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

Assignment As\_02 (Exam 01)

Student Name: **Harsh Sharma**

This assignment As\_02 is due at 11:59 p.m., Sunday, October 1, 2023. 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 NgP\_As02.docx. Please number your problem-answer clearly such as Problem I.1.a, I.1.b, I.1.c, I.2, …, I.7, Problem II.1, II.2, II.3, II.4. The problems’ answers must be arranged according to the order of the given problem. 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 150 points.

Problem I [110 points]:

This problem is an exercise using the formalization of the RSA public-key cryptosystem. To solve 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 I.

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

I.1. [30 pts.] Show that the given values of g, p, and q are prime,

I.1.a Use the Algorithm Sieve (the Sieve of Eratosthenes Method) to check whether p is a prime.

Solution:

In order to check whether p, 991, is prime or not using Sieve of Eratosthenes Method we need to have a list of integers from 2 to 991. Then need to remove all the multiples of numbers from 2 to square root of 991, i.e., till 32 (upper limit).

List of integers which are multiple of 2 and hence to be removed: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128, 130, 132, 134, 136, 138, 140, 142, 144, 146, 148, 150, 152, 154, 156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176, 178, 180, 182, 184, 186, 188, 190, 192, 194, 196, 198, 200, 202, 204, 206, 208, 210, 212, 214, 216, 218, 220, 222, 224, 226, 228, 230, 232, 234, 236, 238, 240, 242, 244, 246, 248, 250, 252, 254, 256, 258, 260, 262, 264, 266, 268, 270, 272, 274, 276, 278, 280, 282, 284, 286, 288, 290, 292, 294, 296, 298, 300, 302, 304, 306, 308, 310, 312, 314, 316, 318, 320, 322, 324, 326, 328, 330, 332, 334, 336, 338, 340, 342, 344, 346, 348, 350, 352, 354, 356, 358, 360, 362, 364, 366, 368, 370, 372, 374, 376, 378, 380, 382, 384, 386, 388, 390, 392, 394, 396, 398, 400, 402, 404, 406, 408, 410, 412, 414, 416, 418, 420, 422, 424, 426, 428, 430, 432, 434, 436, 438, 440, 442, 444, 446, 448, 450, 452, 454, 456, 458, 460, 462, 464, 466, 468, 470, 472, 474, 476, 478, 480, 482, 484, 486, 488, 490, 492, 494, 496, 498, 500, 502, 504, 506, 508, 510, 512, 514, 516, 518, 520, 522, 524, 526, 528, 530, 532, 534, 536, 538, 540, 542, 544, 546, 548, 550, 552, 554, 556, 558, 560, 562, 564, 566, 568, 570, 572, 574, 576, 578, 580, 582, 584, 586, 588, 590, 592, 594, 596, 598, 600, 602, 604, 606, 608, 610, 612, 614, 616, 618, 620, 622, 624, 626, 628, 630, 632, 634, 636, 638, 640, 642, 644, 646, 648, 650, 652, 654, 656, 658, 660, 662, 664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684, 686, 688, 690, 692, 694, 696, 698, 700, 702, 704, 706, 708, 710, 712, 714, 716, 718, 720, 722, 724, 726, 728, 730, 732, 734, 736, 738, 740, 742, 744, 746, 748, 750, 752, 754, 756, 758, 760, 762, 764, 766, 768, 770, 772, 774, 776, 778, 780, 782, 784, 786, 788, 790, 792, 794, 796, 798, 800, 802, 804, 806, 808, 810, 812, 814, 816, 818, 820, 822, 824, 826, 828, 830, 832, 834, 836, 838, 840, 842, 844, 846, 848, 850, 852, 854, 856, 858, 860, 862, 864, 866, 868, 870, 872, 874, 876, 878, 880, 882, 884, 886, 888, 890, 892, 894, 896, 898, 900, 902, 904, 906, 908, 910, 912, 914, 916, 918, 920, 922, 924, 926, 928, 930, 932, 934, 936, 938, 940, 942, 944, 946, 948, 950, 952, 954, 956, 958, 960, 962, 964, 966, 968, 970, 972, 974, 976, 978, 980, 982, 984, 986, 988, 990]

List of integers which are multiple of 3 and hence to be removed: [3, 9, 15, 21, 27, 33, 39, 45, 51, 57, 63, 69, 75, 81, 87, 93, 99, 105, 111, 117, 123, 129, 135, 141, 147, 153, 159, 165, 171, 177, 183, 189, 195, 201, 207, 213, 219, 225, 231, 237, 243, 249, 255, 261, 267, 273, 279, 285, 291, 297, 303, 309, 315, 321, 327, 333, 339, 345, 351, 357, 363, 369, 375, 381, 387, 393, 399, 405, 411, 417, 423, 429, 435, 441, 447, 453, 459, 465, 471, 477, 483, 489, 495, 501, 507, 513, 519, 525, 531, 537, 543, 549, 555, 561, 567, 573, 579, 585, 591, 597, 603, 609, 615, 621, 627, 633, 639, 645, 651, 657, 663, 669, 675, 681, 687, 693, 699, 705, 711, 717, 723, 729, 735, 741, 747, 753, 759, 765, 771, 777, 783, 789, 795, 801, 807, 813, 819, 825, 831, 837, 843, 849, 855, 861, 867, 873, 879, 885, 891, 897, 903, 909, 915, 921, 927, 933, 939, 945, 951, 957, 963, 969, 975, 981, 987]

List of integers which are multiple of 4 and hence to be removed: [] (all multiples of 4 already removed by 2.)

