DÜSTAD
Dünya Sağlık ve Tabiat Bilimleri Dergisi

JOGHENS
Journal of Global Health & Natural Science

ISSN: 2687-637X Yıl: 2023 Cilt: 6 Sayı: 2

Makale Türü	Geliş Tarihi	Kabul Tarihi
Araştırma	05 Haziran 2023	27 Kasım 2023

Olgun Sayılar ve Olgun Sayılar/Mersenne Sayılar Üreteç Algoritması

Ahmet Sedat KAYA[1]

Özet: Sayılar ve sayı sistemleri ile ilgili yapılan çalışmalar geçmişten günümüze insanlığın dikkatini çekmiş ve insanoğlu sayılar arasındaki ilişkileri anladıkça yeni kavramlar üretmiştir. Sayısal kavramlardan en önemlilerinden birisi de hiç şüphesiz asal sayılardır. Sayılar üzerinde araştırma yapan Marin Mersenne; 17.yy. başlarında mersen sayıları kavramını üretmiş ve sonrasında ise hem mersen hem de asal olan sayılara mersenne asal sayısı denilmiştir. Asal sayılarla ilgili yapılan çalışmaların olası sonuçları başta kriptoloji ve bilgi güvenliği olmak üzere hayatımızı etkileyecek birçok gelişmeyi ortaya çıkarma potansiyeline sahiptir. Bu sebeple başta matematik ve bilgisayar bilimi alanları olmak üzere bu konuda birçok çalışma yapılmasına neden olmaktadır. Bu makale kapsamında, Mersenne asal sayıları üzerinde yapılan çalışmalar esnasında ortaya çıkan olgun sayılar kavramı üretilmiştir. Bu kavrama göre her sayının olgun sayı olmak istediği ve her olgun sayının daha sonra genişleyerek yeni olgunluk seviyesine ulaşacağı varsayılmaktadır. Makalede ayrıca 2'lik sayı sistemi için Olgun Sayı/Mersenne Sayı üreteç algoritması ve olgun sayılar ile ilgili bazı kurallar üretilmiştir.

Anahtar kelimeler: Olgun sayılar, mersenne sayıları, algoritma, numeroloji, sayı sistemleri, collatz varsayımı

Mature Numbers and The Algorithm to Generate Mature Numbers/Mersenne Numbers

Abstract: The Studies on numbers and number systems have attracted the attention of humanity from past to present. As people deepen their understanding of the relationships between numbers. One of the most important numerical terms is undoubtedly the prime numbers. Marin Mersenne, who studied on numbers, generated the term of mersen numbers at the beginning of 17th century and subsequently numbers that are both Mersenne and prime came to be known as Mersenne prime numbers. Later on the numbers that are both mersenne and prime were called as the mersenne prime numbers. The potential outcomes of studies related to prime numbers, especially in the fields of cryptography and information security, have the capacity to bring forth numerous developments that can significantly impact our lives. Therefore, this leads to a considerable amount of research being conducted, particularly in the fields of mathematics and computer science. Within the scope of this article, the term of mature numbers, which emerged during the studies on Mersenne prime numbers, was generated. According to this term, each number aims to be a mature number and later on it is assumed that each mature number will expand and reach a new level of maturity. Furthermore, the article introduces the Mature Numbers/MErsenne Number generator algorithm for the binary system and outlines certain rules goverining mature numbers.

Keywords: Mature number, mersenne number, algorithm, numerology, number systems, collatz assumption

[1]Corresponding author, Ankara/Türkiye, ahmetsedatkaya@gmail.com, 0009-0008-3905-7350

GİRİŞ

Olgun Sayılar Kavramının Ortaya Çıkışı

Bu makale ile ilgili çalışmaların başlangıç yılı 2000'li yıllara dayanmaktadır. 2000'li yılların başında Mersenne Vakfı tarafından düzenlenen GIMPS (Great Internet Mersenne Prime Search) yarışmasında Mersenne Asal Sayılarının bulunması ile ilgili çalışmalar yapılmıştır (Great Internet Mersenne Prime Search, 2023). Bu çalışmalar esnasında Mersenne Asal Sayısının bulunması ile ilgili olarak tümdengelim yöntemi (olası asal sayının tüm sayılara bölünerek asal sayı olup olmadığının tespiti) yerine tümevarım (bir sayı kökünün ileride hangi sayıya evirileceği tespit edilerek olası asal sayı listesinden elenmesi) şeklinde bir yöntem izlenmiştir (Anlı, 2021). İzlenen bu yöntem neticesinde sayı kökleri ile ilgili yapılan çalışmalar sırasında "Olgun Sayılar" kavramı ortaya çıkmış ve bu sayılarla ilgili olarak çalışmalar derinleştirilmiştir. Yapılan çalışmalar sırasında Olgun Sayılar yerine daha önce "Güçlü Sayılar" kavramları da düşünülmüş fakat Olgun Sayılar sürekli genişlediği ve her bir Olgun Sayının başka bir Olgun Sayının kökü olabileceği için "Olgun Sayılar" kavramına karar verilmiştir.

