Saleh Alkhalifa
9/19/2022

CS 5200
Module 2

Homework#2

Problem 1 (15 points, 5 each)

Find:
(a) 3^{1500} *mod* 11

<u>Divide B into powers of 2:</u>

$$1500 = 10111011100$$

$$1500 = \cancel{2^0} + \cancel{2^1} + 2^2 + 2^3 + 2^4 + \cancel{2^5} + 2^6 + 2^7 + 2^8 + \cancel{2^9} + 2^{10}$$

$$1500 = 2^2 + 2^3 + 2^4 + 2^6 + 2^7 + 2^8 + 2^{10}$$

$$3^{1500} mod\ 11 = 3^{(4+8+16+64+128+256+1024)} mod\ 11$$

$$3^{1500} mod\ 11 = (3^4 + 3^8 + 3^{16} + 3^{64} + 3^{128} + 3^{256} + 3^{1024})\ mod\ 11$$

<u>Calculate mod C of the powers ≤ B:</u>

$$3^4\ mod\ 11 = 81\ mod\ 11 = 4$$

$$3^8 mod\ 11 = 6561\ mod\ 11 = 5$$

$$3^{16}\ mod\ 11 = (3^8 * 3^8)\ mod\ 11 = (3^8 \text{mod } 11 * 3^8\ mod\ 11)\ mod\ 11 = (5 * 5)\ mod\ 11 = 3$$

$$3^{32}\ mod\ 11 = (3^{16} * 3^{16})\ mod\ 11 = (3^{16} \text{mod } 11 * 3^{16}\ mod\ 11)\ mod\ 11 = (3 * 3)\ mod\ 11 = 9$$

$$3^{64}\ mod\ 11 = (3^{32} * 3^{32})\ mod\ 11 = (3^{32} \text{mod } 11 * 3^{32}\ mod\ 11)\ mod\ 11 = (9 * 9)\ mod\ 11 = 4$$

$$3^{128}\ mod\ 11 = (3^{64} * 3^{64})\ mod\ 11 = (3^{64} \text{mod } 11 * 3^{64}\ mod\ 11)\ mod\ 11 = (4 * 4)\ mod\ 11 = 5$$

$$3^{256}\ mod\ 11 = (3^{128} * 3^{128})\ mod\ 11 = (3^{128} \text{mod } 11 * 3^{128}\ mod\ 11)\ mod\ 11 = (5 * 5)\ mod\ 11 = 3$$

$$3^{512}\ mod\ 11 = (3^{256} * 3^{256})\ mod\ 11 = (3^{256} \text{mod } 11 * 3^{256}\ mod\ 11)\ mod\ 11 = (3 * 3)\ mod\ 11 = 9$$

$$3^{1024}\ mod\ 11 = (3^{512} * 3^{512})\ mod\ 11 = (3^{512} \text{mod } 11 * 3^{512}\ mod\ 11)\ mod\ 11 = (9 * 9)\ mod\ 11 = 4$$

<u>Use modular multiplication properties and combine values:</u>

$$3^{1500} mod\ 11 = (3^4 + 3^8 + 3^{16} + 3^{64} + 3^{128} + 3^{256} + 3^{1024})\ mod\ 11$$

$$3^{1500} mod\ 11 = (3^4\ mod\ 11 * 3^8\ mod\ 11 * 3^{16}\ mod\ 11 * 3^{64}\ mod\ 11 * 3^{128}\ mod\ 11 * 3^{256}\ mod\ 11 * 3^{1024})\ mod\ 11$$

$$3^{1500} mod\ 11 = (4 * 5 * 3 * 4 * 5 * 3 * 4) mod\ 11$$

$$3^{1500} mod\ 11 = 14400\ mod\ 11 = 1$$

$$\therefore \textbf{Therefore, } \mathbf{3^{1500}\ \boldsymbol{mod}\ 11 = 1}$$

Saleh Alkhalifa
9/19/2022

CS 5200
Module 2

Homework#2

(b) 5^{4358} *mod* 10

Divide B into powers of 2:

