob guesses correctly, he wins. Otherwise, Alice wins. However, Bob complains that the game isn’t fair, because even if Bob guessed correctly, Alice could lie and claim that she chose a different number than what she initially chose. What can Alice do to prove that she didn’t change the number she initially chose? Devise a mechanism to address Bob’s concern. Provide a detailed explanation of the mechanism and why it works. An answer with insufficient detail will not receive credit.

**Answer:**

So the problem here is the trust as to whether Alice will lie or not.

In that case we can use the hash functions to address the Bob’s concern and address the game fairness.

Things to consider for the mechanism to work properly.

1. The number chosen by Alice should be hidden from Bob
2. The number chose by Alice should not be change.
3. Bob can verify the number chosen by the Alice in phase 2.
4. In this case puzzle friendliness must be consider.

Work flow mechanism to improve the fairness of the game.

1. I postulate as Alice should choose the number from 1 – 10 and should combine with the randomly generated nonce(key).
2. Now pass this combination from Hash function.
3. This will satisfy the secrecy of the Alice selection will be maintained.
4. Now to make sure that the value is not replace by another number; we can apply the SHA-256 function in order to use the collision infeasible property ad hiding scheme in commitment scheme.
5. Now once Bob has guessed the number then using the nonce/key and SHA-256 function hash function he will create the commitment which will verify that the Alice commitment is same as Bob generated hash.
6. If the Hash matches; then Bob has correctly guessed the Alice selection of number.
7. Hiding and Binding property of the commitment scheme will help to find the solution to the Bob’s concern about the game fairness.

This mechanism uses commitments based on a cryptographic hash function, SHA256. Because SHA256 is collision resistant, Bob can convince himself that Alice won’t be able to find two X’s that hash to the same number to lie to Bob  
about her true guess. Because SHA256 has the hiding property when the input is appended with a cryptographically  
random nonce, Alice can convince herself that Bob won’t be able to reverse the hash she sends him to find X. After the  
game ends, Bob can verify that Alice is not lying by calculating SHA256(X ◦ R) himself and comparing it to the hash Alice sent in step 3.