Atkin Kalburu (Sieve of Atkin)

Hazırlayanlar: Ali Emre Nebiler – 19011070

Yıldız Teknik Üniversitesi, Bilgisayar Mühendisliği,
1. Sınıf, Yapısal Programlamaya Giriş Dersi

Algoritma Tarifi ve Çalışma Prensibi

Asal Sayılar Hakkında Genel Bilgi

Asal sayılar, sadece kendisine ve 1 sayısına bölünebilen, 1'den büyük pozitif tam sayılardır. En küçük asal sayı 2'dir ve asal sayılar sonsuza dek devam ederler.

Asırlardır asal sayılar üzerinde birçok teorem ortaya atılmış ve ispat edilmiştir. Sonsuza dek devam eden bu dizinin belirlenmesi için çeşitli formüller üretilmeye çalışılmış, fakat bunların hiçbiri bir sonuca varamamıştır. Çünkü asal sayılar arasında belirli bir kural bulunmamaktadır.

Asal sayılar hakkındaki pek çok soru günümüzde hâlâ cevaplanamamaktadır.

Atkin Kalburu

Asal sayıları hesaplamaya yönelik birçok algoritma mevcuttur. Bunlardan biri de Atkin Kalburu'dur. Bu algoritma, yine bir tür asal sayı bulma algoritması olan Eratosthenes Kalburu algoritmasının Arthur Oliver Lonsdale Atkin tarafından geliştirilmiş halidir.

Nasıl Çalışır?

İlk olarak, asal sayı bulmak istenen aralığın üst limiti belirlenir. Üst limit belirlendikten sonra, belirlenen aralıktaki tüm sayıları içeren bir liste oluşturulur ve listedeki tüm sayılar "asal değil" olarak işaretlenir.

Bu ön hazırlıktan sonra sayılara işlemler uygulanarak asal olanları "asal" olarak işaretlenir.

- Yapılacak işlemlere 2, 3 ve 5 sayıları dahil olmadığından, bu sayılar "asal" olarak işaretlenir.
- Geriye kalan sayıların mod 60 değerleri aşağıdaki kurallara göre incelenir.

  Eğer mod 60 değeri:
  - {1,13,17,29,37,41,49,53} değerlerinden biriyse, $4x^2+y^2 = n$ sonucunu tek sayıda (x,y) ikilisi ile veren bütün çözümler "asal" olarak işaretlenir.
  - {7,19,31,43} değerlerinden biriyse, $3x^2+y^2 = n$ sonucunu tek sayıda (x,y) ikilisi ile veren bütün çözümler "asal" olarak işaretlenir.
  - {11,23,47,59} değerlerinden biriyse, $3x^2-y^2 = n$ sonucunu tek sayıda (x,y) ikilisi ile veren ve x>y olan bütün çözümler "asal" olarak işaretlenir.

  (Buradaki n değeri mod 60 sonucu elde edilen değer, x ve y değerleri de herhangi bir pozitif tam sayıdır.)

Böylece asal olan tüm sayılar işaretlenmiş olur.

Bu algoritmanın genelleştirilebilmesi için verilen kurallar incelenirse:

- {1,13,17,29,37,41,49,53} sayıları mod 12'de 1 veya 5 değeri veren sayılardır,
- {7,19,31,43} sayıları mod 12'de 7 değeri veren sayılardır,
- {11,23,47,59} sayıları mod 12'de 11 değeri veren sayılardır denebilir.

Buna göre:

- mod 12'de 1 veya 5 değeri veren ve tek sayıda (x,y) ikilisi ile $4x^2+y^2 = n$ sonucunu veren bütün çözümler "asal" olarak işaretlenir.
- mod 12'de 7 değeri veren ve tek sayıda (x,y) ikilisi ile $3x^2+y^2 = n$ sonucunu veren bütün çözümler "asal" olarak işaretlenir.
- mod 12'de 11 değeri veren ve x>y için tek sayıda (x,y) ikilisi ile $3x^2-y^2 = n$ sonucunu veren bütün çözümler "asal" olarak işaretlenir.

Son olarak, tüm işaretlemelerden sonra oluşabilecek istisnaları düzeltmek için, "asal" olarak işaretlenen sayıların kareleri ve karelerinin katları "asal değil" olarak işaretlenir.