List of integers which are multiple of 5 and hence to be removed: [5, 25, 35, 55, 65, 85, 95, 115, 125, 145, 155, 175, 185, 205, 215, 235, 245, 265, 275, 295, 305, 325, 335, 355, 365, 385, 395, 415, 425, 445, 455, 475, 485, 505, 515, 535, 545, 565, 575, 595, 605, 625, 635, 655, 665, 685, 695, 715, 725, 745, 755, 775, 785, 805, 815, 835, 845, 865, 875, 895, 905, 925, 935, 955, 965, 985]

List of integers which are multiple of 6 and hence to be removed: [] (all multiples of 6 already removed by 2 and 3.)

List of integers which are multiple of 7 and hence to be removed: [7, 49, 77, 91, 119, 133, 161, 203, 217, 259, 287, 301, 329, 343, 371, 413, 427, 469, 497, 511, 539, 553, 581, 623, 637, 679, 707, 721, 749, 763, 791, 833, 847, 889, 917, 931, 959, 973]

List of integers which are multiple of 8 and hence to be removed: []

List of integers which are multiple of 9 and hence to be removed: []

List of integers which are multiple of 10 and hence to be removed: []

List of integers which are multiple of 11 and hence to be removed: [11, 121, 143, 187, 209, 253, 319, 341, 407, 451, 473, 517, 583, 649, 671, 737, 781, 803, 869, 913, 979]

List of integers which are multiple of 12 and hence to be removed: []

List of integers which are multiple of 13 and hence to be removed: [13, 169, 221, 247, 299, 377, 403, 481, 533, 559, 611, 689, 767, 793, 871, 923, 949]

List of integers which are multiple of 14 and hence to be removed: []

List of integers which are multiple of 15 and hence to be removed: []

List of integers which are multiple of 16 and hence to be removed: []

List of integers which are multiple of 17 and hence to be removed: [17, 289, 323, 391, 493, 527, 629, 697, 731, 799, 901]

List of integers which are multiple of 18 and hence to be removed: []

List of integers which are multiple of 19 and hence to be removed: [19, 361, 437, 551, 589, 703, 779, 817, 893]

List of integers which are multiple of 20 and hence to be removed: []

List of integers which are multiple of 21 and hence to be removed: []

List of integers which are multiple of 22 and hence to be removed: []

List of integers which are multiple of 23 and hence to be removed: [23, 529, 667, 713, 851, 943, 989]

List of integers which are multiple of 24 and hence to be removed: []

List of integers which are multiple of 25 and hence to be removed: []

List of integers which are multiple of 26 and hence to be removed: []

List of integers which are multiple of 27 and hence to be removed: []

List of integers which are multiple of 28 and hence to be removed: []

List of integers which are multiple of 29 and hence to be removed: [29, 841, 899]

List of integers which are multiple of 30 and hence to be removed: []

List of integers which are multiple of 31 and hence to be removed: [31, 961]

List of integers which are multiple of 32 and hence to be removed: []

After removing all the multiples from the list of integers, we are left with this list: [37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991]

As per Sieve of Eratosthenes Method, this list is a list of all prime numbers. And since 991 is in the list, 991 must be a prime number.

**Note**: To implement the above Sieve of Eratosthenes Method, I wrote a python code. But since no code was asked, I’m only attaching a screenshot of the same:

A screenshot of a computer

Description automatically generated

I.1.b Based on Fermat’s Little Theorem, the algorithm in Figure 1.8 can be used to check whether q is a prime. What is the most reasonable set {a1, a2, …, ak} that is used for applying this algorithm?

Solution:

There are certain conditions which the set of **a** must follow.

1. **a** must be co-prime with **p**, i.e., 997 in our case. a ≢ 0(mod 997).
2. Also, we need to p calculate ap−1≡1(mod). And therefore, we start with smaller values. Typically starting from 2.

Therefore, the most reasonable set would be {2,3,5,7,11…, 991}

Let’s calculate ap−1≡1(mod) for some of the initial values.

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 the above calculations we can say that typically the value 997 could be prime.

I.1.c. How do you check that g is a prime? Show the work of how you compute.

Solution: I’m using Sieve of Eratosthenes Method to check whether g, 59, is prime or not. For this I need to create a list of integers from 2 to 59, i.e., integers = {2, 3, 4…, 59}

Now we need to remove all the multiples of primes from 2 to square root of 59, i.e., till 8 (taking upper limit).

List of integers which are multiple of 2 and hence to be removed: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58]

List of integers which are multiple of 3 and hence to be removed: [3, 9, 15, 21, 27, 33, 39, 45, 51, 57]

List of integers which are multiple of 4 and hence to be removed: [] (all the multiples of 4 already removed by 2).

List of integers which are multiple of 5 and hence to be removed: [5, 25, 35, 55]

List of integers which are multiple of 6 and hence to be removed: []

List of integers which are multiple of 7 and hence to be removed: [7, 49]

List of integers which are multiple of 8 and hence to be removed: []

The above process is presented in form of table below:

A grid of white squares

Description automatically generated

A grid of white squares

Description automatically generated

A white sheet with black and red dots

Description automatically generated with medium confidence

List of integers we left with are: [11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59]. As per Sieve of Eratosthenes Method this list of integers is all prime. Therefore, 59 is a prime number.

I.2.[10 pts.] Compute n = pq and (n) = (p – 1) (q – 1).

Solution:

To calculate the value of n we need to multiply p and q. We have the value of p and q as,

p = 991

and, q = 997

Therefore, n = p \* q

= 991 \* 997

= 988,027

Now, to calculate (n) we need to multiply the value of (p-1) with (q-1). So, let’s first find those values.

(p-1) = 991 – 1 = 990

(q-1) = 997 – 1 = 996

Therefore, (n) = (p – 1) (q – 1)

= 990 \* 996

= 986,040

I.3.[20 pts.] Given a plaintext **M = 506574**, 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:

To calculate C, we need to calculate Mg mod n.

We have M as the message which is **506574.**

g has the value of 59 from the question.

And we have calculated the value of n which is 988,027.

Therefore, on substituting the value in the above equation we have **50657459 mod 988,027**.