$$4358 = 1000100000110$$

$$5^{4358} = \cancel{2^0} + 2^1 + 2^2 + \cancel{2^3} + \cancel{2^4} + \cancel{2^5} + \cancel{2^6} + \cancel{2^7} + 2^8 + \cancel{2^9} + \cancel{2^{10}} + \cancel{2^{11}} + 2^{12}$$

$$4358 = 2^1 + 2^2 + 2^8 + 2^{12}$$

$$4358 = 2 + 4 + 256 + 4096$$

$$5^{4358} \ mod \ 10 = 5^{(2+4+256+4096)} \ mod \ 10$$

$$5^{4358} \ mod \ 10 = (5^2 + 5^4 + 5^{256} + 5^{4096}) \ mod \ 10$$

Calculate mod C of the powers ≤ B:

$$5^2 \ mod \ 10 = 25 \ mod \ 10 = 5$$

$$5^4 \ mod \ 10 = 625 \ mod \ 10 = 5$$

$$5^8 \ mod \ 10 = (5^4 * 5^4) \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{16} \ mod \ 10 = (5^8 * 5^8) \ mod \ 10 = 5^8 \ mod \ 10 * 5^8 \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{32} \ mod \ 10 = (5^{16} * 5^{16}) \ mod \ 10 = 5^{16} \ mod \ 10 * 5^{16} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{64} \ mod \ 10 = (5^{32} * 5^{32}) \ mod \ 10 = 5^{32} \ mod \ 10 * 5^{32} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{128} \ mod \ 10 = (5^{64} * 5^{64}) \ mod \ 10 = 5^{64} \ mod \ 10 * 5^{64} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{256} \ mod \ 10 = (5^{128} * 5^{128}) \ mod \ 10 = 5^{128} \ mod \ 10 * 5^{128} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{1028} \ mod \ 10 = (5^{256} * 5^{256}) \ mod \ 10 = 5^{256} \ mod \ 10 * 5^{256} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{2056} \ mod \ 10 = (5^{1028} * 5^{1028}) \ mod \ 10 = 5^{1028} \ mod \ 10 * 5^{1028} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

$$5^{4096} \ mod \ 10 = (5^{2056} * 5^{2056}) \ mod \ 10 = 5^{2056} \ mod \ 10 * 5^{2056} \ mod \ 10 = (5 * 5) \ mod \ 10 = 5$$

Use modular multiplication properties and combine values:

$$5^{4358} \ mod \ 10 = (5^2 + 5^4 + 5^{256} + 5^{4096}) \ mod \ 10$$

$$5^{4358} \ mod \ 10 = (5^2 \ mod \ 10 * 5^4 \ mod \ 10 * 5^{256} \ mod \ 10 * 5^{4096} \ mod \ 10) \ mod \ 10$$

$$5^{4358} \ mod \ 10 = (5 * 5 * 5 * 5) \ mod \ 10$$

$$5^{4358} \ mod \ 10 = 625 \ mod \ 10 = 5$$

∴ Therefore, $5^{4358} \ mod \ 10 = 5$

Saleh Alkhalifa
9/19/2022

CS 5200
Module 2

Homework#2

(c) 6^{22345} *mod* 7

Divide B into powers of 2:

$$22345 = 101011101001001$$

$$22345 = 2^0 + \cancel{2^1} + \cancel{2^2} + 2^3 + \cancel{2^4} + \cancel{2^5} + 2^6 + \cancel{2^7} + 2^8 + 2^9 + 2^{10} + \cancel{2^{11}} + 2^{12} + \cancel{2^{13}} + 2^{14}$$

$$22345 = 2^0 + 2^3 + 2^6 + 2^8 + 2^9 + 2^{10} + 2^{12} + 2^{14}$$

$$22345 = 1 + 8 + 64 + 256 + 512 + 1024 + 4096 + 16384$$

$$6^{22345} \, mod \, 7 = 6^{(1+8+64+256+512+1024+4096+16384)} \, mod \, 7$$

$$6^{22345} \, mod \, 7 = (6^1 + 6^8 + 6^{64} + 6^{256} + 6^{512} + 6^{1024} + 6^{4096} + 6^{16384}) \, mod \, 7$$