Yapılacak işlemleri daha da hızlandırmak ve sistematikleştirmek için, aralıktaki tüm sayıları incelemek yerine x ve y değerleri incelenebilir. Yani x=1 ve y=1 değerlerinden başlanarak, her adımda x veya y değerlerini bir arttırarak tüm (x,y) ikililerine göre oluşan durumlar incelenebilir.

Sağlanan değerler için işaret değişikliği yapılır ("asal" ise "asal değil", "asal değil" ise "asal").

Bunu bir örnek üzerinde inceleyelim.

Örnek

40 sayısına kadar olan asal sayıları Atkin Kalburu ile bulalım.

x=1 ve y=1 durumu ile çalışmaya başlıyoruz.

- $4x^2+y^2 = n$ denklemi için n=5 bulunur. Bu değer mod 12'de 5'tir ve ilk kümeye girer. Dolayısıyla 5 sayısı için işaret değişikliği yapılır.
- $3x^2+y^2 = n$ denklemi için n=4 bulunur. Bu değer mod 12'de 7 olmadığı için bir işlem yapılmaz.
- $3x^2-y^2 = n$ denkleminde n=2 bulunur ve mod 12'de 11 olmadığı için bir işlem yapılmaz.

Yukarıdaki sorgulama işlemini x ve y değerlerini arttırarak tekrarlıyoruz.

x=1 ve y=2 için işlemler yapıldığında sadece ikinci denklem sağlanır:

- $3x^2 + y^2 = n$ denklemi için n=7 bulunur. Bu değer de mod 12'de 7 olduğu için 7 sayısı için işaret değişikliği yapılır.

Tüm sayılar için oluşan değerler aşağıdaki tablodadır:

x	y	$4x^2+y^2$	$4x^2+y^2$ mod 12	$3x^2+y^2$	$3x^2+y^2$ mod 12	$3x^2-y^2$	$3x^2-y^2$ mod 12
1	1	5	5	4	4	2	2
1	2	8	8	7	7	-1	-1
1	3	13	1	12	0	-6	-6
1	4	20	8	19	7	-13	-1
1	5	29	5	28	4	-22	-10
1	6	40	4	39	3	-33	-9
2	1	17	5	13	1	11	11
2	2	20	8	16	4	8	8
2	3	25	1	21	9	3	3
2	4	32	8	28	4	-4	-4
2	5	41	5	37	1	-13	-1
2	6	52	4	48	0	-24	0
3	1	37	1	28	4	26	2
3	2	40	4	31	7	23	11
3	3	45	9	36	0	18	6
3	4	52	4	43	7	11	11
3	5	61	1	52	4	2	2
3	6	72	0	63	3	-9	-9
4	1	65	5	49	1	47	11
4	2	68	8	52	4	44	8
4	3	73	1	57	9	39	3
4	4	80	8	64	4	32	8
4	5	89	5	73	1	23	11
4	6	100	4	84	0	12	0
5	1	101	5	76	4	74	2
5	2	104	8	79	7	71	11
5	3	109	1	84	0	66	6
5	4	116	8	91	7	59	11
5	5	125	5	100	4	50	2
5	6	136	4	111	3	39	3
6	1	145	1	109	1	107	11
6	2	148	4	112	4	104	8
6	3	153	9	117	9	99	3
6	4	160	4	124	4	92	8
6	5	169	1	133	1	83	11
6	6	180	0	144	0	72	0

Tabloda 40 sayısında büyük değerler işaretlenmemiştir.

Renklendirilen değerler 2 ile 40 arasındaki asal sayılardır ve bu değerlerin oluşturduğu küme aşağıdaki şekildedir:

{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37}

Kırmızı ile işaretlenen 25 sayısı, kurallara uymasına karşın asal sayılara dahil edilmemiştir. Çünkü son adımda yaptığımız işlemle (asal sayıların kareleri ve karelerinin katlarının "asal değil" olarak işaretlenmesi) 25 sayısının asal olmadığı belirlenmiştir. ($25 = 5^2$)

NOT: Tablodaki x ve y değerleri, 40 sayısının kareköküne kadar ilerletilmiştir, çünkü daha büyük değerler istediğimiz aralığın dışındadır. Bu sebeple üst limit değerinin karekökünden büyük x ve y değerlerinin incelenmesine gerek yoktur.