Now, obviously we can’t calculate that value easily. We need to apply some technique to simplify our calculations. We need to break the calculations into simple steps. And we have already learnt this in previous lectures. We can break 59 into power of two’s.

Let’s break down 59 to power of two’s = 32 + 16 + 8+ 2+ 1

= 25 + 24 + 23 + 21 + 20

So the value of k are = 5,4,3,1,0

We can easily calculate 5065741mod 988,027 = 506574

5065742mod 988,027 = 916874

5065744mod 988,027 = (5065742 )2 mod 988,027 = 9168742 mod 988,027 = 99061

5065748mod 988,027 = (5065744 )2 mod 988,027 = 990612 mod 988,027 = 985584

50657416mod 988,027 = (5065748 )2 mod 988,027 = 9855842 mod 988,027 = 40087

50657432mod 988,027 = (50657416 )2 mod 988,027 = 400872 mod 988,027 = 435667

Now, **50657459** can be written as = 50657432 \*50657416 \* 5065748 \* 5065742 \* 5065741

Therefore, **50657459 mod 988,027** is equal to,

((50657432 mod 988,027) \*(50657416 mod 988,027) \* (5065748 mod 988,027) \* (5065742 mod 988,027) \* (5065741 mod 988,027)) mod 988,027

= (435667 \* 40087 \* 985584 \* 916874 \* 506574) mod 988,027

= 661578

Therefore, the cipher text, C, is **661578**.

I.4.[20 pts.] 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:

So, we need to find the multiplicative inverse which is equal to h = g-1 mod (n),

On substituting the value, we get h = 59-1 mod 986040

Now we need to check if the GCD of 59 and 986040 is 1 in order to apply the extended Euclid algorithm. Therefore,

GCD (986040, 59) = GCD(59, 986040 mod 59)

= GCD (59, 32)

= GCD (32, 59 mod 32)

= GCD (32, 27)

= GCD (27, 32mod 27)

= GCD (27, 5)

= GCD (5, 27 mod 5)

= GCD (5, 2)

= GCD (2, 5 mod 2)

= GCD (2, 1)

= GCD (1, 2 mod 1)

= GCD (1, 0)

Since 1 is the common divisor, we can apply extended Euclid algorithm.

Now, h = 59-1 mod 986040 can be written as 59 \* x + 986040 \* y = 1. We need to find the coefficient x which will be our multiplicative inverse.

Therefore,

986040 = 59 (16712) + 32, implies 32 = 986040 - 59 (16712)

59 = 32(1) + 27, implies 27 = 59 – 32(1)

32 = 27(1) + 5, implies 5 = 32 – 27(1)

27 = 5(5) + 2, implies 2 = 27 – 5(5)

5 = 2(2) + 1, implies 1 = 5 – 2(2)

Now, substituting the values from above equations.

1 = 5 – 2(2)

1 = 5 – [27 – 5(5)](2)

1 = 5(11) – 27(2)

1 = [32 – 27(1)](11) – 27(2)

1 = 32(11) – 27(13)

1 = 32(11) – [59 – 32(1)](13)

1 = 32(24) – 59(13)

1 = [986040 - 59 (16712)](24) – 59(13)

1 = 986040(24) – 59(401101)

Now, we have the equation in the **form of g and .** Taking **mod 986040** on both sides.

1 mod 986040 = 986040(24) mod 986040 + 59(-401101) mod 986040

1 mod 986040 = 0 + 59(-401101) mod 986040

Rewriting the equation:

59(-401101) mod 986040 = 1 mod 986040

(-401101 mod 986040) \* 59 mod 986040 = 1 mod 986040

584939 \* 59 mod 986040 = 1 mod 986040 [Since, 986040 – 401101 = 584939]

On further simplification,

584939 = 1 mod 986040 / 59 mod 986040

584939 = (1/59) mod 986040

584939 = 59-1 mod 986040

The above equation is of the form h = g-1 mod (n), and so **h = 584939**

Therefore, **the multiplicative inverse of 59-1 mod 986040 is 584939**.

We can also verify the answer by putting the values in the equation and check if it satisfies the condition.

= g\*h mod φ(n) [This value must be equal to 1 if **h** is the multiplicative inverse.]

= 59 \* 584939 mod 986040

= 34511401 mod 986040

= 1

And hence verified. H = 584939 is the multiplicative inverse.

I.5.[10 pts.] From problem I.4, what is your secret key (p, q, h)?

**Solution**:

In RSA algorithm, the full secret key is the triplet of (p, q, h). We already know the value of p and q from the question which are 991 and 997 respectively.

Here h is the multiplicative inverse of g-1 mod φ(n). In our case, we have already calculated multiplicative inverse of 59-1 mod 986040 which is equal to 584939.

Therefore, **the secret key is (991, 997, 584939)**

I.6.[20 pts.] 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.

**Solution**:

To get a message M = Ch mod n,

M= message

C=cipher text

N=pq

h=private secret key

C is given as C=Mg mod n

now on substitution:

Chmod n=Mg^hmod n

By using **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 = 661578, n=988027

M= Chmod n

Therefore, to decrypt the cipher text we need to calculate Ch mod n.

i.e., 661578584939 mod 988027

Now, obviously we can’t calculate that value easily. We need to apply some technique to simplify our calculations. We need to break the calculations into simple steps. And we have already learnt this in previous lectures. We can break 584939 into power of two’s.

Therefore,

584939 = 524,288 + 32,768 + 8,192 + 4,096 + 1024 + 512 + 64 + 32 + 8 + 1

584939 = 219 + 215 + 214 + 213 + 212 + 210 +29 + 26 + 25 +23 + 20

Now, **661578584939** can be written as = 661578524,288 + 66157832768 + 6615788192 + 6615784096 + 6615781024 + 661578512 + 66157864 + 66157832 + 6615788 + 6615781

We can easily calculate 6615781 mod 988027, and 6615782 mod 988027 and 6615784 mod 988027… similarly up to 661578524,288 mod 988027.

