CS 58000\_01/02I Design, Analysis and Implementation Algorithms (3 cr.)

Assignment As\_02 (Exam 01)

Student Name: \_\_\_Likhita Budhavaram\_\_\_\_\_\_

This assignment As\_02 is due at 10:30 am, Thursday, October 18, 2022. Please submit your assignment to Brightspace (purdue.brightspace.com). No late turn-in is accepted. Please write your name on the first page of your assignment. Your file name should be your last name such as Ng.docx. Please number your problem-answer clearly such as Problem 1a, 1b, 1c, Problem 2a, 2b, …. The problems’ answers must be arranged in order. Please answer your questions using only a Word file (.docx file only). No pdf file will be accepted. Without using a Word file (.docx file) the submitted problems’ answers would not be graded.

The total number of points for this Assignment\_02 (Exam 01) is 140 points.

Problem 1[30 points]:

In Ch 00\_03, we addressed Figure 1.4 Modular Exponentiation: Given a function modexp(x, y, N) for computing xy mod N, where x, y, and N are integers. We also addressed when k is a power of 2, and a is any integer. We also addressed Fermat’s Little Theorem.

**1a. What is (mod 17)?**

**Solution:**

Must find 4 2^2018(𝑚𝑜𝑑 17), for this,  
 42^2018 can also be written as 22 \* 2^2018

The series can be written as:

4n mod 17  
 4 mod 17 = 4  
 42 mod 17 = 16

43 mod 17 = 13

44 mod 17 = 1

45 mod 17 = 4

46 mod 17 = 16

47 mod 17 = 13

48 mod 17 = 1

The above series we write the mod till we get some series, here the remainders are 4,16,13,1 and the series continue.

The above series can be generalized as this:  
 44N mod 17 = 1,

44N + 2 mod 17 = 16,

44N + 3 mod 17 = 13,

44N + 4 mod 17 = 1 ...eq (1)

Now, with these values we can, determine further 4 2^2018 (𝑚𝑜𝑑 17)

Now, finding the last digit of 22018 we can determine 4 2^2018 (𝑚𝑜𝑑 17)

The last digits of 22018 we can be found using Euler’s totient rule where,

22018 can be written as (210)200 \* 218

⇒ (..24)200 \* 262144 (because 210= 1024) … eq (2)

We know that (..24)even number = 76 ( here the last two digits is 24 and 24 raised to any even number gives the ending digits to be ending with 76)

(..24)odd number = 24( here the last two digits is 24 and 24 raised to any odd number gives the ending digits to be ending with 24)

Therefore from eq (2) we know that 210 is 1024, and 200 is even number, the last two digits will be 76  
(..24)200 ⇒ ....76

Eq. 2 will be ⇒ ...76 \* 262144 = ....456 (this ends with 456).  
The last digit will be ending with 6

with 42^2018, the exponent will be a number ending with 6.

Therefore from eq(1) the generalized series

42^2018 mod 17 ⇒ 4...456 mod 17

...456 is of the series 4N+4.  
 Hence, using eq (1) we can say that 4 2^2018(𝑚𝑜𝑑 17) = 1

**1b. What is (𝑚𝑜𝑑 31)? (Hard Problem)**

**Solution:**

Give 4 2^2006(𝑚𝑜𝑑 31),  
 42^2006 will be written as 22 \* 2^2006 = 22^2006

The series can be written as:

4n mod 31  
 4 mod 31 = 4  
 42 mod 31 = 16

43 mod 31 = 2

44 mod 31 = 8

45 mod 31 = 1

46 mod 31 = 4

47 mod 31 = 16

48 mod 31 = 2

49 mod 31 = 8

410 mod 31 = 1

The above series we write the mod till we get some series, here the remainders are 4,16,2,8,1 and the series continue.

