HW1-REPORT

```
public cMatrix(){
             row=10;
             col=10;
                    long[][] elements = new long[row][col];
                    this.elements=elements;
              this.row=10;
              this.col=10;
                   }
```

Bu fonksiyon 10x10 boyutunda matris oluşturur.

```
public cMatrix(int row, int col){ //Constructor
               long[][] elements = new long[row][col];
                    this.elements=elements;
                    this.row = row;
                    this.col = col;


         }
```

Bu fonksiyon row ve col değişkenlerine bağlı olarak matrisler oluşturur. Bir önceki fonksiyonun aksine istenilen değerlerde matris oluşturabilir.

```
public void AssignRandom(){ //Assigning random variables in range 1 – 10000

               int sayi;
               Random rnd = new Random();
                    sayi = rnd.nextInt(10000);

               for(int i=0; i<row; i++) {
                         for(int j=0; j<col; j++) {
                               elements[i][j]=rnd.nextInt(10000);
                         }
                }

         }
```

Bu fonksiyonun amacı oluşturulan boş matrislerin içerisine random sayılar yerleştirmektir. Java.util.Random kütüphanesi içerisindeki Random() komutu aracığıyla 10000'e kadar olan sayıları rastgele üretebiliyoruz. 2 boyutlu bir array'in tüm elemanlarına değer atamak için 2 adet for döngüsüne ihtiyacımız var. İlk olarak 0. Satırın tüm elemanlarından başlayarak sırasıyla tüm satırların içerisine random sayılar atar .

```
public void printMatrix(){

                elements=this.elements;


                        for(int i=0; i<row; i++) {
                              for(int j=0; j<col; j++) {
                              System.out.print(elements[i][j] + " ");}

                              System.out.print("\n");
```

Bu fonksiyon daha önce oluşturulan ve içerisine değerler atanan matrisleri yazdırır. Yine aynı şekilde 2 boyutlu bir array'in tüm elemanlarını yazdırmak için 2 adet for döngüsüne ihtiyacımız var. Matris elemanlarını raporda belirtildiği üzere düzenli bir şekilde gösterebilmek için her eleman yazdırıldıktan sonra hemen ardına bir adet boşluk ekledim. Yeni satırların bir alta geçmesi için de sadece 1. For döngüsünün içine bir adet \n ekledim. Bu alt satırıa geçme kodu sayesinde ilk for değişkeni olan row kadar satırlar oluşturulacaktır.

```
public void printMatrixWithPrime(){ //Prints the matrix with "*" signnear the
prime elements
                long beginTime = 0, endTime = 0;
                beginTime = System.currentTimeMillis();
                      int sayac = 0;

                      for(int i=0; i<row; i++) {
                                  for(int j=0; j<col; j++) {
                                 for(int a = 2; a < elements[i][j]; a++)
                      {
                          if(elements[i][j] % a == 0) {
                              sayac++;
                          }
                      }


                                   if(sayac == 0) {
                          System.out.print(elements[i][j] + "* ");
                      }
                      else {
                          System.out.print(elements[i][j] + " ");
                          sayac=0;
                      }
                      }
                                  System.out.print("\n");
                      }

                      endTime = System.currentTimeMillis();
                             System.out.println("The duration of multiplication of
matrices : " + this.row +"x" +this.col + "   " +((double) (endTime - beginTime)) +
" milisecond");
                             System.out.println("\n");
                 }
```

Bu fonksiyon matrisin içindeki asal sayıları yanında yıldızlı olacak şekilde yazdırır. Bu fonksiyon başlıca 3 adet for döngüsünden oluşmaktadır. İlk 2 for döngüsü matrisin elemanlarını çıkarmak için sonuncu for döngüsü ise matris

elemanlarının asal olup olmadığını test etmek için kullanılır. Matris elemanlarının asal tespitini yapmak için sonuncu for döngüsünün içerisinde matris elemanını, elemanın kendi değerine kadar olan 2 den başlayan tüm sayılara böldüm. Eğer bölümden kalan herhangi bir değerde 1 olursa bu sayı asal değildir, hiçbir şekilde bölümünden kalan 0'ı geçmiyorsa bu sayı asaldır. If else komutu aracılığıyla başta tanımladığım sayac değişkeni 0 olduğunda elemanın kendisini yanında yıldız olacak şekilde yazdırdım. Eğerki sayaç 1 değerine atanırsa elemanın kendisini sadece yanında boşluk olacak şekilde yazdırdım. Sonuncu for döngüsünün 2den başlamasının sebebi; tüm sayıların 1'e tam bölüneceği için sayac hep 1 olacak ve asal sayıları tespit edemeyecekti. Bu fonksiyonu oluştururken basit bir hata yüzünden çok zaman kaybettim. Algoritmayı doğru yazdığımı düşündüğüm halde asal sayıları hiçbir şekilde bulmuyordu. Sonrasında bu kodu boş bir proje içerisinde tekrar denerken sayac değerinin sürekli olarak arttığını farkettim. Sebebi ise her döngüden sonra 0'lamayı unutmamdı. For döngüsünün içine sayac=0 komutunu ekleyerek her elemanı test ettikten sonra yeni eleman geldiğinde 0 değerinden başladı ve doğru sonuçları gösterebildi. Bir üstteki fonksiyonda da belirttiğim gibi matrisin bir tablo formunda yazdırabilmek için döngülerin içerisine boş olan println'ler ekledim.

```
public cMatrix multiplyMatrices(cMatrix Multiplicand){ // Multiply two matrices
and informs the user about time lapse

		long[][] carpan = Multiplicand.elements;
		long[][] newmx = new long [elements.length][carpan.length];
		int roww=Multiplicand.row;
		int coll=Multiplicand.col;

		for(int i=0; i< this.row; i++) {
			for(int j=0; j<col1; j++) {
				for(int k=0; k<row1;k++) {
					newmx[i][j] += cMatrix.this.elements[i][k] *
Multiplicand.elements[k][j];


				}
			}
		}


		return new cMatrix(newmx);

		 }
```

Bu fonksiyon main class içerisinden gelen 2 matrisi çarpar. İlk olarak main class içerisinden gelen m3(Multiplicand) carpan adında bir matrise eşitledim. Çarpılan değerleri içerisine yazacağım m4 matrisini newmx adında boyutlarını ise m2 ve m3 matrisine göre tanımladım. Çarpanın row ve col değerlerini for döngüsünün içerisinde kullanabilmek için başka bir değişkene atadım. 2 matrisi çarpmak ve her elemanı yeni bir matrise yazdırmak için 3 adet for döngüsü oluşturdum. `this.elements[i][k]` m2 matrisinin değelerini tutuyor `Multiplicand.elements[k][j];` komutu ise m3 matrisinin değelerini içinde tutuyor. Matris çarpım kuralına göre sırayla iki matrisin [0][0] ve [0][0], [0][1] ve [1][0], [0][2] ve [2][0] elemanlarını çarpıp topla eşittir komutu ile i ve j değişkenlerine sahip olan m4 matrisinin içerisine yazdırdım. [0][col] x [row][0] çarpımındaki tüm elemanların toplamı m4 matrisinde [0][0] elemanına atandı. Matris elemanlarını çarpmak zor olmasa da bu fonksiyonda beni en çok zorlayan kısım multiplicand'ı içeride nasıl

tanımlayacağım ve return kısmının oluşturulması idi. CMatrix tipinde olan bu Constructor'un nasıl bir return'e sahip olacağı çok kafamı karıştırdı. En son olarak `public cMatrix(long [][] newmx)` adında bir Constructor yazarak return komutunu bunun cinsinden yazdım.