Sieve of Eratosthenes, Algoritma Bilangan Prima

M. R. Al-ghazali – NIM. 13509068
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
algo_rhythm@rocketmail.com

***Abstrak*—Makalah ini membahas tentang bilangan prima, khususnya cara penentuan bilangan prima, baik satu maupun beberapa bilangan (deret) dengan cara yang mangkus. Untuk bilangan yang kecil, dalam menentukan deret bilangan prima memang dapat dicek satu per satu hingga mendapatkan semua bilangan yang merupakan bilangan prima. Namun untuk bilangan yang besar hal ini sangat tidak mangkus. Masalah ini dapat diselesaikan dengan suatu algoritma yang sangat terkenal, yaitu *Sieve of Eratosthenes*. Algoritma ini ditemukan oleh seorang matematikawan, ahli geografi dan astronom bernama Eratosthenes. Metode ini menyaring bilangan-bilangan bukan prima dari sekumpulan bilangan tertentu sehingga dihasilkan kumpulan bilangan yang merupakan bilangan prima. Cara ini sangat mangkus dibandingkan dengan cara yang sederhana terutama untuk bilangan yang besar, dengan kompleksitas O((n log n)(log log n)).**

***Kata Kunci*—algoritma, bilangan, Brute Force, kompleksitas, mangkus, prima, sederhana, *sieve of Eratosthenes*.**

I. PENDAHULUAN

Bilangan prima merupakan bilangan yang sangat unik bila dibandingkan dengan jenis bilangan lainnya. Penyebaran bilangan ini tidak memiliki pola, sehingga menentukan suatu deret bilangan prima tidaklah mudah. Kita dapat mengecek bilangan satu per satu untuk menentukan apakah bilangan tersebut merupakan bilangan prima. Tetapi apabila misalnya kita ingin mencari bilangan prima ke-10.000, sangat tidak mangkus dan sangkil jika kita mengecek bilangan satu per satu hingga menemukan bilangan prima yang ke-10.000. Oleh karena itu, dibutuhkan suatu metode atau algoritma yang mangkus dan sangkil untuk menentukan bilangan prima, terutama dalam cakupan bilangan yang besar.

II. BILANGAN PRIMA DAN ERATOSTHENES

A. Bilangan Prima

Bilangan prima adalah suatu bilangan bulat positif yang hanya habis dibagi bilangan 1 (satu) dan bilangan itu sendiri, dimulai dari 2. Misalnya angka 7, karena 7 habis dibagi 1 dan 7 tetapi tidak habis dibagi bilangan bulat positif yang lain, dapat disimpulkan bahwa 7 merupakan bilangan prima. Semua bilangan prima merupakan bilangan ganjil kecuali angka 2. 2 adalah satu-satunya bilangan prima yang genap, karena bilangan genap selain 2 pasti habis dibagi 2 yang menyebabkan bilangan tersebut tidak memenuhi syarat atau definisi bilangan prima.

Semua bilangan bulat lebih besar dari satu yang bukan merupakan bilangan prima disebut bilangan komposit. Dapat dikatakan, bilangan komposit merupakan bilangan yang mempunyai lebih dari 2 faktor bilangan bulat positif. Misalnya angka 9, karean 9 habis dibagi 1, 3, dan 9, dapat disimpulkan bahwa 9 bukan merupakan bilangan prima dan otomatis merupakan bilangan komposit. Pengertian lain, bilangan komposit merupakan bilangan yang minimal mempunyai 2 (dua) buah faktor bilangan prima. Misalkan angka 9 tadi memiliki 2 faktor bilangan prima yaitu 3 dan 9.

Bilangan prima dilihat dari bentuknya ada bermacam-macam, diantaranya bilangan prima *twin* (pasangan bilangan prima *p* dan *p + 2*), bilangan prima Mersenne (bilangan prima dengan bentuk $2^p - 1$), bilangan prima *primorial* (bilangan prima dengan bentuk *n# +/- 1*), bilangan prima *factorial* (bilangan prima dengan bentuk n! +/- 1), dan bilangan prima Sophie Germain (bilangan prima *p* dimana *2p + 1* juga merupakan bilangan prima).

Matematikawan masa lalu beranggapan bahwa bentuk $2^p - 1$ (bentuk Mersenne) akan menghasilkan semua bilangan prima untuk p bilangan prima, jika p bukan prima maka bilangan yang didapat adalah komposit.. Rumus berbentuk $2^p - 1$ ini tampaknya diadopsi dari rumus bilangan sempurna $2(n - 1)(2n - 1)$ yang mempersyaratkan $2n - 1$ adalah prima. Sementara $2n - 1$ tidak prima jika n bukan prima. Tetapi, pada tahun 1536 Hudalricus Regius menunjukkan bahwa $2^{11} - 1 = 2047$ bukan prima karena $2047 = 23 \times 89$. Selain itu Pietro Cataldi dari Italia pada tahun 1603 melakukan verifikasi terhadap $2^{17} - 1$ dan $2^{19} - 1$, ternyata keduanya adalah prima dan menurut Cataldi pula $2^p - 1$ adalah prima juga untuk p = 23, 29, 31 dan 37. Pada tahun 1640, Pierre de Fermat berhasil menunjukkan bahwa Cataldi keliru untuk p = 29 dan beberapa waktu kemudian Euler menunjukkan bahwa Cataldi kali ini benar untuk p = 31.

