Group C Crypto Algorithm Image

Made By: Kegan Lavoy, Daris Lychuk, Jamie Plunkett

Requirement 1

Req. 1: Generating long list of ints

- Using a simple for loop, generated 200 long int numbers using a long int seed
- Print results to a text file for comparison later

```
public static void main(String args[]){
    Random rand = new Random(40236);

    try
    {
        PrintWriter pr = new PrintWriter("LongList1.txt");

        for (int i=0; i< 200 ; i++)
        {
            pr.println(Math.abs(rand.nextLong()));
        }
        pr.close();
    }
```

LongList - Notepad

File Edit Format View Help

5039914289563315292
2571611009471911582
5856346033122223708
3423234798896025865
5604020300222300186
289428283091810429
8488044247261715449

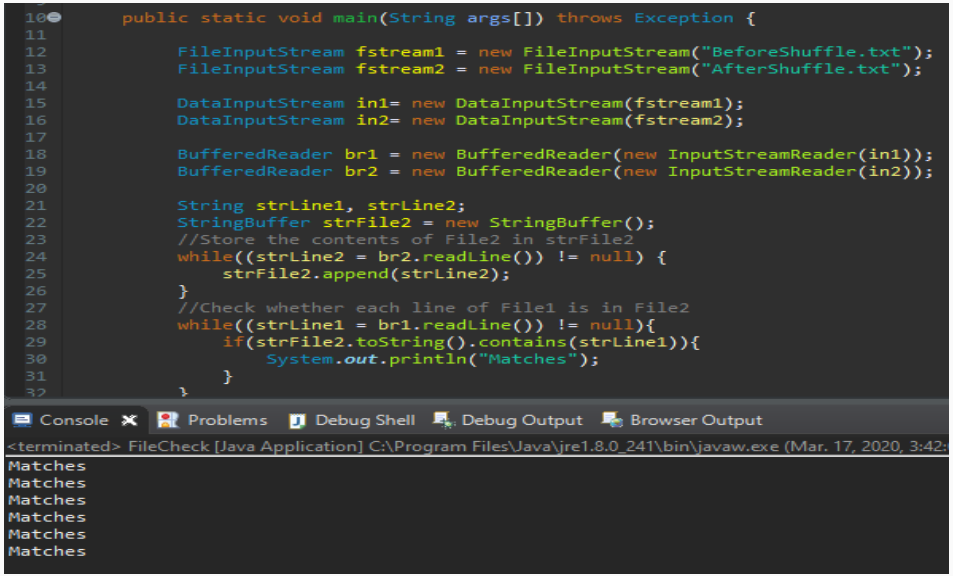
LongList1 - Notepad

File Edit Format View

5039914289563315292
2571611009471911582
5856346033122223708
3423234798896025865
5604020300222300186
289428283091810429
8488044247261715449

Req. 1: Comparison

- Takes 2 text files as input
- Compares files line by line
- If the 2 lines being compared are the same, print “Matches”, else print different