Bu makale kapsamında ortaya atılan Olgun Sayılar kavramı ve geliştirilen algoritma ile matematik alanında yeni bir sayı kavramı ortaya konulmuş ve sayı köklerinden yeni sayıların üretilmesi, geliştirilen algoritmanın tersine kullanımı ile sayıların köklerinin olup olmadığının tespiti, sayıların asal sayı olup olmadığı ile ilgili yeni çalışmaların yapılmasına zemin hazırlanmıştır. Asal sayılar matematik ve kriptoloji olmak üzere pek çok alanda kullanılmaktadır (Çelik, 2022). Asal sayılarla ilgili olarak yapılan çalışmalar ve olası sonuçları başta güvenlik olmak üzere bir çok alanda yeni gelişmeleri tetikleme potansiyeline sahiptir.

Olgun Sayılar Kavramının tersi bir bakış açısı da Collatz varsayımında bulunmaktadır (Conway, 1972). Collatz varsayımı tüm sayıların 1'e indirgenebileceğini savunurken (Özkenar, 2020), Olgun Sayılar kavramı tüm sayıların kendi Olgunluk seviyelerine ulaşacağını düşünmektedir.

Nümerolojik (Dudley, 1997) açıdan bakıldığında, nasıl ki içinde bulunduğumuz evren yapılan bilimsel çalışmalara göre genişliyorsa (Linder, 2023) ve insanlık her yaşadığı dönem içerisinde sürekli iyiyi arıyor ve olgunlaşıyorsa sayı sisteminde de benzer yaklaşımların olduğu ve bu kapsamda sayılarında tıpkı insan, insanlık/medeniyet gibi bir sonraki olgunluk seviyesine ulaşmak isteyecekleri düşünülmüştür.

Olgun Sayı Nedir

Olgun sayılar bulundukları sayı sistemlerine göre değerlendirilir. Eğer bir sayı haneleri bazında içinde bulunduğu sayı sisteminin en yüksek rakamı ile gösteriliyorsa bu sayı olgun sayıdır. Eğer gösterilmiyorsa bu sayı olgun sayı değildir. Bilindiği üzere Matematik'de çeşitli sayı sistemleri vardır. (Shahid, 2011) Bunlardan en bilinenleri 2'lik sayı sistemleri (Binary), Onluk Sayı Sistemleri (Decimal) ve 16'lık Sayı Sistemleridir (Hexadecimal). Olgun sayı kavramı tüm sayı sistemleri için geçerli olmak üzere en çok bilinen bu sayı sistemleri üzerinden örneklendirilmiştir.

Olgun Sayı= Bir sayı gösterildiği sayı sisteminin en yüksek rakamı ile ifade ediliyorsa o sayı olgun sayıdır.

Mature Number = IF Number's All Digits Have Maximize Number According to Number Numerical System

Örneğin;

Olgun sayılar kavramını daha iyi açıklamak için çeşitli sayı sistemlerinden örnekler verilmiştir.

2'lik sayı sisteminde 4 haneli / basamaklı iki sayı seçilmiştir (Kovács, 2001).

$1111_{(2)}$ → Bu sayı tüm haneleri (digitleri) 2'lik sayı sistemindeki en yüksek rakamı olan 1 ile ifade edildiği için bu sayı olgun sayıdır.

$1011_{(2)}$→ Bu sayı ise tüm haneleri 1 ile gösterilmediği için olgun sayı değildir.

10'luk sayı sisteminde 7 haneli iki sayı seçilmiştir (Sarton, 1950).

$9999999_{(10)}$→ Bu sayının tüm haneleri onluk sayı sistemindeki en yüksek rakam olan 9 ile ifade edildiği için bu sayı olgun sayıdır.

$1234567_{(10)}$→ Bu sayı ise tüm haneleri 9 ile gösterilmediği için olgun sayı değildir.

16'lık sayı sisteminde 5 haneli iki sayı seçilmiştir (Mahat, 2021).

$FFFFF_{(16)}$→ Bu sayı tüm haneleri 16'lık sayı sisteminde en yüksek rakam olan F ile ifade edildiği için bu sayı olgun sayıdır.

$F0AB1_{(16)}$→ Bu sayı ise tüm haneleri F olmadığı için olgun sayı değildir.

Bu örnekler diğer sayı sistemleri için çoğaltılabilir. Görüldüğü üzere Olgun Sayılar bulundukları sayı sistemlerinde gösterildikleri haneler bazında değerlendirilir. Eğer sayının bütün haneleri ilgili sayı sisteminin en yüksek değeri ise Olgun Sayıdır değilse Olgun Sayı değildir.

Mersenne Sayısı ve Olgun Sayı İlişkisi

Mersenne Sayıları $=2^n-1$ olarak ifade edilir (Robinson, 1954).

Bu kapsamda bir Mersenne Sayısı 2'lik sayı sistemi için aynı zamanda bir Olgun Sayıdır. Ama Olgun Sayılar kavramı sadece 2'lik sayı sistemi için değil tüm sayı sistemleri için geçerlidir.

Örneğin,

$7_{(10)}$ sayısı bir Mersenne sayısıdır.

$7_{(10)}=2^3-1$ olarak ifade edilir ve ikilik sayı sisteminde bakıldığında ise $111_{(2)}$ olarak gösterilir.

Örneğin,

n yerine 10 koyarak yeni bir Mersenne sayısı üretilmiştir.