Calculate mod C of the powers ≤ B:

$$6^1 \, mod \, 7 = 6$$

$$6^2 \, mod \, 7 = 36 \, mod \, 7 = 1$$

$$6^4 \, mod \, 7 = 1296 \, mod \, 7 = 1$$

$$6^8 \, mod \, 10 = (6^4 * 6^4) \, mod \, 7 = 6^4 \, mod \, 7 * 6^4 \, mod \, 7 = (1 * 1) \, mod \, 7 = 1$$

$$6^{64} \, mod \, 10 = (6^8 * 6^8 * 6^8 * 6^8) \, mod \, 7 = 6^4 \, mod \, 7 * 6^4 \, mod \, 7 * 6^4 \, mod \, 7 * 6^4 \, mod \, 7 = 1 \, mod \, 7 = 1$$

$$6^{256} \, mod \, 10 = (6^{64} * 6^{64} * 6^{64} * 6^{64}) \, mod \, 7 = 6^{64} \, mod \, 7 * 6^{64} \, mod \, 7 * 6^{64} \, mod \, 7 * 6^{64} \, mod \, 7 = 1 \, mod \, 7 = 1$$

$$6^{512} \, mod \, 10 = (6^{256} * 6^{256}) \, mod \, 7 = 6^{256} \, mod \, 7 * 6^{256} \, mod \, 7 = 1 \, mod \, 7 = 1$$

$$6^{1024} \, mod \, 10 = (6^{512} * 6^{512}) \, mod \, 7 = 6^{512} \, mod \, 7 * 6^{512} \, mod \, 7 = 1 \, mod \, 7 = 1$$

$$6^{4096} \, mod \, 10 = (6^{1024} * 6^{1024} * 6^{1024} * 6^{1024}) \, mod \, 7 = 6^{1024} \, mod \, 7 * 6^{1024} \, mod \, 7 * 6^{1024} \, mod \, 7 * 6^{1024} \, mod \, 7 = 1 \, mod \, 7 = 1$$

$$6^{16384} \, mod \, 10 = (6^{4096} * 6^{4096} * 6^{4096} * 6^{4096}) \, mod \, 7 = 6^{4096} \, mod \, 7 * 6^{4096} \, mod \, 7 * 6^{4096} \, mod \, 7 * 6^{4096} \, mod \, 7 = 1 \, mod \, 7 = 1$$

Use modular multiplication properties and combine values:

$$6^{22345} \, mod \, 7 = (6^1 + 6^8 + 6^{64} + 6^{256} + 6^{512} + 6^{1024} + 6^{4096} + 6^{16384}) \, mod \, 7$$

$$6^{22345} \, mod \, 7 = (6 * 1 * 1 * 1 * 1 * 1 * 1 * 1) \, mod \, 7$$

$$6^{22345} \, mod \, 7 = 6 \, mod \, 7 = 6$$

∴ Therefore, 6^{22345} *mod* 7 = 6

Saleh Alkhalifa
9/19/2022

CS 5200
Module 2

Homework#2

Problem 2 (15 points, 5 each)

Compute:

(a) GCD (648, 124)

Use Prime Factorization Method:

$$GCD(648, 124)$$

$$124 = 2 * 2 * 31$$

$$648 = 2 * 2 * 2 * 3 * 3 * 3 * 3$$

$$Common\ Factors = 2 * 2 = 4$$

∴ Therefore, $\boldsymbol{GCD(648, 124) = 4}$

(b) GCD (123456789, 123456788)