Bilangan prima secara matematis jumlahnya tidak terhingga. Hal ini telah dibuktikan oleh Euclid (300 SM) dalam buku IX Elements. Jadi, sebanyak apapun kita menghitung, kita pasti akan menemukan bilangan prima

walaupun makin besar bilangan prima maka jarak suksesornya akan makin besar juga, sehingga bilangan prima menjadi semakin jarang pada bilangan besar. Bilangan prima terbesar yang ditemukan hampir selalu merupakan bilangan prima Mersenne. Bilangan prima terakhir yang ditemukan sampai sekarang (2010 M) adalah bilangan Mersenne ke-47, yaitu $2^{42.643.801} - 1$, bilangan dengan 12.837.064 digit yang ditemukan oleh Odd Magnar Strindmo yang berasal dari Melhus, Norwegia, pada tahun 2009. Namun bilangan prima terbesar yang pernah ditemukan sampai saat ini adalah bilangan Mersenne ke-45, yaitu $2^{43.112.609} - 1$, bilangan dengan 12.978.189 digit. Bilangan ini ditemukan di Universitas California Los Angeles (UCLA) pada tahun 2008.

B. Eratosthenes

Eratosthenes adalah seorang matematikawan, ahli geografi dan seorang astronom yang berasal dari Cyrene (saat ini Libya) di Afrika Utara. Eratosthenes hidup sekitar tahun 276 - 194 SM. Namanya berasal dari bahasa Yunani, Ἐρατοσθένης.

Eratosthenes belajar di Alexandria dan untuk beberapa tahun di Athena. Dia adalah orang kedua setelah Zenodotos yang menjabat sebagai kepala perpustakaan di Universitas Alexandria, Mesir, yang dibangun oleh Alexander Agung.

Gambar 1.1. Eratosthenes (276 – 194 SM)

Eratosthenes yang berteman dengan Archimedes merupakan salah seorang pelajar yang pandai pada masa itu. Dia banyak menulis tentang sains dan filsafat. Sebagai seorang ahli geografi, dia menulis buku geografi pertama yang memberikan basis matematika pada geografi dan memperlakukan bumi sebagai sebuah globe yang terbagi menjadi zona-zona Frigid, Temperate, dan Torrid. Sebagai seorang pustakawan juga Eratosthenes mengetahui dari banyak buku-buku bahwa matahari berada pada titik tertingginya di langit pada siang hari tanggal 22 Juni, pada musim panas. Dia juga melakukan pengukuran sudut kemiringan matahari juga jarak keliling bumi yang tingkat ketelitiannya tinggi. Dan sebagai seorang matematikawan, Eratosthenes menemukan sebuah metode untuk menemukan bilangan-bilangan prima, yaitu *Sieve of Eratosthenes*.

III. ALGORITMA SEDERHANA UNTUK MENENTUKAN BILANGAN PRIMA

Telah diketahui bahwa bilangan prima X hanya mempunyai 2 buah faktor, yaitu angka 1 (satu) dan bilangan X (bilangan itu sendiri). Jadi, untuk menentukan suatu bilangan X adalah bilangan prima atau bukan, maka kita harus mencoba membagi bilangan X dengan semua bilangan bulat positif dari 1 sampai X, bila bilangan X hanya habis dibagi 1 dan X maka bilangan tersebut merupakan bilangan prima, selain itu bilangan komposit.

Akan tetapi, tentunya semua bilangan bulat positif X > 1 akan habis dibagi oleh 1 dan X. Oleh karena itu, dalam menentukan suatu bilangan X merupakan bilangan prima atau bukan, kita hanya perlu mengecek pembagian X dengan 2 sampai dengan X – 1. Jika terdapat satu bilangan yang habis membagi X di antara 2 sampai X – 1, maka bilangan tersebut bukan merupakan bilangan prima. Sebagai contoh untuk bilangan 8:

- 8 dibagi 2 = 4, sisa 0 (habis)
- 8 dibagi 3 = 2, sisa 2
- 8 dibagi 4 = 2, sisa 0 (habis)
- 8 dibagi 5 = 1, sisa 3
- 8 dibagi 6 = 1, sisa 2
- 8 dibagi 7 = 1, sisa 1

di antara 2 sampai 8 – 1 = 7, terdapat bilangan yang habis membagi 8 yaitu 2 dan 4. Berarti, 8 bukan bilangan prima. Contoh lain, untuk bilangan 11:

- 11 dibagi 2 = 5, sisa 1
- 11 dibagi 3 = 3, sisa 2
- 11 dibagi 4 = 2, sisa 3
- 11 dibagi 5 = 2, sisa 1
- 11 dibagi 6 = 1, sisa 5
- 11 dibagi 7 = 1, sisa 4
- 11 dibagi 8 = 1, sisa 3
- 11 dibagi 9 = 1, sisa 2
- 11 dibagi 10 = 1, sisa 1

di antara 2 sampai 11 – 1 = 10, tidak ada bilangan yang habis membagi 11. Berarti dapat disimpulkan bahwa 11 merupakan bilangan prima.