Dikkat edilmesi gereken başka bir durumu incelemek için 70 sayısına kadarki asal sayıları bulmak isteyelim. x ve y değerleri incelenirken:

- $x=2$ $y=7$ durumuna gelindiğinde $4x^2+y^2 = n$ denklemi için $n=65$ bulunur. Bu değer mod 12'de 5'tir ve ilk kümeye girer. Dolayısıyla 65 sayısı için işaret değişikliği yapılır.
  ("asal değil" olan işaret "asal" olarak değiştirilir.)
- $x=4$ $y=1$ durumuna gelindiğinde $4x^2+y^2 = n$ denklemi için n değeri yine 65 bulunur. Bu değer mod 12'de 5'tir ve ilk kümeye girer. Dolayısıyla 65 sayısı için tekrar işaret değişikliği yapılır.
  ("asal" olan işaret "asal değil" olarak değiştirilir.)

$4x^2+y^2 = 65$ için çift sayıda (x,y) ikilisi tespit edilmiştir. Asal olma şartında tek sayıda (x,y) ikilisi olması gerektiğinden 65 sayısının bir asal sayı olmadığı belirlenmiştir.

Karmaşıklık Hesabı

Yaptığımız işlem karmaşık gözükse de algoritmanın temelinde her sayı için gerekli işlemleri bir kere uygulamamız yeterlidir. Kısacası işlem sayısı, hesaba sokulacak sayı kadardır. Bu sebeple karmaşıklığımız Big O notasyonuna göre O(N)'dir.

Uygulama Alanları

Atkin Kalburu, asal sayıları bulmak için tasarlandığından, asal sayıları kullanan her işlem uygulama alanı dahilindedir. Asal sayıların pek çok kullanım alanı vardır.

- Kriptografi alanında çoğu şifreleme yönteminde asal sayılar kullanılır.
  (Örnek: RSA Şifreleme algoritmasında)
- Elektronik haberleşme hizmeti içinde güvenli ses/veri haberleşmesinde şifreleme yöntemlerinden yararlanıldığından asal sayılar kullanılır.
- Bankacılıkta ve banka sistemlerinde taksitlendirmenin yapılamaması, paranın bölünememesi gibi durumlarla karşılaşılmaması için asal sayılardan yararlanılır.
- Matematiğin en büyük inceleme alanlarından biri olduğu için ve Matematik alanında birçok hesabın yapılmasında kullanıldığı için asal sayılar fazlasıyla kullanılır.

Algoritmanın Kodu

```c
#include <stdio.h>
#include <stdbool.h>
#include <time.h>
#define SIZE 30000

void sieveOfAtkin(int limit, bool sieve[SIZE]){
	//n: denenecek değer
	int x, y, n;

	//tüm sayıları başta asal değil işaretlenmesi
	for(n=2 ; n <= limit ; n++){
		sieve[n] = false;
	}
	//limit=2 durumu
	if(limit >= 2){
		sieve[2] = true;
	}
	//limit=3 durumu
	if(limit >= 3){
		sieve[3] = true;
	}
	//x=1, y=1'den başlanarak şartların denenmesi
	for(x=1 ; x*x < limit ; x++){
		for(y=1 ; y*y < limit ; y++){
			//4*x*x + y*y mod 12 kuralı
			n = 4*x*x + y*y;
			if(n <= limit && (n % 12 == 1 || n % 12 == 5)){
				sieve[n] ^= true;
			}
			//3*x*x + y*y mod 12 kuralı
			n = 3*x*x + y*y;
			if(n <= limit && n % 12 == 7){
				sieve[n] ^= true;
			}
			//3*x*x - y*y mod 12 kuralı
			n = 3*x*x - y*y;
			if(x>y && n <= limit && n % 12 == 11){
				sieve[n] ^= true;
			}
		}
	}
	//asal bulunan sayıların karesinin katlarının çıkarılması
	for (n = 5 ; n*n < limit ; n++){
		if (sieve[n] == true){
			for (x = n*n ; x < limit ; x = x + n*n){
				sieve[x] = false;
			}
		}
	}
}

void printPrimes(int limit, bool sieve[SIZE]){
	int x = 0, n;
	printf("\nAsal Sayilar:\n");
	for (n = 2; n < limit; n++){
		//her satıra 8 adet asal sayı yazdırılması
		if(sieve[n] == true){
			x = x + 1;
			printf("%d\t", n);
			if(x == 8){
				printf("\n");
				x = 0;
			}
		}
	}
}
```

```c
int main(){
	//limit: bulunmasını istediğimiz aralığın üst sınırı
	//sieve[]: değerlerin asal veya asal değil olarak işaretleneceği dizi
	//sieve[n] = true (asal)
	//sieve[n] = false (asal değil)
	int limit;
	bool sieve[SIZE];
	double duration; //geçen zaman
	struct timespec start, end; //başlangıç ve bitiş zamanı

	//pozitif ve tamsayı bir limit değeri girilmesi
	printf("-----");
	printf("\n\nAtkin Kalburu (Sieve of Atkin)");
	printf("\n\n2 ile istediginiz araliktaki tum asal sayilari bulun.");
	printf("\n\n-----");
	printf("\n\nAralik Ust Limiti: ");
	scanf("%d", &limit);
	while(limit <= 1){
		printf("(!) Limit değeri 1'den buyuk pozitif bir tamsayi olmalidir.");
		printf("\nLutfen uygun bir limit degeri giriniz: ");
		scanf("%d", &limit);
	}

	//başlangıç zamanının belirlenmesi
	clock_gettime(CLOCK_MONOTONIC, &start);

	//asal sayıların bulunması
	sieveOfAtkin(limit, sieve);

	//bitiş zamanının belirlenmesi
	clock_gettime(CLOCK_MONOTONIC, &end);

	//asal sayıların yazdırılması
	printPrimes(limit, sieve);

	//geçen sürenin hesaplanması
	duration = (end.tv_sec - start.tv_sec) * 1e9;
	duration = (duration + (end.tv_nsec - start.tv_nsec)) * 1e-9;

	//hesaplama süresinin yazdırılması
	printf("\n\nHesaplama Suresi: %f sn", duration);
}
```