Use Euclid's Algorithm: $Let\ A \geq B, GCD(A, B) = GCD(B, A\ mod\ B)$

$$GCD(123456789, 123456788)$$

$$Let\ A = 123456789$$

$$Let\ B = 123456788$$

$$GCD(123456789, 123456788) = \mathrm{GCD}(123456788, 123456789 \bmod 123456788)$$

$$= \mathrm{GCD}(123456788, 123456789 \bmod 123456788) = \frac{|123456789 * 123456788|}{123456789 * 123456788} = 1$$

∴ Therefore, $\boldsymbol{GCD(123456789, 123456789) = 1}$

(c) GCD (2^{300}*3^{200}, 2^{200})

Use Prime Method:

$$What\ we\ can\ do\ is\ find\ the\ common\ prime\ terms:$$

$$2^3 = 1, 2, \ldots$$

$$3^2 = 1, 3, \ldots$$

$$Common\ Factors = 1\ (There\ are\ no\ others.)$$

∴ Therefore, $GCD(2^{300}, 3^{200}, 2^{200}) = \mathbf{1}$

Saleh Alkhalifa CS 5200
9/19/2022 Module 2

Homework#2

Problem 3 (10 points)

Alice wants to secretly send Bob a specific number. They can communicate only over a public (non-secret, insecure) channel. How to do it using Diffie-Hellman Key Exchange Protocol?

Reminder: The Diffie-Hellman Key Exchange Protocol allows Alice and Bob to jointly establish a shared secret number over an insecure channel. But this secret number does not necessarily coincide with the specific number (which Alice wants to send to Bob).

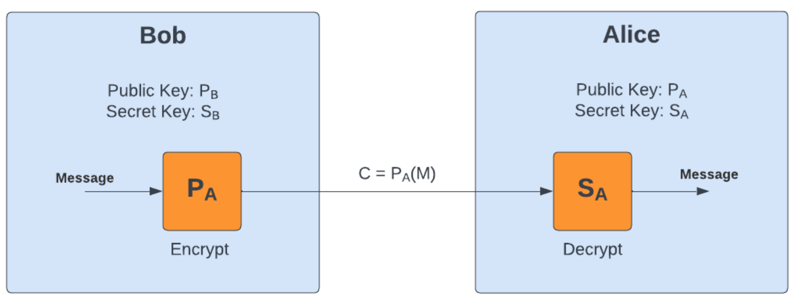
Explanation:

The Diffie-Hellman key exchange was one of the earliest methods used to safely develop and exchange keys within the environment of an unsecure network. Also known as exponential key exchange, the idea here is the use of a method for digital encryption that utilizes number that are raised to specific powers that in return produce decryption keys, on the basis of components that are never transmitted. First, **Alice** and **Bob** will need to determine a prime number *P* and a generator *g*. Alice then determines a random number x, and send the number g^x to Bob, and in return, Bob will determine a random number *y*, and respond with g^y, and in this moment, only Bob and Alice can compute the value:

$$g^{xy} = (g^x)^y = (g^y)^x$$

It is important to note that an eavesdropper will not able to compute this value in a reasonable time, since the process of decrypting them computationally expensive. The algorithm here allows those who have never met before communicate in a secure manner within having to have set a shared key beforehand. We can accomplish a communication within a public channel using this key exchange, over the course of several steps:

1- In order for Bob to send Alice a secure message, Bob will need to obtain Alice's public key, which we will call P_A from a given public directory.
2- Bob will then need to compute a ciphertext C for message M, given that $C = P_A(M)$.
3- Finally, Alice will then receive the ciphertext C and then apply her secret key denoted as S_A in order to decode the message and retrieve the original message M, given that $M = S_A(M)$



It is important to note that S_A and P_A are inverse functions in the sense that Alice is able to compute a given message M from the ciphertext C, and so Assuming that Alice is the only one with the secret key, it is reasonable to assume only Alice is able to accomplish this in a reasonable time. We can look at this key exchange in a little more depth:

1- The first step here is to select at random two large prime numbers *p* and *q* such that $p \neq q$
2- Compute the product of *p* and *q* which we will denote as *n, such that* $n = p * q$
3- We will then select an odd integer *e* relatively prime to $\phi(n)$, which by Euler's phi function $(p-1)(q-1)$
4- We can then compute *d* which is the multiplicative inverse of *e*, mod $\phi(n)$.
5- Next, we can now go ahead and generate two keys, the public key and private key

$P = (e, n)$, which is the RSA Public Key

$S = (d, n)$, which is the RSA Secret Key

6- While keeping S secret, we can transform a message M associated with a give public key P as:

$$P(M) = M^e \ (mod\ n)$$

Homework#2

7- Finally, we can see that the transformation of the ciphertext associated with the secret key is:

$$S(C) = C^4 \ mod \ n$$

Example:

Let us go ahead and take a look at an Example in which Alice wants to send Bob a message. First, they will need to agree on two numbers prime p and generator g.

$$p = 17$$

$$g = 4$$

Since they have now decided on these numbers, Alice can determine a secret number a whereas bob determines a secret number b.

$$a = 3$$

$$b = 6$$

That said, Alice and Bob can now perform the following calculations independently:

$$A = \ g^3 \ mod \ p = 64 \ mod \ 17 = 13$$

$$B = \ g^3 \ mod \ p = 4096 \ mod \ 17 = 16$$

Next, Alice can send her result for A to Bob, and Bob can do the same for Alice. Alice and Bob can now calculate it using their secrets:

$s_A = \ B^a mod \ p = \ 16^3 mod \ 17 = 4096 \ mod \ 17 = 16$

$s_B = \ A^a mod \ p = \ 13^3 mod \ 17 = 4826809 \ mod \ 17 = 16$

Since the two arrived at the same answer, this is the shared secret which Alice and Bob will know, buy given eavesdropper will not.

Sources: Introduction to Algorithms, Second Edition, Chapter 31.

Saleh Alkhalifa
9/19/2022

CS 5200
Module 2

Homework#2

Problem 4: Programming Assignment (20 points)

Consider an array, A, whose contents are integer values. Given a particular threshold value *t*, an event between indices *i*<*j* is a critical event if $a_i > t*a_j$, where *t* is the threshold value given as input from the user.

In this problem, write a full program that outputs the number of critical events for an arbitrary array of integers and any arbitrary threshold value *t.* Array input can be from a file- please provide an example file with the format expected from the code in your submission.

Below is a link to test cases you can use to check your code.

Module2HWTestCases.txt   Download Module2HWTestCases.txt

NOTE: If you see an ambiguity or have any assumptions about the behavior of the expected program behavior, please write them in comments, in your code or other submission documentation.

Programming assignment submission requirements:

A. Submit the code of working program (You can use any programming language).
B. Submit at least one file for input, formatted with an example array.
C. Submit a screen capture showing that your program outputs correct values. You only need to repeat over 2 different arrays running with 2 different thresholds.

Answer:

For this problem I chose to use the Python programming language, specifically using Jupyter notebook. Please find within the attached notebook three functions:

1. **thresholdCriticalEventCounter()** which is the main program based on the specifications above
2. **loadTestCases()** which is a helper function to load the cases from the provided txt file
3. **runTests()** which is a helper function to run the test cases against the first function

thresholdCounter()

This function is based on the specifications above to determine the critical event count for a given an input array A and threshold t. The following screen shot shows the function and two test cases to demonstrate the functionality.

Program with two Examples:

```
[1]: def thresholdCriticalEventCounter(A, t):
         """
         Function that will find the total critical event count.
         A: An Array of integers such as [1 ,5 -7, 0]
         t: Threshold integer such as 3.

         Returns the count
         """
         count = 0
         for i in range(len(A)):
             for j in range(i+1, len(A)):
                 if A[i] > A[j]*t:
                     count = count + 1
         return count
```

```
[2]: array_1 = [1, 1, 1, 1, 1, 1, 1, 1, -1, 1]
     threshold_1 = 1
     thresholdCriticalEventCounter(array_1, threshold_1)
```

```
[2]: 8
```

```
[7]: array_2 = [-1, 5, 100, 45, -12, 11, 80]
     threshold_2 = 2.0
     thresholdCriticalEventCounter(array_2, threshold_2)
```

```
[7]: 7
```