Therefore,

**661578584939 mod 988027** = ((661578524,288) mod 988027 \* (66157832768) mod 988027\* (6615788192) mod 988027 \* (6615784096) mod 988027 \* (6615781024) mod 988027 \* (661578512) mod 988027) \* (66157864) mod 988027 \* (66157832) mod 988027 \* (6615788) mod 988027) \* (6615781) mod 988027)) mod 988027

= **506574**

Which is our original message. Hence, we have successfully decrypted the encrypted message.

**I.7 (Bonus)[5 points]:**

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

**Solution:**

We can map each integer of message to alphabet to get the final message. We can map 0 to A, 1 to B, 2 to C… etc. Then 506574 would translate to:

5 = F

0 = A

6 = G

5 = F

7 = H

4 = E

Therefore, the message in alphabet would be **FAGFHE.**

Problem II[40 points]:

Assume that we define

h1(k) = └ m(k A mod 1) ┘, where m = 13 and A = 0.62,

and

h2(k) = 1 + └ m(k A mod 1) ┘, where m = 11 and A = 0.62,

II.1. if linear probing is employed.

Given K = {369, 119, 287, 712, 141, 503, 186, 295, 528, 625} and size of table is 13.

Therefore, the function becomes h(k) = k mod 13

**Solution:**

We have the following definition for linear probing:

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

Where, h1(k) = └ m(k A mod 1) ┘, where m = 13 and A = 0.62

1. For K = 369,

We have h1(k) = └ m(k A mod 1) ┘, where m = 13 and A = 0.62,

On substituting the value:

h1(k) = └ 13 (369 \* 0.62 mod 1) ┘

= └ 13 (228.78 mod 1) ┘

= └ 13 (0.78) ┘

= └ 10.14 ┘

= 10

Now, h(k, 0) = (h1(k) + i) mod m

= (10 + 0) mod 13

= 10

Therefore, k = 369, goes to the 10th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  | 369 |  |  |

1. For K = 119,

On substituting the value:

h1(k) = └ 13 (119 \* 0.62 mod 1) ┘

= └ 13 (73.78 mod 1) ┘

= └ 13 (0.78) ┘

= └ 10.14 ┘

= 10

Now, h(k, 0) = (h1(k) + i) mod m

= (10 + 0) mod 13

= 10

**We have our first collision since the 10th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (10 + 1) mod 13

= 11

Therefore, k = 119, goes to the 11th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  | 369 | 119 |  |

1. For K = 287,

On substituting the value:

h1(k) = └ 13 (287 \* 0.62 mod 1) ┘

= └ 13 (177.94 mod 1) ┘

= └ 13 (0.94) ┘

= └ 12.21999999 ┘

= 12

Now, h(k, 0) = (h1(k) + i) mod m

= (12 + 0) mod 13

= 12

Therefore, k = 287, goes to the 12th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  | 369 | 119 | 287 |

1. For K = 712,

On substituting the value:

h1(712) = └ 13 (712 \* 0.62 mod 1) ┘

h1(712) = └ 13 (441.44 mod 1) ┘

h1(712) = └ 13 (0.44) ┘

h1(712) = └ 5.72 ┘

h1(712) = 5

Now, h(k, 0) = (h1(k) + i) mod m

= (5 + 0) mod 13

= 5

Therefore, k = 712, goes to the 5th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  | 712 |  |  |  |  | 369 | 119 | 287 |

1. For K = 141,

On substituting the value:

h1(141) = └ 13 (141 \* 0.62 mod 1) ┘

h1(141) = └ 13 (87.42 mod 1) ┘

h1(141) = └ 13 (0.42) ┘

h1(141) = └ 5.46 ┘

h1(141) = 5

Now, h(k, 0) = (h1(k) + i) mod m

= (5 + 0) mod 13

= 5

**We have our second collision since the 5th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (5 + 1) mod 13

= 6

Therefore, k = 141, goes to the 6th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  | 712 | 141 |  |  |  | 369 | 119 | 287 |

1. For K = 503,

On substituting the value:

h1(503) = └ 13 (503 \* 0.62 mod 1) ┘

h1(503) = └ 13 (311.86 mod 1) ┘

h1(503) = └ 13 (0.86) ┘

h1(503) = └ 11.18 ┘

h1(503) = 11

Now, h(k, 0) = (h1(k) + i) mod m

= (11 + 0) mod 13

= 11

**We have our third collision since the 11th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 1) mod 13

= 12

But since 12 is also occupied, we update i = 2. Also, number of collisions = 4

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 2) mod 13

= 0

Therefore, k = 503, goes to the 0th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 |  |  |  |  | 712 | 141 |  |  |  | 369 | 119 | 287 |

1. For K = 186,

On substituting the value:

h1(186) = └ 13 (186 \* 0.62 mod 1) ┘

h1(186) = └ 13 (115.32 mod 1) ┘

h1(186) = └ 13 (0.32) ┘

h1(186) = └ 4.16 ┘

h1(186) = 4

Now, h(k, 0) = (h1(k) + i) mod m

= (4 + 0) mod 13

= 4

Therefore, k = 186, goes to the 4th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 |  |  |  | 186 | 712 | 141 |  |  |  | 369 | 119 | 287 |

1. For K = 295,

On substituting the value:

h1(295) = └ 13 (295 \* 0.62 mod 1) ┘

h1(295) = └ 13 (182.9 mod 1) ┘

h1(295) = └ 13 (0.9) ┘

h1(295) = └ 11.70 ┘

h1(295) = 11

Now, h(k, 0) = (h1(k) + i) mod m

= (11 + 0) mod 13

= 11

**We have our fifth collision since the 11th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 1) mod 13

= 12

But since 12 is also occupied, we update i = 2. Also, number of collisions = 6

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 2) mod 13

= 0

But since 0 is also occupied, we update i = 3. Also, number of collisions = 7

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 3) mod 13

= 1

Therefore, k = 295, goes to the 1st slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 | 295 |  |  | 186 | 712 | 141 |  |  |  | 369 | 119 | 287 |