The above series can be generalized as this:

45N mod 31 = 1,

45N + 2 mod 31 = 16,

45N + 3 mod 31 = 2,

45N + 4 mod 31 = 8 ...eq (1)

From the last digit of 22006 we can determine 4 2^2006(𝑚𝑜𝑑 31)

To find the last digits of 22006 we can use Euler’s totient.

22006 can be written as (210)200 \* 26

⇒ (24)200 \* 64 … eq (2)

We know that (..24)even number = 76 ( here the last two digits is 24 and 24 raised to any even number gives the ending digits to be ending with 76)

(..24)odd number = 24( here the last two digits is 24 and 24 raised to any odd number gives the ending digits to be ending with 24)

from eq (2) we know that 210 is 1024, and 200 is even number, the last two digits will be 76  
(..24)200 ⇒ ....76

...76 \* 64 = ....4864 from eq(2)  
The last digit will be 4

Back to 42^2006 the exponent will be a number ending with 4. Therefore we can use eq(1)

42^2006 mod 31 ⇒ 4...4684 mod 31

...4684 is of the series 5N+4.  
 Hence, using eq(1) we can say that 4 2^2006(𝑚𝑜𝑑 31) = 8

**1c. Construct (Design) a polynomial-time algorithm for computing (mod p), where**

**x, y, z, and a prime p.**

**Solution:**

Polynomial time algorithm for computing (mod p) can be done as per the “Fermat’s little theorem”

= 1 mod P

This theorem can be written as

mod P = 1

Now move 1 to left and the above equation becomes

mod P -1 = 0

therefore yz can be written as given below

yz = yz mod (p-1)

Then the expression xy^z  can be given as follows:

xy^z  = xy^z  mod (P – 1) mod P …eq(1)

Now the equation is to be calculated “yz mod (p-1) “

Therefore, the above-mentioned expression is a modular exponentation.

This can be possible in polynomial time

Compute “yz mod (p-1) “ , It has to be repeatedly squared “yz mod (p-1) “

The above equation takes O(n) multiplications.

It is needed to be repeated, So each multiplication will take O(n^2)

Thus the expression mentioned above will take total of O(n^3)

Considering that yz mod (p-1) is A, then eq(1) can be written as

xy^z  = xAmod (p-1)

Repeated squaring algorithm is performed

Therefore running time of this **O(n^2)**

The total running time of the algorithm is **O(n3)**

Problem 2 [60 points]:

This problem is an exercise using the formalization of the RSA public-key cryptosystem. For solving the problems, you are required to use the following formalization of the RSA public-key cryptosystem.

Given the following formalization of the RSA public-key cryptosystem, each participant creates their public key (n, g) where a is a small prime number, and n is the product of two large primes, p and q. However, the two large primes p and q are secret keys.

1. Select two very large prime numbers p and q. The number of bits needed to represent p and q might be 1024.
2. Compute

n = pq

(n) = (p – 1) (q – 1).

The formula for (n) is owing to the Theorem: The number of elements in is given by Euler’s totient function, which is

where the product is over all primes that divide n, including n if n is prime.

1. Choose a small prime number as an encryption component g, that is relatively prime to (n). That means,

gcd(g, (n) ) = 1, i.e.,

gcd(g, (p-1)(q-1)) = 1.

1. Compute the multiplicative inverse That is,

The inverse exists and is unique.

That is, the decryption component h = g-1 mod (n).

1. Let pkey = (n, g) be the public key, and skey = (p, q, h) be the secret key.

* For any message M mod n, the encryption of M is C = Mg mod n.
* The decryption of C is M = Ch mod n.

End of the formalization of the RSA public-key cryptosystem.

Use the RSA Cryptosystem formalism for solving problem 2.

Given g = 59, p = 991 and q = 997.

**2a. Show that the given values of g, p, and q are prime.**

**Solution:**

To check if the number is prime divide it by numbers like. 2,3,5,7,,…

If it completely divides or remainder is zero then it is not a prime number.