Çıktı

```
E:\Dev-C++\Odevler\Atkin Kalburu\atkin_single\atkinkalburu-aliemrenebiler-single.exe
-----

Atkin Kalburu (Sieve of Atkin)

2 ile istediginiz araliktaki tum asal sayilari bulun.

-----

Aralik Ust Limiti: 40

Asal Sayilar:
2       3       5       7       11      13      17      19
23      29      31      37

Hesaplama Suresi: 0.000002 sn
--------------------------------
Process exited after 2.597 seconds with return value 31
Press any key to continue . . .
```

Zaman Analizi

İşlemi farklı değerlerde birden fazla kez yaptırabilmek ve bar diyagramını yazdırabilmek için gerekli kod aşağıdaki gibidir.

```
int main(){
	//limit[]: bulunmasını istediğimiz aralığın üst sınırı
	//sieve[]: değerlerin asal veya asal değil olarak işaretleneceği dizi
	//sieve[n] = true (asal)
	//sieve[n] = false (asal değil)
	int limit[8], i, j, max;
	bool sieve[SIZE];
	double duration[8]; //geçen zamanlar
	struct timespec start, end; //başlangıç ve bitiş zamanları

	//pozitif ve tamsayı bir limit değeri girilmesi
	printf("-----");
	printf("\n\nAtkin Kalburu (Sieve of Atkin)");
	printf("\n\n2 ile istediginiz araliktaki tum asal sayilari bulun.");
	printf("\n\n-----");
	printf("\n\nAraliklarin Ust Limitleri\n");
	for(i=0 ; i<8 ; i++){
		printf("Limit %d: ", i+1);
		scanf("%d", &limit[i]);
		while(limit[i] <= 1){
			printf("(!) Limit değeri 1'den buyuk pozitif bir tamsayi olmalidir.");
			printf("\nLutfen uygun bir limit degeri giriniz: ");
			scanf("%d", &limit[i]);
		}
	}
	for(i=0 ; i<8 ; i++){
		//başlangıç zamanının belirlenmesi
		clock_gettime(CLOCK_MONOTONIC, &start);

		//asal sayıların bulunması
		sieveOfAtkin(i, limit, sieve);

		//bitiş zamanının belirlenmesi
		clock_gettime(CLOCK_MONOTONIC, &end);

		//asal sayıların yazdırılması
		printPrimes(i, limit, sieve);

		//geçen sürenin hesaplanması
		duration[i] = (end.tv_sec - start.tv_sec) * 1e9;
		duration[i] = (duration[i] + (end.tv_nsec - start.tv_nsec)) * 1e-9;
	}
	//hesaplama sürelerinin yazdırılması
	printf("\n\n------------------");
	printf("\nHesaplama Sureleri");
	printf("\n\nUst Limit:\tSure:");
	printf("\n----------\t--------");
	for(i=0 ; i<8 ; i++){
		printf("\n%10d\t%f sn", limit[i], duration[i]);
	}
	//bar diyagramının yazdırılması
	printf("\n\nBar Diyagrami");
	printf("\n---------------------------------------------------------> Sure");
	for(i=0 ; i<8 ; i++){
		printf("\n       | \n %5d | ", limit[i]);
		max = duration[i] * 200000;
		for(j=0 ; j<max ; j++){
			printf("o");
		}
	}
	printf("\n       | \n       | \n       V");
	printf("\n       Girdiler");
}
```