1. For K = 528,

On substituting the value:

h1(528) = └ 13 (528 \* 0.62 mod 1) ┘

h1(528) = └ 13 (327.36 mod 1) ┘

h1(528) = └ 13 (0.36) ┘

h1(528) = └ 4.68 ┘

h1(528) = 4

Now, h(k, 0) = (h1(k) + i) mod m

= (4 + 0) mod 13

= 4

**We have our eighth collision since the 4th location is already occupied**. Therefore, we update the value of i to 1.

We keep on updating the value from I = 1 to I = 3. Then we finally settle the value in 7th position. Also, number of collisions so far = 10

Therefore, k = 528, goes to the 7th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 | 295 |  |  | 186 | 712 | 141 | 528 |  |  | 369 | 119 | 287 |

1. For K = 625,

On substituting the value:

h1(625) = └ 13 (625 \* 0.62 mod 1) ┘

h1(625) = └ 13 (387.5 mod 1) ┘

h1(625) = └ 13 (0.5) ┘

h1(625) = └ 6.5 ┘

h1(625) = 6

Now, h(k, 0) = (h1(k) + i) mod m

= (6 + 0) mod 13

= 6

**We have our eleventh collision since the 6th location is already occupied**. Therefore, we update the value of i to 1.

We keep on updating the value from I =0 to I = 2. Then we finally settle the value in 8th position. Also, number of collisions so far = 12

Therefore, k = 625, goes to the 8th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 | 295 |  |  | 186 | 712 | 141 | 528 | 625 |  | 369 | 119 | 287 |

**Final answer for linear probing. This is how the keys are distributed in the table if linear probing is used:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 503 | 295 |  |  | 186 | 712 | 141 | 528 | 625 |  | 369 | 119 | 287 |

**Total number of collisions: 12**

II.2. if quadratic probing is employed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Solution:

For **Quadratic Probing** we have the following function definition:

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

where, h1(k) = └ m(k A mod 1) ┘, where m = 13 and A = 0.62,

and c1 3 c2 = 5 and i = {0,1, … m-1}

1. For K = 369,

On substituting the value, On substituting the value:

h1(k) = └ 13 (369 \* 0.62 mod 1) ┘

= └ 13 (228.78 mod 1) ┘

= └ 13 (0.78) ┘

= └ 10.14 ┘

= 10

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

= (10 + 3\*0 + 5 \* 02 ) mod 13

= 10

Therefore, k = 369, goes to the 10th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  | 369 |  |  |

1. For K = 119,

On substituting the value:

h1(k) = └ 13 (119 \* 0.62 mod 1) ┘

= └ 13 (73.78 mod 1) ┘

= └ 13 (0.78) ┘

= └ 10.14 ┘

= 10

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

= (10 + 3\*0 + 5 \* 02 ) mod 13

= 10

**We have our first collision since the 10th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (10 + 3\*1 + 5 \* 12 ) mod 13

= (10 + 3 + 5) mod 13

= 18 mod 13

= 5

Therefore, k = 119, goes to the 5th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  | 119 |  |  |  |  | 369 |  |  |

1. For K = 287,

On substituting the value:

h1(k) = └ 13 (287 \* 0.62 mod 1) ┘

= └ 13 (177.94 mod 1) ┘

= └ 13 (0.94) ┘

= └ 12.21999999 ┘

= 12

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

= (12 + 3\*0 + 5 \* 02 ) mod 13

= 12

Therefore, k = 287, goes to the 12th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  | 119 |  |  |  |  | 369 |  | 287 |

1. For K = 712,

On substituting the value:

h1(712) = └ 13 (712 \* 0.62 mod 1) ┘

h1(712) = └ 13 (441.44 mod 1) ┘

h1(712) = └ 13 (0.44) ┘

h1(712) = └ 5.72 ┘

h1(712) = 5

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

= (5 + 3\*0 + 5 \* 02 ) mod 13

= 5

**We have our second collision since the 5th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (5 + 3\*1 + 5 \* 12 ) mod 13

= (5 + 3 + 5) mod 13

= 13 mod 13

= 0

Therefore, k = 712, goes to the 0th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 |  |  |  |  | 119 |  |  |  |  | 369 |  | 287 |

1. For K = 141,

On substituting the value:

h1(141) = └ 13 (141 \* 0.62 mod 1) ┘

h1(141) = └ 13 (87.42 mod 1) ┘

h1(141) = └ 13 (0.42) ┘

h1(141) = └ 5.46 ┘

h1(141) = 5

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

= (5 + 3\*0 + 5 \* 02 ) mod 13

= 5

**We have our third collision since the 5th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (5 + 3\*1 + 5 \* 12 ) mod 13

= (5 + 3 + 5) mod 13

= 13 mod 13

= 0

**We have our fourth collision since the 0th location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 2) = (h1(k) + i) mod m

= (5 + 3\*2 + 5 \* 22 ) mod 13

= (5 + 6 + 20) mod 13

= 31 mod 13

= 5

**We have our fifth collision since the 5th location is already occupied**. Therefore, we update the value of i to 3.

Now, h(k, 3) = (h1(k) + i) mod m

= (5 + 3\*3 + 5 \* 32 ) mod 13

= (5 + 9 + 45) mod 13

= 31 mod 13

= 7

Therefore, k = 141, goes to the 7th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 |  |  |  |  | 119 |  | 141 |  |  | 369 |  | 287 |