Another method is HCF factors,

Another method is fermant’s theorem

g=59

using Fermat’s theorem,

1. a=2

258mod 59

=((22mod 59)\*( 28mod 59) (216mod 59) (232mod 59)mod 59

=(4\*20\*26\*27)mod 59

=1

1. a=3

358mod 59

=((32mod 59)\*( 38mod 59) (216mod 59) (232mod 59)mod 59

=(4\*20\*26\*27)mod 59

=1

From above 59 is a prime number

P=991

1. a=2

2990mod 991

=((22mod 991)\*( 24mod 991) (28mod 991) (216mod 991) (264mod 991) (2128mod 991) (2256mod 991) (2512mod 991)mod 991

=1

1. a=3

3990mod 991

=((32mod 991)\*( 34mod 991) (38mod 991) (316mod 991) (364mod 991) (3128mod 991) (3256mod 991) (3512mod 991)mod 991

=(4\*20\*26\*27)mod 59

=1

From above 59 is a prime number

q=997

1. a=2

2996mod 997

=((22mod 997)\*( 24mod 997) (28mod 997) (216mod 997) (264mod 997) (2128mod 997) (2256mod 997) (2512mod 997)mod 997

=1

1. a=3

3996mod 997

=((32mod 997)\*( 34mod 997) (38mod 997) (316mod 997) (364mod 997) (3128mod 997) (3256mod 997) (3512mod 997)mod 997

=1

From above we can see that all numbers are prime numbers.

**2b. Compute n = pq and (n) = (p – 1) (q – 1).**

**Solution:**

Given are the values from the data above:

g=59, p=991 and q=997

**n = pq**

n = 991\*997

**n = 988027**

**(n) = (p – 1) (q – 1)**

**=**(991-1)(997-1)

**(n) = 986040**

**2c. Given a plaintext M = 5065747269, what is the encryption of M, using C = Mg mod n. Show in detail how you derive C, which is the ciphertext of the plaintext M.**

**Solution:**

Encryption of M, C = Mg mod n

Here, M = plain text

C= cipher text

g = public key

n = pq which is 988027

therefore the equation is given as C = Mg mod n and on substitution we get the following:

C= (5065747269)^59 mod 988027

Here g is 59 which can be written as 32+16+8+2+1

=5065747269­ 32+16+8+2+1 mod 988027

=5065747269­32 mod 988027 =140485

=5065747269­16 mod 988027 =361242

=50657472698 mod­ 988027=144736

=5065747269­4  mod 988027=361242

=5065747269­2  mod 988027=140485

From the above results multiple and find mod to find C

=(140485 \* 361242 \* 144736 \* 361242 \* 140485 ) mod 988027

**C = 433940**

2d. **Compute the multiplicative inverse That is, the decryption component h = g-1 mod (n).**

[Hints: Compute a GCD as a Linear Combination. Then, find an Inverse Modulo n. In other words, you can apply the extended Euclid algorithm to find the linear combination of g and Then find a positive inverse of g mod ]

**Solution:**

g=59, p=991, q =996

h = g-1 mod (n)

To find multiplicative inverse, we have to use Euclid algorithm which is as follows:

GCD(59,986040)

=986040=16712\*59 + 32 …eq(1)

=59=1\*32+27 …eq(2)

=32=1\*27+5 …eq(3)

=27= 5\*5+2. …eq(4)

=5=2\*2+1 …eq(5)

=2=2\*1+0

Therefore, GCD is 1

Now solve recursively for linear combination:

* Now solving equations we get value of h:

1= 5- 2\*2. …from eq(5)

=5-2\*(27 -5\*5)

=11\*5 -27\*2

=11(32-27)-2\*27 …from eq(3)

=11\*32-13\*27

=11\*32-13\*(59 -32)