$2^{10}-1=1023_{(10)}=1111111111_{(2)}$ olarak gösterilir. Bu örnekte görüldüğü gibi tüm mersenne sayıları tanımları itibariyle 2'lik sayı sistemleri açısından Olgun sayılardır fakat diğer sayı sistemleri için Olgun sayılar olmayabilir (Örneğin $1023_{(10)}$ sayısı 10'luk sayı sistemi için Olgun Sayı değildir).

YÖNTEM

2'lik sayı sistemleri için Olgun Sayı/Mersenne Sayısı üretme algoritması oluşturulmuştur. Geliştirilen bu algoritma kendisine verilen 2'lik sayı sisteminden sayı kökünü kullanarak yeni bir olgun sayı üretmektedir. Bu sayede herhangi bir 2'lik sayı sistemindeki sayının bir sonraki Olgun Sayısı belirlenebilmekte ve oluşan sayı tekrar bu algoritmaya girdi teşkil ederek yeni Olgun Sayılar üretilebilmektedir.

Algoritmanın çalışma mantığı Collatz varsayımında bahsedilen algoritmanın tersi bir mantıkla çalışmaktadır. Collatz varsayımındaki algoritma kendisine verilen tüm sayıları 1'e indirgemekte, Olgun Sayılar/Mersenne Sayılar Algoritması ise kendisine verilen 2'lik sayı sistemindeki sayıyı bir sonraki Olgunluk seviyesine çıkarmaktadır.

Algoritma tümevarım yöntemi ile çalışmaktadır. İleride yapılacak çalışmalarda tersine-mühendislik yöntemleri kullanılabilir. Bu sayede tümden gelim yöntemi kullanılarak sayıların köklerinin neler

olabileceği kapsamında çalışmalar derinleştirilebilir. Bu çalışmaların çıktıları bir sayının asal olup olmadığının kontrolü kapsamında kullanılabilir.

Algoritmanın daha iyi anlaşılması için öncelikle algoritmanın çalışma mantığı aktarılmış sonrasında ise algoritmanın fonksiyonel gösterimi yapılmıştır. Bu algoritma 2'lik sayı sistemine göre en düşük seviyeli bitten (LSB) en yüksek seviyeli bite doğru (MSB) ilerler (Yüce, 2014).

Algoritmanın Çalışma Mantığı

2'lik sayı sistemi için geliştirilen Olgun Sayılar algoritmasının çalışma mantığı aşağıda adımlar halinde açıklanmaktadır. Sayı N olarak alınır.

1. Sayının Tek veya Çift olduğuna bakılır. Eğer sayı Çift ise 1 eklenerek sayı tek haline getirilir.
2. Daha sonra bu sayı en düşük seviyeli bitten (LSB) en yüksek seviyeli bite doğru(MSB) döngüye sokulur. $3_{(10)} \rightarrow 11_{(2)}$ ve $5_{(10)} \rightarrow 101_{(2)}$ sayıları örnek sayı olarak belirlenmiştir.

   $3_{(10)}$ sayısına göre İşlem Yönü — $5_{(10)}$ Sayısına Doğru İşlem Yönü

   $11_{(2)}$ — $101_{(2)}$

   ← — ←

3. Ayrıca sayının ilk hali StartN değişkenine atılır ve Index değişkeni sayının uzunluğu olacak şekilde ve en düşük seviyeli biti gösterecek şekilde ayarlanır. Bir adet Counter değişkeni oluşturularak N sayısının ilgili basamağını bulmak için kullanılır.

   $11_{(2)}$ — $101_{(2)}$

   StartN=11 — StartN=101

   Counter=0 — Counter=0

4. Daha sonra en düşük seviyeli bitten (LSB) başlayarak basamak basamak ilerlenir. Öncelikle işlem yapılacak Index değeri hesaplanır. Eğer N sayısının dizi Index değeri "1" ise sayının en yüksek bitine(MSB) "0" eklenir. Değilse yani Eğer N sayının dizi Index değeri "0" ise StartN değeri ile toplanır. Bu kontrol yapıldıktan sonra StartN değişkeni sayısal olarak 2 ile çarpılır veya String olarak LSB tarafına "0" eklenir ve Counter değeri 1 arttırılır.

   Index yeni Hali

   Index=Length(N)-Counter=2-0=2 — Index=Lengh(N)-Counter=3-0=3

   11[Index]→11[2]→"1" — 101[Index]→101[3]→"1"

   N yeni hali

   $11_{(2)} \rightarrow 011_{(2)}$ — $101_{(2)} \rightarrow 0101_{(2)}$

   StartN yeni hali

   $11_{(2)} \rightarrow 110_{(2)}$ — $101_{(2)} \rightarrow 1010_{(2)}$

   Counter yeni Hali

   Counter=Counter+1→0+1=1 — Counter=Counter+1→0+1=1

5. Daha sonra N değişkeninin Olgun Sayı olup olmadığı kontrol edilir (2'lik sayı sistemi için tüm basamaklarının 1 olduğu). Eğer Olgun Sayı değil ise diğer basamakları kontrol etmek için döngüye devam edilir. Eğer Olgun Sayı ise döngüden çıkılır.

   IF N is Mature Number

   Then return/break;

   Else Continue;

IsMature(N=$001_{(2)}$)=False→Continue IsMature(N=$1010_{(2)}$=False→Continue

6. Bir sonraki basamağa geçilir ve tekrar 5. basamaktaki işlemler tekrar edilir. Yeni Index değeri hesaplanır ve N değişkeninin dizi Index değeri kontrol edilir.