1. For K = 503,

On substituting the value:

h1(503) = └ 13 (503 \* 0.62 mod 1) ┘

h1(503) = └ 13 (311.86 mod 1) ┘

h1(503) = └ 13 (0.86) ┘

h1(503) = └ 11.18 ┘

h1(503) = 11

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

= (11 + 3\*0 + 5 \* 02 ) mod 13

= 11

Therefore, k = 503, goes to the 11th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 |  |  |  |  | 119 |  | 141 |  |  | 369 | 503 | 287 |

1. For K = 186,

On substituting the value:

h1(186) = └ 13 (186 \* 0.62 mod 1) ┘

h1(186) = └ 13 (115.32 mod 1) ┘

h1(186) = └ 13 (0.32) ┘

h1(186) = └ 4.16 ┘

h1(186) = 4

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

= (4 + 3\*0 + 5 \* 02 ) mod 13

= 4

Therefore, k = 186, goes to the 4th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 |  |  |  | 186 | 119 |  | 141 |  |  | 369 | 503 | 287 |

1. For K = 295,

On substituting the value:

h1(295) = └ 13 (295 \* 0.62 mod 1) ┘

h1(295) = └ 13 (182.9 mod 1) ┘

h1(295) = └ 13 (0.9) ┘

h1(295) = └ 11.70 ┘

h1(295) = 11

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

= (11 + 3\*0 + 5 \* 02 ) mod 13

= 11

**We have our sixth collision since the 11th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (11 + 3\*1 + 5 \* 12 ) mod 13

= (11+ 3 + 5) mod 13

= 19 mod 13

= 6

Therefore, k = 295, goes to the 6th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 |  |  |  | 186 | 119 | 295 | 141 |  |  | 369 | 503 | 287 |

1. For K = 528,

On substituting the value:

h1(528) = └ 13 (528 \* 0.62 mod 1) ┘

h1(528) = └ 13 (327.36 mod 1) ┘

h1(528) = └ 13 (0.36) ┘

h1(528) = └ 4.68 ┘

h1(528) = 4

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

= (4 + 3\*0 + 5 \* 02 ) mod 13

= 4

**We have our seventh collision since the 4th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (4 + 3\*1 + 5 \* 12 ) mod 13

= (4+ 3 + 5) mod 13

= 12 mod 13

= 12

**We have our eighth collision since the 12th location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 2) = (h1(k) + i) mod m

= (4 + 3\*2 + 5 \* 22 ) mod 13

= (4+ 6 + 20) mod 13

= 30 mod 13

= 4

**We have our ninth collision since the 4th location is already occupied**. Therefore, we update the value of i to 3.

Now, h(k, 3) = (h1(k) + i) mod m

= (4 + 3\*3 + 5 \* 32 ) mod 13

= (4+ 9 + 45) mod 13

= 30 mod 13

= 6

**We have our tenth collision since the 6th location is already occupied**. Therefore, we update the value of i to 4.

Now, h(k, 4) = (h1(k) + i) mod m

= (4 + 3\*4 + 5 \* 42 ) mod 13

= (4+ 12 + 80) mod 13

= 96 mod 13

= 5

**We have our eleventh collision since the 5th location is already occupied**. Therefore, we update the value of i to 5.

Now, h(k, 5) = (h1(k) + i) mod m

= (4 + 3\*5 + 5 \* 52 ) mod 13

= (4+ 15 + 125) mod 13

= 144 mod 13

= 1

Therefore, k = 528, goes to the 1st slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 | 528 |  |  | 186 | 119 | 295 | 141 |  |  | 369 | 503 | 287 |

1. For K = 625,

On substituting the value:

h1(625) = └ 13 (625 \* 0.62 mod 1) ┘

h1(625) = └ 13 (387.5 mod 1) ┘

h1(625) = └ 13 (0.5) ┘

h1(625) = └ 6.5 ┘

h1(625) = 6

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

= (6 + 3\*0 + 5 \* 02 ) mod 13

= 6

**We have our twelfth collision since the 6th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i) mod m

= (6 + 3\*1 + 5 \* 12 ) mod 13

= (6+ 3 + 5) mod 13

= 14 mod 13

= 1

**We have our thirteenth collision since the 1st location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 2) = (h1(k) + i) mod m

= (6 + 3\*2 + 5 \* 22 ) mod 13

= (6+ 6 + 20) mod 13

= 32 mod 13

= 6

**We have our fourteenth collision since the 6th location is already occupied**. Therefore, we update the value of i to 3.

Now, h(k, 3) = (h1(k) + i) mod m

= (6 + 3\*3 + 5 \* 32 ) mod 13

= (6+ 9 + 45) mod 13

= 30 mod 13

= 8

Therefore, k = 625, goes to the 8th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 | 528 |  |  | 186 | 119 | 295 | 141 | 625 |  | 369 | 503 | 287 |

**Final answer for Quadratic Probing. This is how the keys are distributed in the table if Quadratic probing is used:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 712 | 528 |  |  | 186 | 119 | 295 | 141 | 625 |  | 369 | 503 | 287 |

**Total number of collisions: 14**

**The number of collisions is surprising as I was thinking it be lesser than linear probing considering the fact that it has more complex hashing function.**

II.3. if double hashing is employed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

**Solution:**

Double hashing is defined as

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

where, h1(k) = └ m(k A mod 1) ┘, where m = 13 and A = 0.62,

and, h2(k) = 1 + └ m(k A mod 1) ┘, where m = 11 and A = 0.62,

1. For K = 369,

On substituting the value:

h1(369) = └ 13 (369 \* 0.62 mod 1) ┘

h1(369) = └ 13 (228.78 mod 1) ┘

h1(369) = └ 13 (0.78) ┘

h1(369) = └ 10.14 ┘

h1(369) = 10

Calculating h2

h2(369) = 1 + └ 11 (369 \* 0.62 mod 1) ┘

h2(369) = 1 + └ 11 (228.78 mod 1) ┘

h2(369) = 1 + └ 11 (0.78) ┘

h2(369) = 1 + └ 8.58 ┘

h2(369) = 1 + 8

h2(369) = 9

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

= (10 + 0 \* 9) mod 13

= 10

Therefore, k = 369 occupies the 10th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  |  |  |  |  | 369 |  |  |