=24\*32-13\*59

= 24(986040-16712\*59)-13\*59

=24\*986040-401101\*59

=24\*986040+(-401101)\*59

From above we can write

24\*986040+(-401101)\*59=1

Taking mod on both sides

0+(-401101 mod 986040)\*59 mod 986040 = 1 mod 98604

=584930 \*59 mod 986040 =1[-401101 mod 986040]

=584939 mod 986040

59=584939-1 mod 986040((n))

From the above step

**H=584939** is multiplicative of g-1 mod (n)

To verify this h, we have h,g (n) values and the equation

=g\*h mod (n)

=59\*584939 mod 986040

=34511401 mod 986040

**=1**

**2e. From problem 2d, what is your secret key (p, q, h)?**

**Solution:**

secret key (p, q, h)

secret key (991, 997, **584939**)?

**2f. What is the decryption of C using M = Ch mod n? Show in detail how you derive M, which is the plaintext M of the ciphertext C.**

To get a message M = Ch mod n,

M= message

C=cipher text

N=pq

h=private secret key

c is given as c=mgmod n

now on substitution:

chmod n=mg^hmod n

by Euler’s totient,

mg^hmod n= mgh mod (n) mod n

=M mod n (gh are multiplicative inverse)

Now using secret key value we can decrypt:

H=584939, c= 433940 n=988027

M= chmod n

=433940584939mod 988027

=4339402\*2924679 \* 433940 mod 988027

**=**797805292469. \* 433940 mod 988027

=7978052\*146234 \* 797805 \*433940 mod 988027

=884490146234. \* 797805 \*433940 mod 988027

=88449017\*11\*782. \* 797805 \*433940 mod 988027

=1322840

2g (Bonus)[5 points]:

What is the message (in terms of the alphabet)?

Problem 3[30 points]:

Assume that we define

h1(k) = k mod 13, and

h2(k) = 1 + (k mod 11).

For the open addressing, consider the following methods

**Linear Probing**

Given an ordinary hash function h: U {0, 1, 2, …, m-1}, the method of *linear probing* uses the hash function

h(k, i) = (h1(k) + i) mod m for i = 0, 1, 2, …, m-1.

**Quadratic Probing**

Uses a hashing function of the form

h(k, i) = (h1(k) + c1i + c2i2 ) mod m,

where h1 is an auxiliary hash function, c1 and c2 0 are auxiliary constants c1 3 c2 = 5,

and i = 0, 1, 2, …, m-1.

**Double hashing**

Uses a hashing function of the form

h(k, i) = (h1(k) + i h2(k) ) mod m,

where h1 and h2 are auxiliary hash functions.

The value of h2(k) must never be zero and should be relatively prime to m for the sequence to include all possible addresses.

Given K = {79, 69, 98, 72, 14, 50, 88, 99, 78, 65} and the size of a table is 13, with indices counting from 0, 1, 2, …, 12. Store the given K in a table with the size 13 counting the indices from 0, 1, 2, …, 12. Show the resultant table with 10 given keys for each method applied:

**3a. if linear probing is employed.**

The Resultant Table with 10 given keys is: Complete the table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 78 | 79 | 14 | 65 | 69 |  |  | 98 | 72 | 99 | 88 | 50 |  |