Zaman analizini 40, 200, 1000, 2000, 5000, 10000, 15000, 20000 değerleri için yapacağız.

Bu değerler için elde edilen hesaplama süreleri ve sürelerin bar diyagramı şeklinde gösterimi şu şekildedir:

```
E:\Dev-C++\Odevler\Atkin Kalburu\atkin_multiple\atkinkalburu-aliemrenebiler-multiple.exe
19889   19891   19913   19919   19927   19937   19949   19961
19963   19973   19979   19991   19993   19997

------------------
Hesaplama Sureleri

Ust Limit:      Sure:
----------      --------
        40      0.000002 sn
       200      0.000005 sn
      1000      0.000019 sn
      2000      0.000036 sn
      5000      0.000081 sn
     10000      0.000154 sn
     15000      0.000202 sn
     20000      0.000298 sn

Bar Diyagrami
--------------------------------------------------------> Sure
       |
    40 |
       |
   200 | o
       |
  1000 | ooo
       |
  2000 | ooooooo
       |
  5000 | oooooooooooooooo
       |
 10000 | oooooooooooooooooooooooooooooo
       |
 15000 | oooooooooooooooooooooooooooooooooooooooo
       |
 20000 | ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
       |
       |
       V
       Girdiler
--------------------------------
Process exited after 32.66 seconds with return value 16
Press any key to continue . . .
```

Algoritmanın Rakipleri

Aşağıdaki yöntemler, Atkin Kalburu'na alternatif asal sayı bulma veya bir sayının asallığını test etme yöntemleridir.

- AKS asallık testi
- Baillie-PSW asallık testi
- Fermat asallık testi
- Lucas asallık testi
- Miller–Rabin asallık testi
- Eratosthenes Kalburu
- Sundaram Kalburu

Bu noktada benzerlikleri ve hızlarının yakın olması sebebiyle Erastosthenes Kalburu, Atkin Kalburu'nun en büyük rakibidir.

Bu yöntemleri Atkin Kalburu ile kıyaslayacağız.

Kısıtlar, Avantajlar ve Dezavantajlar

Atkin algoritmasının en büyük kısıtlarından biri bellektir. Çünkü sonsuza giden asal sayıları bulmak için Kalbur yöntemleri fazlasıyla elverişli ve hızlı olmalarına karşın, ancak günümüz teknolojisinin ulaşabileceği maksimum bellek kapasitesine kadarki asal sayıları bulabilirler.

AKS, Baillie-PSW, Fermat, Lucas, Miller–Rabin asallık testleri bir sayının asal olduğunu veya olmadığını belirlemek için tasarlanmışlardır. Sadece bir sayının asallığı araştırılırken Atkin Kalburu, bu yöntemlere göre dezavantajlıdır.

Fakat belli bir aralıktaki tüm asal sayılar istendiğinde Kalbur (Sieve) yöntemleri çok daha hızlı çalışacağından daha avantajlıdır.

Özellikle AKS gibi asallık testleri anlaşılması ve kodlanması zor olduğundan, Kalbur yöntemleri daha avantajlıdır.

Kalbur yöntemlerini birbirleriyle kıyaslayalım.