Index yeni Hali
Index=Length(N)-Counter=3-1=2
011[Index]→011[2]→"1"

Index=Lengh(N)-Counter=4-1=3
0101[Index]→0101[3]→"0"

Bu durumda sol taraftaki blokta "1" değeri sağ taraftaki blokta ise "0" değeri ile karşılaşılmıştır. Daha net anlaşılması için ayrı ayrı işlem yapılacaktır.

Sol taraf şu şekilde işlenecektir. "1" olduğu için.

N yeni hali
$011_{(2)}$ → $0011_{(2)}$

StartN yeni hali
$110_{(2)}$→$1100_{(2)}$

Counter yeni hali
Counter=Counter+1→1+1=2

Sağ taraf şu şekilde işleyecektir. "0" olduğu için.

N Yeni Hali
N=N+StartN
N=0101+1010=1111

StartN yeni hali
$1010_{(2)}$→$10100_{(2)}$

Counter yeni Hali
Counter=Counter+1→1+1=2

Daha sonra N değişkeninin Mature olup olmadığı kontrol edilir.

IsMature(N=$0011_{(2)}$)=False→Continue

Sol taraf için döngüye devam edilir.

IsMature(N=$1111_{(2)}$=True→Break

Sağ taraf için Olgun Sayıya ($1111_{(2)}$) ulaşıldığı için döngüden çıkılır.

7. Sol taraf için döngüye devam edilir.

Index yeni Hali
Index=Length(N)-Counter=4-2=2
0011[Index]→0011[2]→"0"

N Yeni Hali
N=N+StartN
N=0011+1100=1111

StartN yeni hali
$1100_{(2)}$→$11000_{(2)}$

Counter yeni Hali

Counter=Counter+1→2+1=3
IsMature(N=1111)→True=Break

N değişkeni Olgun Sayıya ulaştığı için döngüden çıkılır.

Algoritma çalıştırıldığında görüldüğü üzere başlangıçta $3_{(10)}$→$11_{(2)}$ sayısı Olgun Sayı olmasına rağmen bir sonraki Olgunluk Sayısı olan $15_{(10)}$ →$1111_{(2)}$ değerine de ulaşmıştır.

Aynı şekilde $5_{(10)}$ →$101_{(2)}$ sayısı Olgun Sayı olmamasına rağmen bir sonraki Olgunluk Sayısı olan $15_{(10)}$ →$1111_{(2)}$ değerine ulaşmıştır. Bu da bize Numerolojik açıdan her sayının Olgun Sayıya ulaşmak istediğini ve her Olgun Sayının bir sonraki Olgunluk seviyesine ulaşabildiğini göstermektedir.

Algoritmanın Fonksiyonel Gösterimi

Algoritmanın fonksiyon hali gösterilmiştir.

```
Function GenerateMatureandMersenneNumber(N)

Begin
  IF N is EVEN
      Then N=N+1
  index=0
  StartN=N
  Counter=0
  Repeat
      Index=Length(N)-Counter
          IF (N[Index]==”1”)
          Then
              N=”0”+N
          IF (N[Index]==”0”)
          Then
              N=N+StartN
      StartN=StartN+”0”
      Counter=Counter+1
  Until (IsMature(N));
  Return N;
End
```

BULGULAR

Test Ortamı

Olgun Sayılar ve Mersenne Sayılar için hazırlanan Algoritmanın çalıştırıldığı test ortamına ilişkin donanım, işletim sistemi Tablo 1'de, yazılım, geliştirme ortamı ve kullanılan dil bilgileri ise Tablo 2'de verilmiştir.

Tablo 1. Algoritmanın çalıştığı test ortamına ilişkin donanım ve işletim sistemi bilgileri.

CPU	Intel(R) Core (TM) i7-4700HQ CPU -2.40GHz
RAM	16 GB
HDD	500 GB SSD
Operating System	Windows 10

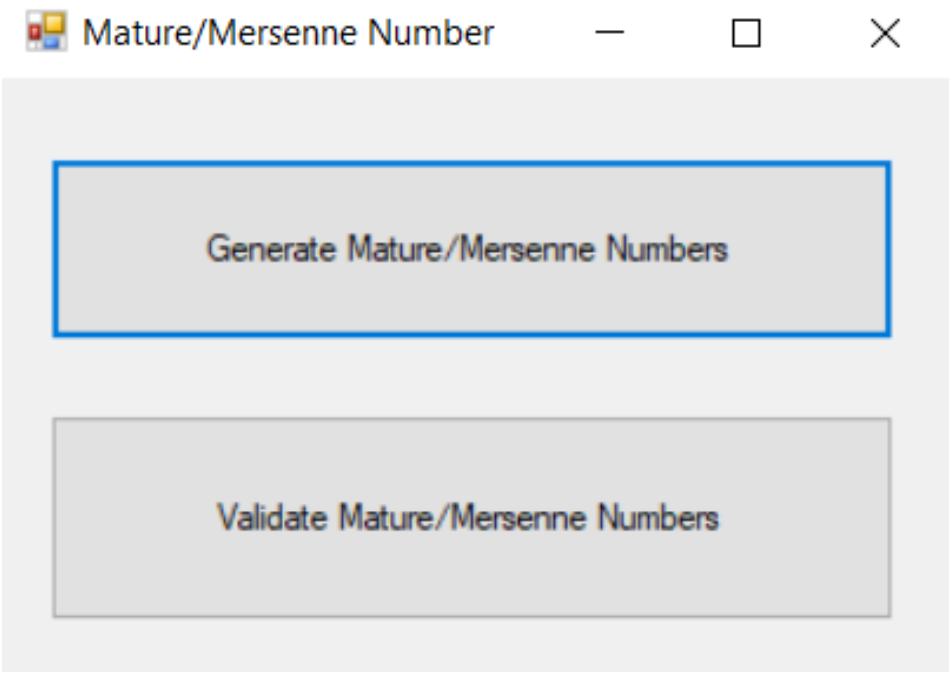
Tablo 2. Algoritmanın çalıştığı test ortamına ilişkin framework, geliştirme ortamı ve dili bilgileri.