Requirement 2

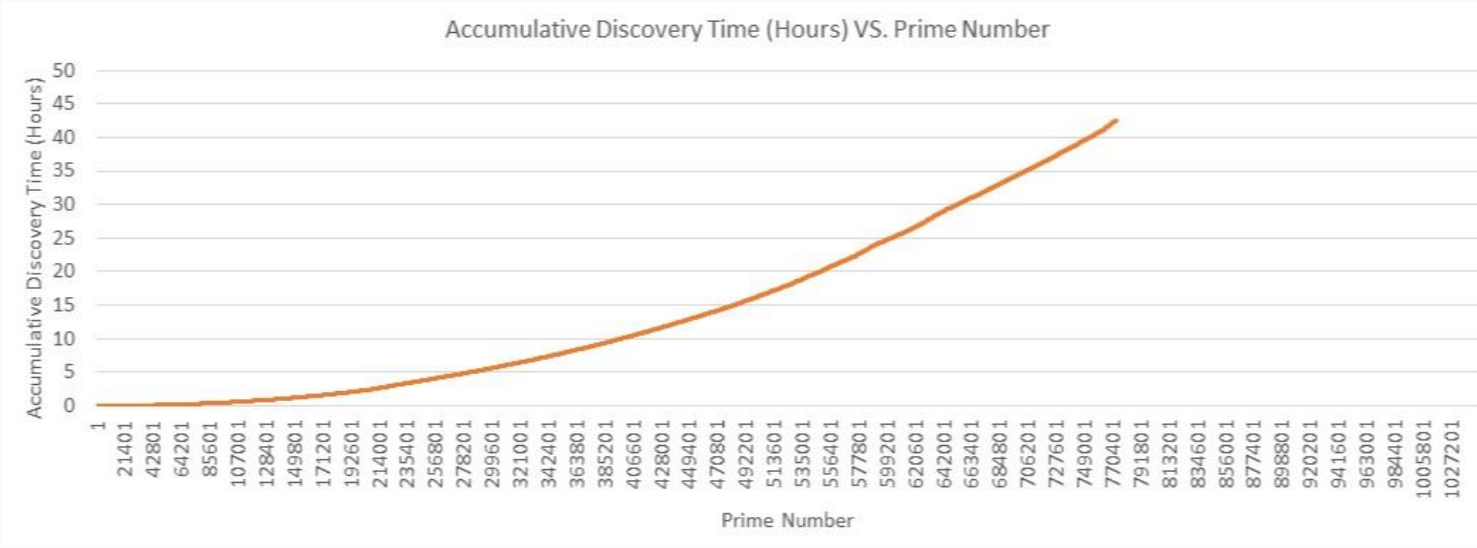
Req. 2: Prime Number Generation

- Tests numbers for primality using modulo division
- Increases a counter when a modulo division = 0
- If counter becomes greater than 2, number is not prime

```
int i =0;
int num =0;
String  primeNumbers = "";
long n = 999999999;
long startTime = System.nanoTime();
for (i = 1; i <= n; i++){
    int counter=0;
for(num =i; num>=1; num--){
    if(i%num==0){
        counter = counter + 1;
    }
}
if (counter ==2){
   long endTime = System.nanoTime();
   long timeElapsed = endTime - startTime;
   primeNumbers = primeNumbers + i + "," +
           timeElapsed / 1000000 + "\n";
   try{
       PrintWriter pr = new PrintWriter("PrimeBank.csv");
           pr.println(primeNumbers);
```

Req. 2: Prime Number Graphs

	A	B
773601	11764747	1.53E+08
773602	11764801	1.53E+08
773603	11764813	1.53E+08
773604	11764849	1.53E+08
773605	11764861	1.53E+08
773606	11764873	1.53E+08
773607	11764877	1.53E+08
773608	11764889	1.53E+08
773609	11764967	1.53E+08
773610	11765023	1.53E+08
773611	11765051	1.53E+08
773612	11765053	1.53E+08
773613	11765057	1.53E+08
773614	11765059	1.53E+08
773615	11765063	1.53E+08
773616	11765069	1.53E+08
773617	11765081	1.53E+08
773618	11765093	1.53E+08
773619	11765111	1.53E+08
773620	11765141	1.53E+08
773621	11765179	1.53E+08
773622	11765209	1.53E+08
773623	11765231	1.53E+08
773624	11765239	1.53E+08



Requirement 3

Req. 3: Bob/Alice Key Generation

- Get two passwords from user, turn into ints for DH algorithm
- Apply DH algorithm and use shared key in next steps

```
16          P = 275911; //This is a prime taken from excel
17          G = 2;
18
19          System.out.println("Enter password for Alice : ");
20          String APass = AlicePass.nextLine();
21
22          for(int i = 0; i < APass.length();  i++)
23          {
24              a = a + Character.getNumericValue(APass.charAt(i));
25          }
26          a = (long) Math.sqrt(a); // a is the chosen private key
27          x = (long) (Math.pow(G, a)); // gets the generated key
28          x = x%P;
29
30          System.out.println("Enter password for Bob : ");
31          String BPass = BobPass.nextLine();
32
33          for(int i = 0; i < BPass.length();  i++)
34          {
35              b = b + Character.getNumericValue(BPass.charAt(i));
36          }
37          b = (long) Math.sqrt(b); // b is the chosen private key
38          y = (long) (Math.pow(G, b)); // gets the generated key
39          y = y%P;
40
41          keyA = (long) (Math.pow(y, a)%P); //Secret key for Alice
42          keyB = (long) (Math.pow(x, b)%P); //Secret key for Bob
43
44          System.out.println("Secret key for the Alice is : " + keyA);
```