h(k, i) = (h1(k) + i) mod m

K = {79, 69, 98, 72, 14, 50, 88, 99, 78, 65}

for i = 0, 1, 2, …, m-1

m=13

1. H(79,0)=( h1(79)+0) mod 13

=(79 mod 13 + 0) mod 13

=1 mod 13

=1

1. H(69,0)=( h1(69)+0) mod 13

=(69 mod 13 + 0) mod 13

=4 mod 13

=4

1. H(98,0)=( h1(98)+0) mod 13

=(98 mod 13 + 0) mod 13

=7 mod 13

=7

1. H(72,0)=( h1(72)+0) mod 13

=(72 mod 13 + 0) mod 13

=7 mod 13

=7

There is a collision over here so increment i value

H(72,1)=( h1(72)+1) mod 13

=(72 mod 13 + 1) mod 13

=8 mod 13

=8

1. H(14,0)=( h1(14)+0) mod 13

=(14 mod 13 + 0) mod 13

=1 mod 13

=1

There is collision so increment i value

H(14,1)=( h1(14)+1) mod 13

=(14 mod 13 + 1) mod 13

=2 mod 13

=2

1. H(50,0)=( h1(50)+0) mod 13

=(50 mod 13 + 0) mod 13

=11 mod 13

=11

1. H(88,0)=( h1(88)+0) mod 13

=(88 mod 13 + 0) mod 13

=10 mod 13

=10

1. H(99,0)=( h1(99)+0) mod 13

=(99 mod 13 + 0) mod 13

=7 mod 13

=7

1. H(78,0)=( h1(978)+0) mod 13

=(78 mod 13 + 0) mod 13

=0 mod 13

=0

1. H(65,0)=( h1(65)+0) mod 13

=(65mod 13 + 0) mod 13

=0 mod 13

=0

There is a collision so increment i value

H(65,1)=( h1(65)+1) mod 13

=(65 mod 13 + 1) mod 13

=1 mod 13

=1

There is a collision again increment i value

H(65,2)=( h1(65)+2) mod 13

=(65 mod 13 + 2) mod 13

=2 mod 13

=2

There is a collision again increment i value

H(65,3)=( h1(65)+3) mod 13

=(65 mod 13 + 3) mod 13

=3 mod 13

=3

**3b. if quadratic probing is employed.**

The Resultant Table with 10 given keys is: Complete the table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 78 | 79 | 72 | 65 | 69 |  |  | 98 | 99 | 14 | 88 | 50 |  |