Framework	Microsoft.Net Framework 4.8
Geliştirme Ortamı	Microsoft Visual Studio 2019
Geliştirme Dili	C#

Algoritma Geliştirme Yöntemi

Algoritma geliştirilirken C# dili tercih edilmiştir (Hejlsberg, 2008). Bazı Olgun Sayıların milyon basamaklı olduğu görüldüğünden dolayı işlemler sayısal toplama yerine (sayısal değişken tipleri maksimum 64 bit örneğin Int64 olduğundan) String (karakter dizisi) kullanılarak yapılmıştır (String Computer Science, 2023).

Mature/Mersenne Sayıların üretilmesi ve doğrulanması için bir uygulama geliştirilmiştir (Şekil 1).

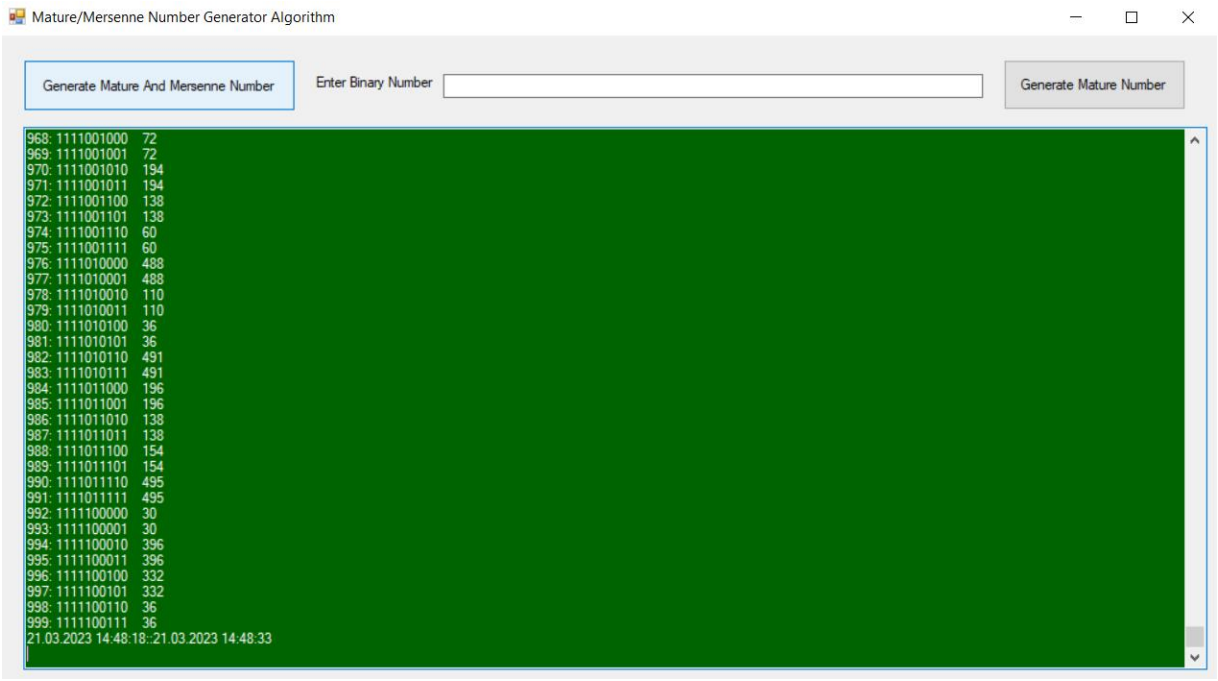


Şekil 1. Olgun/Mersenne Sayıları üretilmesi ve doğrulanması için geliştirilen uygulama.

Uygulama iki formdan oluşmaktadır. Mature/Mersenne sayılarının üretildiği "Generate Mature/Mersenne Numbers" formu diğeri ise üretilen Mature/Mersenne sayılarının doğrulandığı "Validate Mature/Mersenne Numbers formudur.

Mature/Mersenne Sayılarının Üretilmesi

Şekil 2'de ekran görüntüsü verilen "Mature/Mersenne Number Generator Algorithm" formu sayesinde girilen herhangi bir 2'lik sayı sistemindeki (binary) sayı Olgun hale getirilebilmekte veya $1\text{-}1000_{(10)}$ arasındaki sayılar kullanılarak otomatik olarak Olgun sayıların üretilmesi sağlanmaktadır.