- Sundaram; Atkin ve Eratosthenes'e göre çok yavaştır, bu sebeple Atkin Kalburu daha avantajlıdır. Ayrıca, Sundaram Kalburu yavaş olmasından dolayı günümüzde kullanılmamaktadır.
- Eratosthenes, temel Atkin algoritmasına göre daha hızlıdır, fakat aradaki fark çok düşüktür. Yine de Atkin Kalburu, hız konusunda Eratosthenes Kalburu'na göre dezavantajlıdır denebilir.
- Atkin Kalburu daha yavaş olsa da, çeşitli implementasyonlarla Eratosthenes Kalburu'ndan daha hızlı hale getirilebilmektedir. Bu implementasyonlar özellikle işlemin ilerleyen kısımlarda daha büyük farklar yaratmaktadır. Yani bulunabilecek en büyük asal sayıyı bulma yolunda Atkin Kalburu daha avantajlıdır.
- Eratosthenes, temel Atkin algoritmasına göre anlaşılması daha basittir. Bu sebeple Atkin Kalburu'na göre daha fazla tercih edilmektedir.
- Atkin Kalburu'nun Big O notasyonuna göre karmaşıklığı O(N) iken, Eratosthenes Kalburu'nun karmaşıklığı O(N*log(logN))'dir.

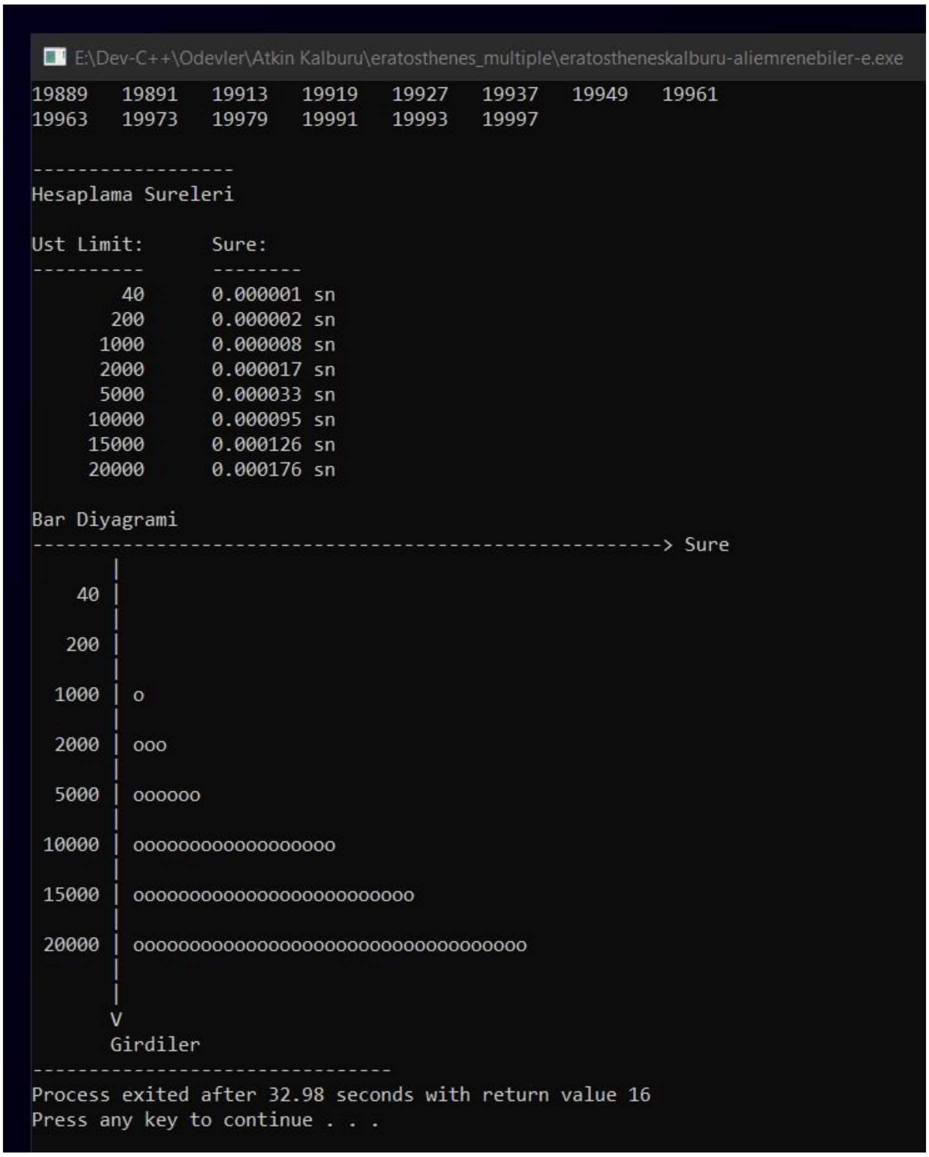
Daha iyi görmek için Eratosthenes ve Atkin verilerini karşılaştıralım.