Console | Problems | Debug Shell | Debug Output | Browser Output

```
<terminated> DHKeyAgreement [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw
Enter password for Alice :
mypassword
Enter password for Bob :
hellothere
Secret key for the Alice is : 186874
Secret Key for the Bob is : 186874
```

Requirement 4

Req. 4: Shared Key test on random ints

- Used shared key generated by passwords as a check for other functions
- This was tested before in the previous requirements

Requirement 5

Req. 5: Shuffle Algorithms

```
public static int[] Shuffle(int[] toShuffle, int key)
    {
        int size = toShuffle.length;
        int[] exchanges = GetShuffleExchanges(size, key);
        for (int i = size - 1; i > 0; i--)
        {
            int n = exchanges[size - 1 - i];
            int tmp = toShuffle[i];
            toShuffle[i] = toShuffle[n];
            toShuffle[n] = tmp;
        }
        return toShuffle;
    }
public static int[] DeShuffle(int[] shuffled, int key)
    {
        int size = shuffled.length;
        int[] exchanges = GetShuffleExchanges(size, key);
        for (int i = 1; i < size; i++)
        {
            int n = exchanges[size - i - 1];
            int tmp = shuffled[i];
            shuffled[i] = shuffled[n];
            shuffled[n] = tmp;
        }
        return shuffled;
    }
```

```
public static int[] GetShuffleExchanges(int size, int key)
{
    int[] exchanges = new int[size - 1];
    Random rand = new Random(key);
    for (int i = size - 1; i > 0; i--)
    {
        int n = rand.nextInt(i + 1);
        exchanges[size - 1 - i] = n;
    }
    return exchanges;
}
```

- Uses shared key as a seed for random number generator
- Applies shuffling algorithm to reorder pixels in the array

Req. 5: Image Shuffle

- Load in an image and extract all pixel values into a 1D array
- Shuffle that array using shared key as parameter and set new location for pixels

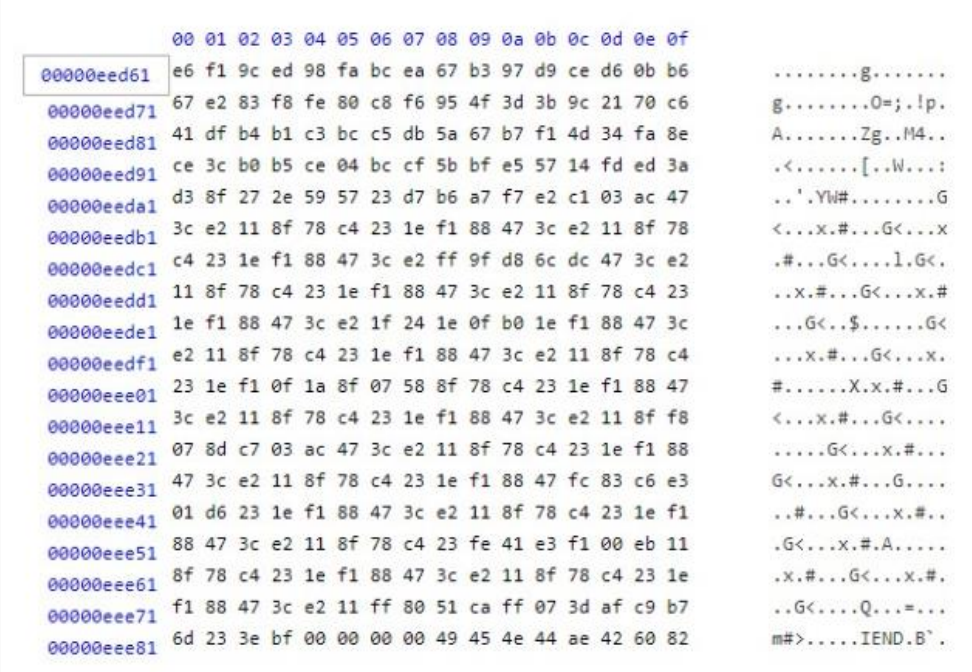
```
PrintWriter pr = new PrintWriter("BeforeShuffle.txt");
PrintWriter pr1 = new PrintWriter("AfterShuffle.txt");
BufferedImage originalImage = ImageIO.read(new File(
        "c:\\Users\\Owner\\Documents\\ENSE\\ENSE496AE\\Final\\TERRY.jpg"));

 width = originalImage.getWidth();
 height = originalImage.getHeight();
 int[] pixels = new int[(width*height)];


 for(int i=0; i<height; i++) {

     for(int j=0; j<width; j++) {

         int p = originalImage.getRGB(j,i);
         //p = p/167238;
         pixels[count] = p;
         count++;
        // Color c = new Color(originalImage.getRGB(j, i));
        // System.out.println("S.No: " + count + " Red: " + c.getRed() +"  Gr
         System.out.print(pixels[count-1]);
```