Şekil 2. Olgun/Mersenne Sayıları üretim sonuçları.

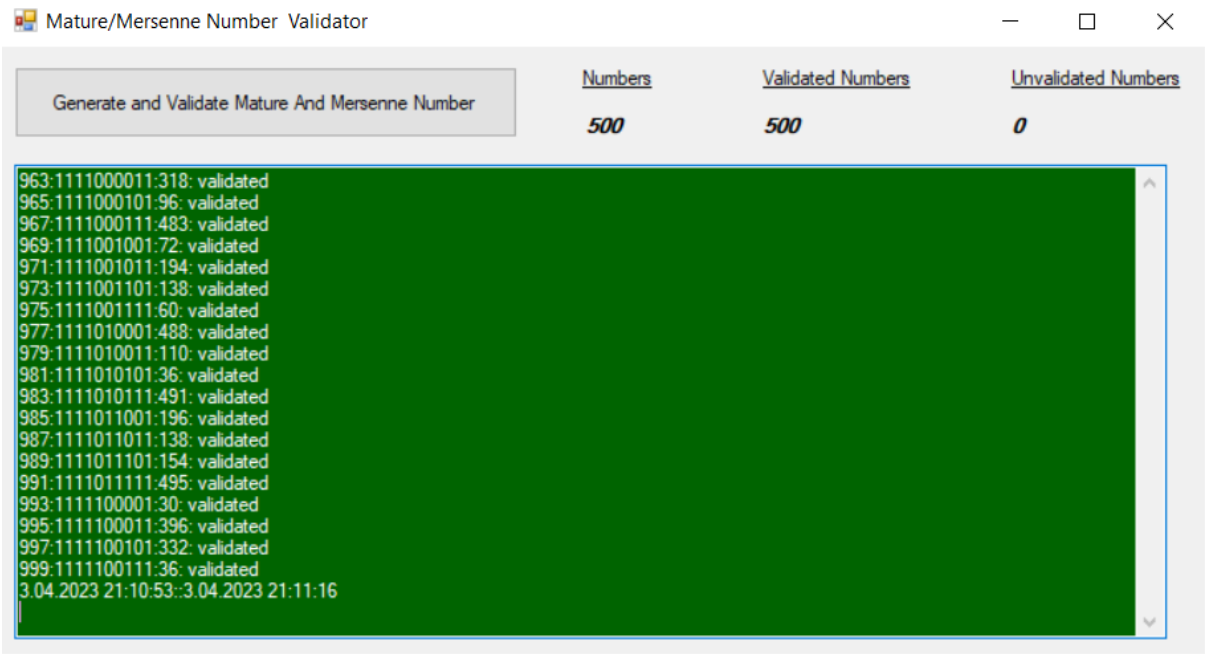
Burada çıktı gösterimi şu şekilde özetlenebilir.
999: 1111100111 36
999: N sayısının 10'luk sayı sisteminde gösterimidir.
1111100111: N sayısının 2'lik sisteminde gösterimidir.
36: Oluşan Olgun Sayı Mersenne Sayı Formatında gösterimidir. $2^{36}-1$
21.03.2023 14:48:18::21.03.2023 14:48:33 : Algoritmanın çalıştırıldığı ve tamamlandığı andaki tarihsel gösterimidir. Buna göre $1_{(10)}$ den $1000_{(10)}$ e kadar olan sayıların Olgun sayı haline gelmesi test ortamında yaklaşık 15 saniye sürmüştür.

Algoritma Sonuçlarının Test Edilmesi

Her bir Olgun sayı bir sayı kökünden türetilmektedir. Bundan dolayı üretilen Olgun sayının kök sayıya kalansız bölünmesi gerekmektedir. Bu kapsamda her bir üretim sonrasında sayısal olarak üretilen Olgun Sayı kök sayıya bölünerek sonucun doğruluğu kontrol edilmiştir. Bu kapsamda 2'lik sayı sisteminde bölme işlemi yapılmış ve bölme yöntemi olarak Long Division yöntemi kullanılmıştır (Korkmaz, 2021).

"Mature/Mersenne Number Validator" formu sayesinde $1\text{-}1000_{(10)}$ arasındaki tek kök sayılar kullanılarak öncelikle Olgun Sayıları üretilmekte daha sonra üretilen Olgun Sayılar kök sayılara bölünerek doğrulanması sağlanmaktadır.

Formdaki "Generate and Validate Mature and Mersenne Number" butonuna basıldığında Şekil 3'teki ekran görüntüsü oluşmaktadır.



Şekil 3. Üretilen Olgun/Mersenne Sayılarının doğrulanması.

Numbers (500): 1 -$1000_{(10)}$ sayıları arasındaki tek sayı köklerinin sayısını gösterir.
Validated Numbers(500): Bu tek sayı kökleri ile oluşturulan Mature/Mersenne numberların kaç tanesinin doğrulandığını gösterir.
Unvalidated Numbers(0): Doğrulanmayan sayıların kaç adet olduğunu gösterir.
999: 1111100111 36 validated
999: N sayısının 10'luk sayı sisteminde gösterimidir.
1111100111: N sayısının 2'lik sisteminde gösterimidir.
36: Oluşan Olgun Sayı Mersenne Sayı Formatında gösterimidir. $2^{36}-1$
Validated: Sayının doğrulandığını göstermektedir.
03.04.2023 21:10:53::03.04.2023 21:11:16: Algoritmanın çalıştırıldığı ve tamamlandığı andaki tarihsel gösterimidir. Buna göre $1_{(10)}$ den $1000_{(10)}$ e kadar olan sayıların Olgun sayı haline gelmesi ve doğrulanması test ortamında yaklaşık 23 saniye sürmüştür.