Eratosthenes Kalburu algoritmasının kodu:

```
void sieveOfEratosthenes(int limit, bool sieve[SIZE]){
	int n, m;

	//tüm sayıları başta asal işaretlenmesi
	for(n=2 ; n <= limit ; n++){
		sieve[n] = true;
	}

	//asal sayıların katlarının asal değil işaretlenmesi
	for(n=2 ; n*n < limit ; n++){
		if (sieve[n] == true){
			for(m=n*n ; m < limit ; m = m + n){
				sieve[m] = false;
			}
		}
	}
}
```

Zaman analizini yine 40, 200, 1000, 2000, 5000, 10000, 15000, 20000 değerleri için yapacağız.

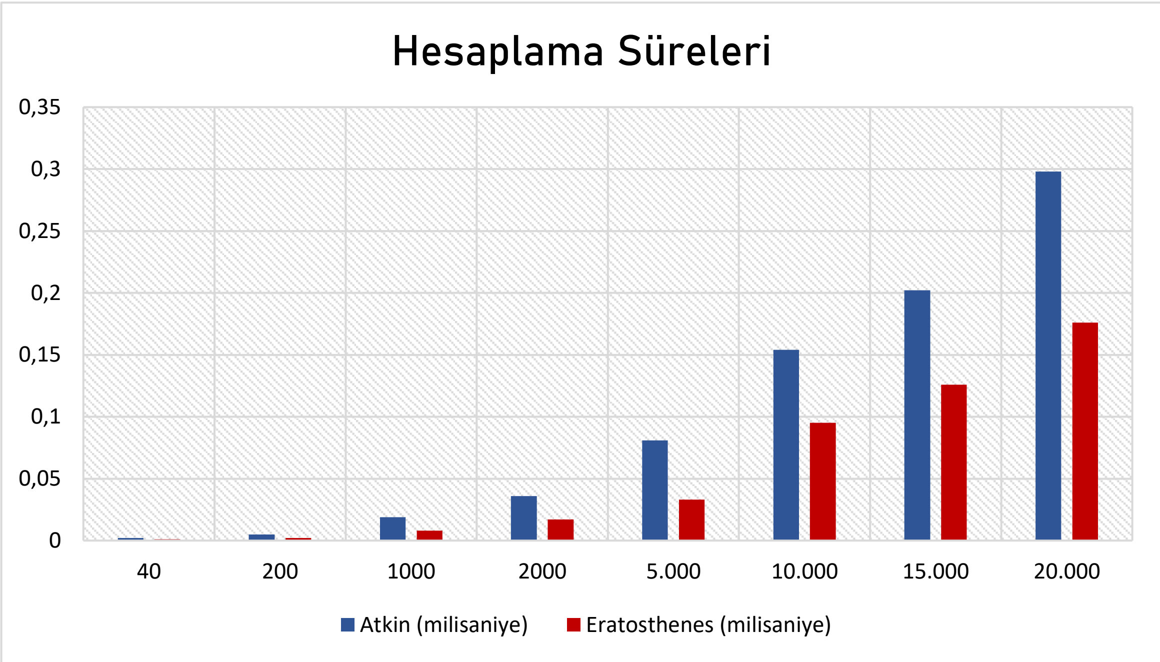
Bu değerler için Eratosthenes Kalburu'nun hesaplama süreleri aşağıdadır.



```
E:\Dev-C++\Odevler\Atkin Kalburu\eratosthenes_multiple\eratostheneskalburu-aliemrenebiler-e.exe
19889    19891    19913    19919    19927    19937    19949    19961
19963    19973    19979    19991    19993    19997

------------------
Hesaplama Sureleri

Ust Limit:      Sure:
----------      --------
        40      0.000001 sn
       200      0.000002 sn
      1000      0.000008 sn
      2000      0.000017 sn
      5000      0.000033 sn
     10000      0.000095 sn
     15000      0.000126 sn
     20000      0.000176 sn

Bar Diyagrami
---------------------------------------------------------> Sure
       |
    40 |
       |
   200 |
       |
  1000 | o
       |
  2000 | ooo
       |
  5000 | oooooo
       |
 10000 | oooooooooooooooooo
       |
 15000 | ooooooooooooooooooooooooo
       |
 20000 | ooooooooooooooooooooooooooooooooooo
       |
       |
       V
       Girdiler
--------------------------------

Process exited after 32.98 seconds with return value 16
Press any key to continue . . .
```

(Atkin Kalburu'nun hesaplama sürelerini önceden hesaplamıştık.)