Req. 5: Shuffled Image example

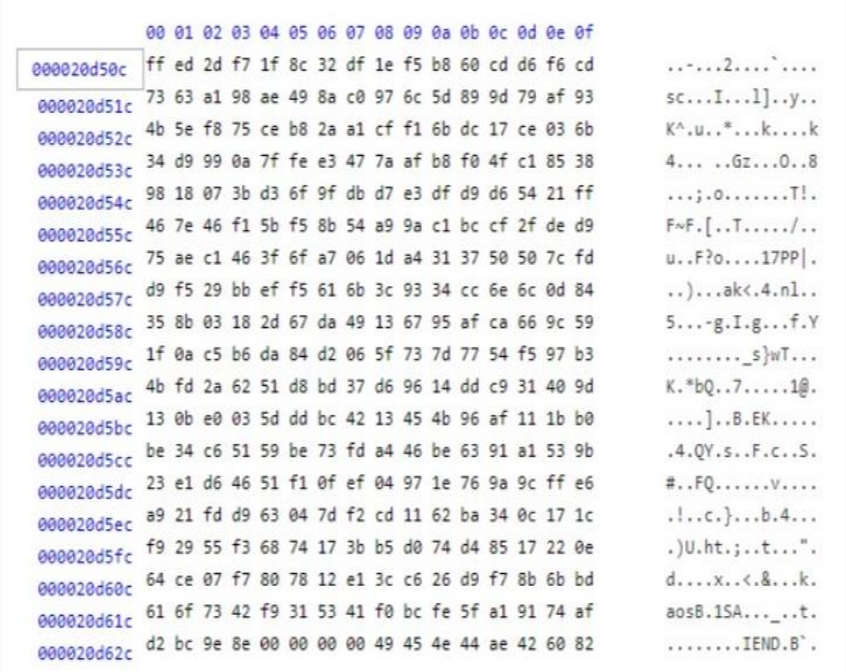
Req. 5: DeShuffle

- Call deshuffle using same shared key as a parameter
- Reinsert pixels into their proper places