Saleh Alkhalifa
9/19/2022

Homework#2

loadTestCases():

This is a helper function to load the test cases from the txt file. On the **left** you can see the function itself containing the logic of parsing the file. On the **right**, you see the function being called and the test cases being organized.

```
[4]: from ast import literal_eval
```

```
[5]: def loadTestCases(filename):
    """
    Loads a test case from the file into the testCases array.
    Filename: String that reprensents the input file's name
    """
    test_cases = []
    # Load the file
    file = open(filename, mode = 'r', encoding = 'utf-8-sig')
    lines = file.readlines()
    # Ignore the instructions at the top
    lines = lines[5:]
    file.close()
    # Iterate over the file's lines
    for idx, line in enumerate(lines):
        try:
            # Evaluate to determine if its an array
            line = literal_eval(line)
            # If its an array, append arra, threshold, and count to list
            if isinstance(line, list):
                print("Found new test case: ", line)
                tmp_a = line
                tmp_t = float(lines[idx+1])
                tmp_ans = float(lines[idx+2])
                test_cases.append({"array": tmp_a, "threshold":tmp_t, "count":tmp_ans})
        except:
            continue
    # Print the total number of cases found
    print(f"-- Found {len(test_cases)} test cases --")
    return test_cases
```

```
test_cases = loadTestCases('Module2HWTestCases.txt')
```

```
Found new test case:  [1, 1, 1, 1, 1, 1, 1, 1, -1, 1]
Found new test case:  [1, 1, 1, 1, 1, 1, 1, 1, -1, 1]
Found new test case:  [-10, -10, 4]
Found new test case:  [100, 1, 100, 1]
Found new test case:  [-1, 100, 100, 100, 100, 100]
Found new test case:  [-100, 100, 100, 100, 100, 100]
Found new test case:  [16, 13, 2, 45]
Found new test case:  [5, 4, 3, 2, 1]
Found new test case:  [-1, 5, 100, 45, -12, 11, 80]
Found new test case:  [6, 5, 4, 3, 2, 1]
Found new test case:  [-6, 5, 4, 3, 2, 1]
-- Found 11 test cases --
```

runTests()

The last function uses the previous two functions to automatically iterate over the test cases to determine whether the expected and predicted counts match. Given the use of the literal_eval function, it is assumed for this function that the value of 5.0 and 5 are "the same". We can see at the end that all 11 tests were passed.

```
[6]: def runTests(test_cases):
    """
    Iterates over the test cases and determines if success or not
    test_cases: An array of test cases
    """
    count_success = 0
    # Iterate over the cases
    for test in test_cases:
        # Determine the predicted and actual counts
        actual_count = test["count"]
        predicted_count = thresholdCriticalEventCounter(test["array"], test["threshold"])
        # If successful the state it
        # Note that 5.0 and 5 will be accepted as being the same
        if predicted_count == actual_count:
            print(f"{predicted_count} == {actual_count} -> success")
            count_success = count_success + 1
        else:
            print("{predicted_count} != {actual_count} -> failed")
    print("###########################")
    print(f"### {count_success}/{len(test_cases)} Tests Passed! ###")
    print("###########################")

runTests(test_cases)
```

```
8 == 8.0 -> success
36 == 36.0 -> success
0 == 0.0 -> success
6 == 6.0 -> success
15 == 15.0 -> success
10 == 10.0 -> success
6 == 6.0 -> success
10 == 10.0 -> success
7 == 7.0 -> success
3 == 3.0 -> success
2 == 2.0 -> success
###########################
### 11/11 Tests Passed! ###
###########################
```

** Please note that the results above can be easily replicated and verified by installing Jupyter notebook using Python:

```
>> pip install jupyter notebook
>> jupyter notebook
```