Atkin Kalburu ve Eratosthenes Kalburu hesaplama sürelerini bir arada inceleyelim.



Verilerden görüyoruz ki 20.000'e kadarki asal sayıları; Eratosthenes, Atkin Kalburu'ndan 0,122 milisaniye daha önce hesaplamıştır.

Kaynaklar

https://en.wikipedia.org/wiki/Sieve_of_Atkin - Wikipedia | Sieve of Atkin

https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes - Wikipedia | Sieve of Eratosthenes

https://en.wikipedia.org/wiki/Prime_number#Sieves - Wikipedia | Prime numbers - Sieves

https://en.wikipedia.org/wiki/Primality_test - Wikipedia | Primality Test

http://bilgisayarkavramlari.sadievrenseker.com/2011/04/13/atkin-kalburu-sieve-of-atkin/ - Bilgisayar Kavramları | Atkin Kalburu (Sieve of Atkin)

https://www.geeksforgeeks.org/sieve-of-atkin/ - GeeksforGeeks | Sieve of Atkin

https://www.geeksforgeeks.org/measure-execution-time-with-high-precision-in-c-c/ - GeeksforGeeks | Measure Execution Time

https://iq.opengenus.org/sieve-of-atkin/ - OpenGenus.org | Sieve of Atkin

https://cp-algorithms.com/algebra/prime-sieve-linear.html - CP-Algorithms | Sieve of Eratosthenes Having Linear Time Complexity

https://www.matematikciler.com/asal-sayilar-ve-sifreleme-kriptoloji/ - Matematikciler.com | Asal Sayılar ve Şifreleme

https://afyonluoglu.org/PublicWebFiles/Reports-TR/Uzmanlik_Tez/BTK/siber/2013%20Haziran%20Kriptolu%20Haberle%c5%9fme%20%c4%b0ncelemesi.PDF - Mustafa TEFON | Elektronik Haberleşme Hizmeti İçinde Güvenli Ses/Veri Haberleşmesi Açısından Kriptolu Haberleşmenin İncelenmesi, Düzenlemeler, Öneriler ve Türkiye Analizi

https://services.tubitak.gov.tr/edergi/user/yaziForm1.pdf?cilt=45&sayi=782&sayfa=68&yaziid=33620 - Oğulcan AÇIKGÖZ, Aslı ŞENSOY | Asal Sayıların Hikayesi

https://books.google.com.tr/books?id=Y_JpCQAAQBAJ&pg=PA206&lpg=PA206&dq=prime+sieve+test+report&source=bl&ots=z3amtJKv9m&sig=ACfU3U1h8o_vrkvTfM1OsFwQJrjrEn_PYQ&hl=tr&sa=X&ved=2ahUKEwj_g6vv9vnpAhVSzRoKHQZwAugQ6AEwAHoECAkQAQ#v=onepage&q&f=false - Florian HESS, Sebastian PAULI, Michael POHST | Algorithmic Number Theory: 7th International Symposium, ANTS-VII

https://www.researchgate.net/publication/323788948_Prime_Numbers_Comparison_using_Sieve_of_Eratosthenes_and_Sieve_of_Sundaram_Algorithm - D Abdullah, R Rahim, D Apdilah, S Efendi, T Tulus and S Suwilo | Prime Numbers Comparison using Sieve of Eratosthenes and Sieve of Sundaram Algorithm

https://stackoverflow.com/questions/5235865/comparison-of-sieve-of-sundaram-and-sieve-of-atkin-for-generating-a-list-of-prim - StackOverflow.com | Comparison of Sieve of Sundaram and Sieve of Atkin for generating a list of prime numbers

https://www.quora.com/Is-the-AKS-Primality-Test-better-than-the-Sieve-of-Eratosthenes-If-yes-why-is-it-not-used-by-coders-in-competitive-programing - Quora | Is the AKS Primality Test better than the Sieve of Eratosthenes?