Req. 5: Hex Comparison



	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f	
00000eed61	e6	f1	9c	ed	98	fa	bc	ea	67	b3	97	d9	ce	d6	0b	b6	........g.......
00000eed71	67	e2	83	f8	fe	80	c8	f6	95	4f	3d	3b	9c	21	70	c6	g........O=;.!p.
00000eed81	41	df	b4	b1	c3	bc	c5	db	5a	67	b7	f1	4d	34	fa	8e	A.......Zg..M4..
00000eed91	ce	3c	b0	b5	ce	04	bc	cf	5b	bf	e5	57	14	fd	ed	3a	.<......[..W...:
00000eeda1	d3	8f	27	2e	59	57	23	d7	b6	a7	f7	e2	c1	03	ac	47	..'.YW#........G
00000eedb1	3c	e2	11	8f	78	c4	23	1e	f1	88	47	3c	e2	11	8f	78	<...x.#...G<...x
00000eedc1	c4	23	1e	f1	88	47	3c	e2	ff	9f	d8	6c	dc	47	3c	e2	.#...G<....l.G<.
00000eedd1	11	8f	78	c4	23	1e	f1	88	47	3c	e2	11	8f	78	c4	23	..x.#...G<...x.#
00000eede1	1e	f1	88	47	3c	e2	1f	24	1e	0f	b0	1e	f1	88	47	3c	...G<..$......G<
00000eedf1	e2	11	8f	78	c4	23	1e	f1	88	47	3c	e2	11	8f	78	c4	...x.#...G<...x.
00000eee01	23	1e	f1	0f	1a	8f	07	58	8f	78	c4	23	1e	f1	88	47	#......X.x.#...G
00000eee11	3c	e2	11	8f	78	c4	23	1e	f1	88	47	3c	e2	11	8f	f8	<...x.#...G<....
00000eee21	07	8d	c7	03	ac	47	3c	e2	11	8f	78	c4	23	1e	f1	88	.....G<...x.#...
00000eee31	47	3c	e2	11	8f	78	c4	23	1e	f1	88	47	fc	83	c6	e3	G<...x.#...G....
00000eee41	01	d6	23	1e	f1	88	47	3c	e2	11	8f	78	c4	23	1e	f1	..#...G<...x.#..
00000eee51	88	47	3c	e2	11	8f	78	c4	23	fe	41	e3	f1	00	eb	11	.G<...x.#.A.....
00000eee61	8f	78	c4	23	1e	f1	88	47	3c	e2	11	8f	78	c4	23	1e	.x.#...G<...x.#.
00000eee71	f1	88	47	3c	e2	11	ff	80	51	ca	ff	07	3d	af	c9	b7	..G<....Q...=...
00000eee81	6d	23	3e	bf	00	00	00	00	49	45	4e	44	ae	42	60	82	m#>.....IEND.B`.



	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f	
000020d50c	ff	ed	2d	f7	1f	8c	32	df	1e	f5	b8	60	cd	d6	f6	cd	..-...2....`....
000020d51c	73	63	a1	98	ae	49	8a	c0	97	6c	5d	89	9d	79	af	93	sc...I...l]..y..
000020d52c	4b	5e	f8	75	ce	b8	2a	a1	cf	f1	6b	dc	17	ce	03	6b	K^.u..*...k....k
000020d53c	34	d9	99	0a	7f	fe	e3	47	7a	af	b8	f0	4f	c1	85	38	4... ..Gz...O..8
000020d54c	98	18	07	3b	d3	6f	9f	db	d7	e3	df	d9	d6	54	21	ff	...;.o.......T!.
000020d55c	46	7e	46	f1	5b	f5	8b	54	a9	9a	c1	bc	cf	2f	de	d9	F~F.[..T...../..
000020d56c	75	ae	c1	46	3f	6f	a7	06	1d	a4	31	37	50	50	7c	fd	u..F?o....17PP|.
000020d57c	d9	f5	29	bb	ef	f5	61	6b	3c	93	34	cc	6e	6c	0d	84	..)...ak<.4.nl..
000020d58c	35	8b	03	18	2d	67	da	49	13	67	95	af	ca	66	9c	59	5...-g.I.g...f.Y
000020d59c	1f	0a	c5	b6	da	84	d2	06	5f	73	7d	77	54	f5	97	b3	........_s}wT...
000020d5ac	4b	fd	2a	62	51	d8	bd	37	d6	96	14	dd	c9	31	40	9d	K.*bQ..7.....1@.
000020d5bc	13	0b	e0	03	5d	dd	bc	42	13	45	4b	96	af	11	1b	b0	....]..B.EK.....
000020d5cc	be	34	c6	51	59	be	73	fd	a4	46	be	63	91	a1	53	9b	.4.QY.s..F.c..S.
000020d5dc	23	e1	d6	46	51	f1	0f	ef	04	97	1e	76	9a	9c	ff	e6	#..FQ......v....
000020d5ec	a9	21	fd	d9	63	04	7d	f2	cd	11	62	ba	34	0c	17	1c	.!..c.}...b.4...
000020d5fc	f9	29	55	f3	68	74	17	3b	b5	d0	74	d4	85	17	22	0e	.)U.ht.;..t...".
000020d60c	64	ce	07	f7	80	78	12	e1	3c	c6	26	d9	f7	8b	6b	bd	d....x..<.&...k.
000020d61c	61	6f	73	42	f9	31	53	41	f0	bc	fe	5f	a1	91	74	af	aosB.1SA..._..t.
000020d62c	d2	bc	9e	8e	00	00	00	00	49	45	4e	44	ae	42	60	82	........IEND.B`.

Any Questions?