TARTIŞMA ve SONUÇ

Olgun Sayı Kuralları

Makale kapsamında 2'lik sayı sistemine göre yapılan çalışmalar kapsamında bazı sayıların olgun sayılara dönüşürken bazı kurallara göre dönüştüğü gözlemlenmiştir. Bu kurallar üzerinden çeşitli formüller oluşturularak sadece sayı kökü ve kural setleri kullanarak sayıların ne zaman olgun sayı olacağı algoritma kullanmadan bulunabilmektedir. Belirtilen kuralların test edilmesi amacıyla her bir kural için kurala uygun 1000 adet 2'lik sayı sisteminden sayılar seçilmiş ve kurallar test edilmiştir. Yapılan test sonucunda kurallara uygun olmayan herhangi bir sonuç ile karşılaşılmamıştır.

Kural1:

Eğer bir sayı Olgun Sayı ise, bir sonraki Olgun Sayının basamak uzunluğu Olgun Sayının basamak uzunluğunun iki katıdır.

IF(N=Mature(N) THEN NextMatureLength=Length(N)*2

Örnek1:

$11_{(2)}$ → Length(N)=2 → NextMatureLength=Length(N)*2=2*2=4 → Next Mature Number=$1111_{(2)}$

$111_{(2)}$ → Length(N)=3 → 3*2=6 → Next Mature Number=$111111_{(2)}$

$1111_{(2)}$ → Length(N)=4 →4*2=8 → Next Mature Number=$11111111_{(2)}$

$11111_{(2)}$ → Length(N)=5 →5*2=10 → Next Mature Number=$1111111111_{(2)}$

Kural2:

Eğer bir sayı "1" ile başlıyor ve "1" ile bitiyor ve diğer tüm haneleri "0" ise o zaman oluşacak olgun sayının hane sayısı sayının hane sayısına "0" ların sayısı eklenerek bulunur.

IF(N is Start equal "1" and End equal "1" and Others equal "0") THEN MatureLengh=Length(N)+Count("0")

Örnek2:

$101_{(2)}$→MatureLength=Length(N)+Count("0")→MatureLength=3+1=4→$1111_{(2)}$

$1001_{(2)}$→ MatureLength=4+2=6→$111111_{(2)}$

$10001_{(2)}$→ MatureLength=5+3=8→$11111111_{(2)}$

$100001_{(2)}$→ MatureLength=6+4=10→$1111111111_{(2)}$

Kural3:

Eğer bir sayının hanelerini oluşturan "1" ler ve "0" lar arasında bir eşitlik örüntüsü var ise oluşacak olgun sayısı hane sayısına örüntü uzunluğu (her bir eşitin uzunluğu) eklenerek bulunur.

IF (N has Pattern between "1" and "0") THEN MatureLength=Length(N)+Length(Pattern)

Örnek3:

$1100110011_{(2)}$→MatureLength=Length(N)+Length(Pattern)→MatureLength=10+2=12→$111111111111_{(2)}$

$111000111000111_{(2)}$→MatureLength=15+3=18→$111111111111111111_{(2)}$

$11110000111100001111_{(2)}$→MatureLength=20+4=24→$11111111111111111111111111_{(2)}$

$11111000001111100000111111_{(2)}$→MatureLength=25+5=30→$1111111111111111111111111111111111_{(2)}$

Sonuç ve Öneriler

Bu makale kapsamında Olgun Sayılar kavramı ortaya konulmuş, Mersenne ve Olgun Sayılar arasındaki ilişki gösterilmiş ve aynı zamanda Olgun Sayı olan Mersenne sayıları için 2'lik sayı sisteminde çalışan bir üreteç algoritması oluşturulmuştur.

Ayrıca 2'lik sayı sistemi için Olgun Sayılarla ilgili olarak kurallar oluşturularak Olgun Sayıların üretimi için çeşitli yöntemler önerilmiştir.

Bu kapsamda bundan sonra yapılabilecek çalışmaları aşağıdaki şekilde özetleyebiliriz.

- Bu makale kapsamında 2'lik sayı sistemi için yapılan üreteç algoritmasının diğer sayı sistemlerine uygulanması.
- Olgun Sayı kurallarının zenginleştirilmesi ve diğer sayı sistemlerine uygulanması.