h(k, i) = (h1(k) + c1i + c2i2 ) mod m

K = {79, 69, 98, 72, 14, 50, 88, 99, 78, 65}

for i = 0, 1, 2, …, m-1

m=13

1. h(79, 0) = (h1(79) + (3)(0) + (5)(0)2 ) mod 13

=(79 mod 13 + 0 + 0) mod 13

= 1 mod 13

= 1

1. h(69, 0) = (h1(69) + (3)(0) + (5)(0)2 ) mod 13

=(69 mod 13 + 0 + 0) mod 13

= 4 mod 13

= 4

1. h(98, 0) = (h1(98) + (3)(0) + (5)(0)2 ) mod 13

=(98 mod 13 + 0 + 0) mod 13

= 7 mod 13

= 7

1. h(72, 0) = (h1(72) + (3)(0) + (5)(0)2 ) mod 13

=(72 mod 13 + 0 + 0) mod 13

= 7

There is a collision here so increment i

h(72, 0) = (h1(72) + (3)(1) + (5)(1)2 ) mod 13

=(79 mod 13 + 3 + 5) mod 13

=15 mod 13

=2

1. h(14, 0) = (h1(14) + (3)(0) + (5)(0)2 ) mod 13

=(79 mod 13 + 0 + 0) mod 13

= 1 mod 13

=1

There is a collision here so increment i

h(14, 1) = (h1(14) + (3)(1) + (5)(1)2 ) mod 13

=(79 mod 13 + 03+ 5) mod 13

= 9 mod 13

=9

1. h(50, 0) = (h1(50) + (3)(0) + (5)(0)2 ) mod 13

=(50 mod 13 + 0 + 0) mod 13

=11 mod 13

=11

1. h(88, 0) = (h1(88) + (3)(0) + (5)(0)2 ) mod 13

=(88 mod 13 + 0 + 0) mod 13

=10 mod 13

=10

1. h(99, 0) = (h1(50) + (3)(0) + (5)(0)2 ) mod 13

=(99 mod 13 + 0 + 0) mod 13

=8 mod 13

=8

1. h(78, 0) = (h1(78) + (3)(0) + (5)(0)2 ) mod 13

=(78mod 13 + 0 + 0) mod 13

=0 mod 13

=0

1. h(65, 0) = (h1(65) + (3)(0) + (5)(0)2 ) mod 13

=(65 mod 13 + 0 + 0) mod 13

=0 mod 13

=0

There is a collision here so increment i

h(65, 1) = (h1(65) + (3)(1) + (5)(1)2 ) mod 13

=(65 mod 13 + 3 + 5) mod 13

=8 mod 13

=8

There is a collision here so increment i

h(65, 2) = (h1(65) + (3)(2) + (5)(2)2 ) mod 13

=(65 mod 13 + 6 + 20) mod 13

=26 mod 13

=0

There is a collision here so increment i

h(65, 3) = (h1(65) + (3)(3) + (5)(3)2 ) mod 13

=(65 mod 13 + 9 + 45) mod 13

=54 mod 13

=2

There is a collision here so increment i

h(65, 4) = (h1(65) + (3)(4) + (5)(4)2 ) mod 13

=(65 mod 13 + 12 + 80) mod 13

=92 mod 13

=1

There is a collision here so increment i

h(65, 5) = (h1(65) + (3)(5) + (5)(5)2 ) mod 13

=(65 mod 13 + 15+ 125) mod 13

=140 mod 13

= 10

There is a collision here so increment i

h(65, 6) = (h1(65) + (3)(6) + (5)(6)2 ) mod 13

=(65 mod 13 + 18 + 180) mod 13

=198 mod 13

=3

**3c. if double hashing is employed.**

The Resultant Table with 10 given keys is: Complete the table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 78 | 79 |  | 65 | 69 | 14 |  | 98 | 72 | 99 | 88 | 50 |  |

h(k, i) = (h1(k) + i h2(k) ) mod m,

h1(k) = k mod 13, and

h2(k) = 1 + (k mod 11).