1. For K = 119

On substituting the values:

h1(119) = └ 13 (119 \* 0.62 mod 1) ┘

h1(119) = └ 13 (73.78 mod 1) ┘

h1(119) = └ 13 (0.78) ┘

h1(119) = └ 10.14 ┘

h1(119) = 10

Calculating h2

h2(119) = 1 + └ 11 (119 \* 0.62 mod 1) ┘

h2(119) = 1 + └ 11 (73.78 mod 1) ┘

h2(119) = 1 + └ 11 (0.78) ┘

h2(119) = 1 + └ 8.58 ┘

h2(119) = 1 + 8

h2(119) = 9

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

= (10 + 0 \* 9) mod 13

= 10

**We have our first collision since the 10th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (10 + 1 \* 9) mod 13

= 6

Therefore, k = 119 occupies the 6th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  | 119 |  |  |  | 369 |  |  |

1. For k = 287,

On substituting the values:

h1(287) = └ 13 (287 \* 0.62 mod 1) ┘

h1(287) = └ 13 (177.94 mod 1) ┘

h1(287) = └ 13 (0.94) ┘

h1(287) = └ 12.219999999999999 ┘

h1(287) = 12

Calculating h2

h2(287) = 1 + └ 11 (287 \* 0.62 mod 1) ┘

h2(287) = 1 + └ 11 (177.94 mod 1) ┘

h2(287) = 1 + └ 11 (0.94) ┘

h2(287) = 1 + └ 10.34 ┘

h2(287) = 1 + 10

h2(287) = 11

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

= (12 + 0 \*11) mod 13

= 12

Therefore, k = 287 occupies the 12th spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  |  | 119 |  |  |  | 369 |  | 287 |

1. For k = 712,

On substituting the values:

h1(712) = └ 13 (712 \* 0.62 mod 1) ┘

h1(712) = └ 13 (441.44 mod 1) ┘

h1(712) = └ 13 (0.44) ┘

h1(712) = └ 5.72 ┘

h1(712) = 5

Calculating h2

h2(712) = 1 + └ 11 (712 \* 0.62 mod 1) ┘

h2(712) = 1 + └ 11 (441.44 mod 1) ┘

h2(712) = 1 + └ 11 (0.44) ┘

h2(712) = 1 + └ 4.84 ┘

h2(712) = 1 + 4

h2(712) = 5

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

= (5 + 0 \* 5) mod 13

= 5

Therefore, k = 712 occupies the 5th slot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  |  |  | 712 | 119 |  |  |  | 369 |  | 287 |

1. For k = 141

On substituting the values:

h1(141) = └ 13 (141 \* 0.62 mod 1) ┘

h1(141) = └ 13 (87.42 mod 1) ┘

h1(141) = └ 13 (0.42) ┘

h1(141) = └ 5.46 ┘

h1(141) = 5

Calculating h2

h2(141) = 1 + └ 11 (141 \* 0.62 mod 1) ┘

h2(141) = 1 + └ 11 (87.42 mod 1) ┘

h2(141) = 1 + └ 11 (0.42) ┘

h2(141) = 1 + └ 4.62 ┘

h2(141) = 1 + 4

h2(141) = 5

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

= (5 + 0 \* 5) mod 13

= 5

**We have our second collision since the 5th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (5 + 1 \* 5) mod 13

= 10

**We have our third collision since the 10th location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (5 + 2 \* 5) mod 13

= 2

therefore, k = 141 occupies the 2nd spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 |  |  | 712 | 119 |  |  |  | 369 |  | 287 |

1. For k = 503,

On substituting the values:

h1(503) = └ 13 (503 \* 0.62 mod 1) ┘

h1(503) = └ 13 (311.86 mod 1) ┘

h1(503) = └ 13 (0.86) ┘

h1(503) = └ 11.18 ┘

h1(503) = 11

Calculating h2

h2(503) = 1 + └ 11 (503 \* 0.62 mod 1) ┘

h2(503) = 1 + └ 11 (311.86 mod 1) ┘

h2(503) = 1 + └ 11 (0.86) ┘

h2(503) = 1 + └ 9.459999999999999 ┘

h2(503) = 1 + 9

h2(503) = 10

Now, h (k, 1) = (h1(k) + i h2(k)) mod m,

= (11 + 0 \* 10) mod 13

= 11

Therefore, k = 503 occupies the 11th spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 |  |  | 712 | 119 |  |  |  | 369 | 503 | 287 |

1. For k = 186,

On substituting the values:

h1(186) = └ 13 (186 \* 0.62 mod 1) ┘

h1(186) = └ 13 (115.32 mod 1) ┘

h1(186) = └ 13 (0.32) ┘

h1(186) = └ 4.16 ┘

h1(186) = 4

Calculating h2

h2(186) = 1 + └ 11 (186 \* 0.62 mod 1) ┘

h2(186) = 1 + └ 11 (115.32 mod 1) ┘

h2(186) = 1 + └ 11 (0.32) ┘

h2(186) = 1 + └ 3.52 ┘

h2(186) = 1 + 3

h2(186) = 4

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (4 + 0 \* 4) mod 13

= 4

Therefore, k = 186 occupies the 4th spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 |  | 186 | 712 | 119 |  |  |  | 369 | 503 | 287 |