- Bu makale çalışmasının başlangıcı olan Mersenne Asal Sayıları ve Asal sayılar kapsamında çalışmalar yapılarak, Olgun Sayıların genişleme yöntemi tersine-mühendislik bakış açısıyla değerlendirilip bir sayıya bakıldığında onun herhangi bir sayı kökü ve/veya böleni olup olmadığı konusunda çalışmaların yapılması.
- Olgun Sayılar üzerinde yapılan çalışmalar kapsamında bazı sayıların Olgun hale gelmesinin yakın diğer sayılara göre daha uzun sürede ve basamakta/hanede ($>2^{1000000}$) olgunlaştığı gözlemlenmiştir. Bu sayı köklerinin saklanarak daha sonra geliştirilecek çalışmalar kapsamında sayı kökleri ve Olgunlaşmaları arasında yeni algoritmalar geliştirilmesi.
- Olgun Sayılar ve sayı kökleri ile ilgili olarak yeni yöntem ve metotların geliştirilmesi.
- Olgun Sayılar üzerinde yapılan çalışmalar kapsamında bazı çift sayılarının Olgunlaştığı zaman (Not: Algoritma kapsamında Çift sayılar öncelikle 1 eklenerek tek sayı haline getirilir.) ulaştığı hane sayının sayının 10'luk(Decimal) karşılığı olduğu gözlemlenmiştir. Bu kapsamda sayı kökleri ile bir bağlantının veya kuralın olup olmadığı konusunda çalışmaların derinleştirilmesi.

Örneğin,

$6_{(10)} \rightarrow 110_{(2)} \rightarrow$ MatureLength=Decimal(N)=6$\rightarrow 111111_{(2)}$

$10_{(10)} \rightarrow 1010_{(2)} \rightarrow$ MatureLength=Decimal(N)=10$\rightarrow 1111111111_{(2)}$

$12_{(10)} \rightarrow 1100_{(2)} \rightarrow$ MatureLength=Decimal(N)=12$\rightarrow 111111111111_{(2)}$

$18_{(10)} \rightarrow 10010_{(2)} \rightarrow$ MatureLength=Decimal(N)=18$\rightarrow 111111111111111111_{(2)}$

Bu kapsamda $1000_{(10)}$'e kadar belirlenen 10'luk sayı sistemindeki sayılar şunlardır.

6, 10, 12, 18, 28, 36, 52, 58, 60, 66, 82, 100, 106, 130, 138, 148, 162, 172, 178, 180, 196, 210, 226, 268, 292, 31 6, 348, 372, 388, 418, 420, 442, 460, 466, 490, 508, 522, 540, 546, 556, 562, 586, 612, 618, 652, 658, 660, 676, 700, 708, 756, 772, 786, 796, 820, 826, 828, 852, 858, 876, 882, 906, 940, 946.

KAYNAKÇA

Anlı, Ö. F. (2011). Aristoteles ve Yöntem-Tümevarım ve Tümdengelim. Ankara Üniversitesi DTCF Felsefe Bölümü, Bilim Tarihi Anabilim Dalı Ders Notu.

Yuce, B., Uğurdağ, H.F., Gören, S. & Dündar, G. (2014). Fast and Efficient Circuit Topologies forFinding the Maximum of n k-Bit Numbers. *IEEE Transactions on Computers, 63*(8), 1868-1881. https://doi.org/10.1109/TC.2014.2315634.

Conway, J.H. (1972). Unpredictable Iterations. *Proceedings. Number Theory Conference. University of Colorado, S.U.A,* 49-52.

Çelik, K. C. (2022). Çelik Asal Sayılar. *Dünya Sağlık ve Tabiat Bilimleri Dergisi, 5(2),76-80.* https://doi.org/10.56728/dustad.1179688

Dudley, U. (1997). Numerology, or, what Pythagoras wrought. *Cambridge University Press*.

Great Internet Mersenne Prime Search. (2023). https://www.mersenne.org/primes

Hejlsberg, A., Torgersen, M., Wiltamuth, S., & Golde, P. (2008). The C# programming language. *Pearson Education.*

Korkmaz, E. (2021). Instructional Explanations of Class Teachers and Primary School Mathematics Teachers about Division. *International Journal of Progressive Education, 17*(2), 29-54.

Kovács, A. (2001). Generalized binary number systems. *In Annales Univ. Sci. Budapest, Sect. Comp., 20,* 195-206.

Linder, E. V. (2003). Exploring the expansion history of the universe. *Physical review letters, 90*(9), 091301.

Mahat, M. S. S. (2021). Number System Conversion for Beginners (Decimal to Binary, Octal and Hexadecimal Conversion). *Turkish Journal of Computer and Mathematics Education (TURCOMAT), 12*(14), 1445-1458.

Robinson, R. M. (1954). Mersenne and Fermat numbers. *Proceedings of the American Mathematical Society 5*(5), 842-846.

Özkenar, M. (2020). Collatz Konjektürü'nün bilgisayar programı ile hesaplanmasında parite sekansı yöntemi yaklaşımı. *Acta Infologica, 4*(2), 97-121. https://doi.org/10.26650/acin.843275

Sarton, G. (1950). Decimal systems early and late. *Osiris, 9*, 581-601.

Latif, S., Qayyum, J., Lal, M., & Khan, F. (2011). Complete description of well-known number systems using single table. *International Journal of Engineering and Computer Science (IJECS-IJENS), 11*(3).

String Computer Science. (2023). https://en.wikipedia.org/wiki/String_(computer_science).

How to cite this article/Bu makaleye atıf için:

Kaya, A. S. (2023). Olgun sayılar ve olgun sayılar/mersenne sayılar üreteç algoritması. *DÜSTAD-Dünya Sağlık ve Tabiat Bilimleri Dergisi*, *6*(2), 52-63. https://doi.org/10.56728/dustad.1310123