K = {79, 69, 98, 72, 14, 50, 88, 99, 78, 65}

for i = 0, 1, 2, …, m-1

m=13

1. h(79, 0) = (h1(79) + (0)h2(79) ) mod m,

= ( 79 mod 13 + 0) mod 13

= 1 mod 13

= 1

1. h(69, 0) = (h1(69) + (0)h2(69) ) mod m,

= ( 69 mod 13 + 0) mod 13

= 4 mod 13

= 4

1. h(98, 0) = (h1(98) + (0)h2(98) ) mod m,

= ( 98 mod 13 + 0) mod 13

= 7 mod 13

= 7

1. h(72, 0) = (h1(72) + (0)h2(72) ) mod m,

= ( 72 mod 13 + 0) mod 13

= 7 mod 13

= 7

There is collision so increase the i value

h(72, 1) = (h1(72) + (1)h2(72) ) mod m,

= ( 72 mod 13 + 1(1+ 72 mod 11) mod 13

= (7 +7)mod 13

= 14 mod 13

= 1

There is a collision again, hence increment

h(72, 2) = (h1(72) + (2)h2(72) ) mod m,

= ( 72 mod 13 + 2(2+ 72 mod 11)) mod 13

= (7 + 14) mod 13

= 21 mod 13

= 8

1. h(14, 0) = (h1(14) + (0)h2(14) ) mod m,

= ( 14 mod 13 + 1(1+ 14 mod 11)) mod 13

=(1+0)mod13

= 1

There is collision so increase the i value

h(14, 1) = (h1(14) + (1)h2(14) ) mod m,

= ( 14 mod 13 + 1(1+ 14 mod 11)) mod 13

= (1+4)mod13

= 5

1. h(50, 0) = (h1(50) + (0)h2(50) ) mod m,

= ( 50 mod 13 + 0(1+ 50 mod 11)) mod 13

= 11mod13

= 11

1. h(88, 0) = (h1(88) + (0)h2(88) ) mod m,

= ( 88 mod 13 + 0(1+ 88 mod 11)) mod 13

= 10

1. h(99, 0) = (h1(99) + (0)h2(99) ) mod m,

= ( 99 mod 13 + 0(1+ 99 mod 11)) mod 13

= 8 mod 13

=8

There is collision so increase the i value

h(99, 1) = (h1(99) + (1)h2(99) ) mod m,

= ( 99 mod 13 + 1(1+ 99 mod 11)) mod 13

= 9 mod 13

=9

1. h(78, 0) = (h1(78) + (0)h2(78) ) mod m,

= ( 78 mod 13 + 0(1+ 78 mod 11)) mod 13

= 0 mod 13

=0

1. h(65, 0) = (h1(65) + (0)h2(65) ) mod m,

= ( 65 mod 13 + 0(1+65 mod 11)) mod 13

= 0 mod 13

=0

There is collision so increase the i value

h(65, 1) = (h1(65) + (1)h2(65) ) mod m,

= ( 65 mod 13 + 1(1+65 mod 11)) mod 13

= (0 +1(1+10))mod 13

=11

There is collision so increase the i value

h(65, 2) = (h1(65) + (2)h2(65) ) mod m,

= ( 65 mod 13 + 2(1+65 mod 11)) mod 13

= (0 +2(1+10))mod 13

=9

There is collision so increase the i value

h(65, 3) = (h1(65) + (3)h2(65) ) mod m,

= ( 65 mod 13 + 3(1+65 mod 11)) mod 13

= (0 +3(1+10))mod 13

=7

There is collision so increase the i value

h(65, 4) = (h1(65) + (4)h2(65) ) mod m,

= ( 65 mod 13 + 4(1+65 mod 11)) mod 13

= (0 +4(1+10))mod 13

=5

There is collision so increase the i value

h(65, 5) = (h1(65) + (1)h2(65) ) mod m,

= ( 65 mod 13 + 5(1+65 mod 11)) mod 13

= (0 +5(1+10))mod 13

=3

Problem 4 [20 points]:

For the division method for creating hash functions, map a key k into one of the m slots by taking the remainder of k divided by m. The hash function is:

h(k) = k mod m,

where m should not be a power of 2.

For the multiplication method for creating hash functions, the hash function is

h(k) = └ m(kA –└ k A ┘) ┘ = └ m(k A mod 1) ┘

where “k A mod 1” means the fractional part of k A and a constant A in the range

0 < A < 1.

An advantage of the multiplication method is that the value of m is not critical.

Choose m = 2p for some integer p.

Give your explanations for the following questions:

4a. Why m should not be a power of 2 in the division method for creating a hash

function?

**Solution:**

This is because if m = 2^p, h becomes just the p lowest-order bits of k. Usually we choose m to be a prime number not too close to a power of 2.

It is good choose a prime not close to power of 2.

The hash function might not distribute the values evenly if m is not prime.

It is always advice to positive value for k and prime array size m.

All the bits of m play a significant role.

Also we have to design the hash function that depends on all bits and all patterns are equally likely.

As a result of this the keys are evenly distributed between 0 and m-1.

4b. Why m = 2p, for some integer p, could be (and in fact, favorably) used in the

multiplication method?

**Solution:** Hash function is defined as :

H(k)=k mod m

In division method of creating hash functions, we map a key k to one of m slots by taking the remainder of k mod m, which is as defined in the equation.

Eg: if m =12, k=90 then the hash can be written as h(k)=6. This is fast as it has single division operation.

We avoid certain m values in this division method, m cant be power of 2.

If m=20 then h(k)is just the p lowest order bits of k.

It is better to make the hash function depend on all bits of k, unless It is known that all low order p bit patterns are equally likely.

Permuting the characters of k does not change final answer.

For a number to be good selection for m is when it is a prime not to an exact power of 2.

**Note: If you provide your answer in your handwriting, good handwriting is required.**

**Proper numbering of your answer to each problem is strictly required. The problem’s solution must be orderly given. (10 points off if not)**