1. For k = 295,

On substituting the values:

h1(295) = └ 13 (295 \* 0.62 mod 1) ┘

h1(295) = └ 13 (182.9 mod 1) ┘

h1(295) = └ 13 (0.9) ┘

h1(295) = └ 11.700000000000001 ┘

h1(295) = 11

Calculating h2

h2(295) = 1 + └ 11 (295 \* 0.62 mod 1) ┘

h2(295) = 1 + └ 11 (182.9 mod 1) ┘

h2(295) = 1 + └ 11 (0.9) ┘

h2(295) = 1 + └ 9.9 ┘

h2(295) = 1 + 9

h2(295) = 10

Now, h(k, 0) = (h1(k) + i h2(k) ) mod m,

= (11 + 0 \* 10) mod 13

= 11

**We have our fourth collision since the 11th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (11 + 1 \* 10) mod 13

= 21 mod 13

= 8

Therefore, k = 295 occupies the 8th spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 |  | 186 | 712 | 119 |  | 295 |  | 369 | 503 | 287 |

1. For k = 528,

On substituting the values:

h1(528) = └ 13 (528 \* 0.62 mod 1) ┘

h1(528) = └ 13 (327.36 mod 1) ┘

h1(528) = └ 13 (0.36) ┘

h1(528) = └ 4.68 ┘

h1(528) = 4

Calculating h2

h2(528) = 1 + └ 11 (528 \* 0.62 mod 1) ┘

h2(528) = 1 + └ 11 (327.36 mod 1) ┘

h2(528) = 1 + └ 11 (0.36) ┘

h2(528) = 1 + └ 3.96 ┘

h2(528) = 1 + 3

h2(528) = 4

Now, h(k, 0) = (h1(k) + i h2(k) ) mod m,

= (4 + 0 \* 4) mod 13

= 4

**We have our fifth collision since the 4th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (4 + 1\* 4) mod 13

= 8 mod 13

= 8

**We have our sixth collision since the 8th location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 2) = (h1(k) + i h2(k) ) mod m,

= (4 + 2\* 4) mod 13

= 12 mod 13

= 12

**We have our sixth collision since the 12th location is already occupied**. Therefore, we update the value of i to 3.

Now, h(k, 3) = (h1(k) + i h2(k) ) mod m,

= (4 + 3\* 4) mod 13

= 16 mod 13

= 3

Therefore, k = 528 occupies the 3rd spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 | 528 | 186 | 712 | 119 |  | 295 |  | 369 | 503 | 287 |

1. For k = 625,

On substituting the values:

h1(625) = └ 13 (625 \* 0.62 mod 1) ┘

h1(625) = └ 13 (387.5 mod 1) ┘

h1(625) = └ 13 (0.5) ┘

h1(625) = └ 6.5 ┘

h1(625) = 6

Calculating h2

h2(625) = 1 + └ 11 (625 \* 0.62 mod 1) ┘

h2(625) = 1 + └ 11 (387.5 mod 1) ┘

h2(625) = 1 + └ 11 (0.5) ┘

h2(625) = 1 + └ 5.5 ┘

h2(625) = 1 + 5

h2(625) = 6

Now, h(k, 0) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 0) mod 13

= 6 mod 13

= 6

**We have our seventh collision since the 6th location is already occupied**. Therefore, we update the value of i to 1.

Now, h(k, 1) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 1) mod 13

= 12 mod 13

= 12

**We have our eighth collision since the 12th location is already occupied**. Therefore, we update the value of i to 2.

Now, h(k, 2) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 2) mod 13

= 18 mod 13

= 5

**We have our ninth collision since the 5th location is already occupied**. Therefore, we update the value of i to 3.

Now, h(k, 3) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 3) mod 13

= 24 mod 13

= 11

**We have our ninth collision since the 11th location is already occupied**. Therefore, we update the value of i to 4.

Now, h(k, 4) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 4) mod 13

= 30 mod 13

= 4

**We have our tenth collision since the 4th location is already occupied**. Therefore, we update the value of i to 5.

Now, h(k, 5) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 5) mod 13

= 36 mod 13

= 10

**We have our eleventh collision since the 10th location is already occupied**. Therefore, we update the value of i to 6.

Now, h(k, 6) = (h1(k) + i h2(k) ) mod m,

= (6 + 6 \* 6) mod 13

= 42 mod 13

= 3

**We have our twefth collision since the 3rd location is already occupied**. Therefore, we update the value of i to 7.

Now, h(k, 7) = (h1(k) + i h2(k) ) mod m,

= (6 + 7 \* 6) mod 13

= 48 mod 13

= 9

Therefore, k = 625 occupies the 3rd spot.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 | 528 | 186 | 712 | 119 |  | 295 | 625 | 369 | 503 | 287 |

**Final answer for Double hashing. This is how the keys are distributed in the table if Double probing is used**:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  | 141 | 528 | 186 | 712 | 119 |  | 295 | 625 | 369 | 503 | 287 |

**Total number of collisions: 12**

II.4. Compare the linear probing, quadratic probing, and double hashing in terms of the number of occurred collisions for the given K.

**Solution:**

Linear probing is the easiest to implement and the hashing function is comparatively simple. Because of the above fact I was expecting it to perform the worse since there were more sophisticated techniques like quadratic and double probing. Linear probing had a total of 12 collisions. The number of collisions became like a benchmark for comparing the other two probing techniques.

Quadratic probing had a total of 14 collisions. This wasn’t expected as its hashing function is much more complicated than linear probing. It suffered more collisions indicating that it’s less efficient. At least in our case. Also, more collisions mean there is more **clustering** of keys.

Double probing had 12 collisions again. In the beginning Double probing was performing much better than the other two. It barely **had 6 collisions** after putting all the 9 keys. But in the last key it suffered **another 6 collisions** which **degraded** its performance. It performed better than quadratic probing and similar to linear. But still, I think **Double hashing is better performing** than the rest. Because it only had 6 collisions after putting 9 keys which is remarkable.