Jadi, untuk menentukan apakah suatu bilangan X merupakan bilangan prima, dapat digunakan prosedur dalam notasi algoritmik sebagai berikut:

```
procedure Prima(input X : integer, output
                IsPrima : boolean)
{
 Menentukan apakah X merupakan bilangan
 prima
 I. S. sembarang
 F. S. IsPrima bernilai true jika X
 bilangan prima, false jika tidak
}
Deklarasi
       i : integer
Algoritma
       IsPrima ← true
       if (X ≠ 2) then
              for i ← 2 to X – 1 do
                     if (X mod i = 0) then
                            IsPrima ←false
                            exit for
                     end if
              end for
       end if
```

Algoritma di atas hanya menentukan apakah suatu bilangan tunggal merupakan bilangan prima atau bukan. Bila kita ingin mencari suatu deret atau kumpulan bilangan prima yang kurang dari atau sama dengan N, maka kita dapat menggunakan prosedur di atas seperti pada prosedur berikut:

```
procedure DeretPrima(input N : integer)
{
 Mencetak semua bilangan yang kurang dari
 atau sama dengan N yang merupakan
 bilangan prima
 I. S. sembarang
 F. S. semua bilangan prima ≤ N dicetak
}
Deklarasi
       i : integer
       IsPrima : boolean
       procedure Prima(input X : integer,
                       output IsPrima :
                       boolean)
Algoritma
       for i ← 2 to N do
              Prima(i, IsPrima)
              if IsPrima then
                     output(i)
              end if
       end for
```

Pada algoritma di atas, bila kita memasukkan N = 15, maka akan dihasilkan bilangan-bilangan 2, 3, 5, 7, 11, dan 13. Program di atas mengecek mulai dari bilangan 2, 3, 4, dan seterusnya hingga 15 apakah bilangan tersebut merupakan bilangan prima. Dalam setiap pengecekan (misal bilangan X), program mengecek hasil pembagian bilangan dengan 2 sampai X – 1 satu per satu apakah habis dibagi. Algoritma ini tentunya tidak mangkus karena terlalu banyak melakukan pengecekan, terlebih lagi untuk bilangan yang besar. Algoritma ini bisa disebut dengan Algoritma *Brute Force*, yaitu algoritma yang menggunakan pemikiran sederhana dan jelas sehingga disebut juga algoritma *naive*. Dalam beberapa literatur algoritma ini lebih dikembangkan dibanding algoritma sebelumnya. Hal ini berkait dengan teorema bahwa "Jika bilangan n komposit, maka n mempunyai pembagi bilangan prima yang kurang dari atau sama dengan $\sqrt{n}$". Jadi algoritma sebelumnya diubah berupa pengecekan bilangan hanya sampai $\sqrt{X}$ (pembulatan) saja. Dengan pengembangan ini, algoritma ini dapat dituliskan menjadi:

```
procedure Prima(input X : integer, output
                IsPrima : boolean)
{
 Menentukan apakah X merupakan bilangan
 prima
 I. S. sembarang
 F. S. IsPrima bernilai true jika X
 bilangan prima, false jika tidak
}
Deklarasi
       i : integer
Algoritma
       IsPrima ← true
       if (X ≠ 2) then
              for i ← 2 to ⌈√X⌉ do
                     if (X mod i = 0) then
                            IsPrima ←false
                            exit for
                     end if
              end for
       end if
```

Sebenarnya hanya perlu dilakukan pembulatan ke bawah untuk membulatkan $\sqrt{X}$. Namun untuk menghindari perulangan yang tidak valid (yaitu 2 ke 1) maka pembulatan dilakukan ke atas.

Algoritma ini tentunya lebih baik dari algoritma sebelumnya, terutama untuk bilangan yang besar. Namun, algoritma ini tetap tidak mangkus jika digunakan untuk menentukan deret bilangan prima yang sangat besar.

IV. SIEVE OF ERATOSTHENES

Misalnya akan ditentukan suatu bilangan prima ke-10.000. Kita dapat menggunakan metode *Brute Force* untuk menentukannya, karena cara ini cukup baik untuk mencari satu bilangan prima. Kemudian akan ditentukan 10.000 bilangan prima pertama. Hal ini sangat tidak mangkus jika menggunakan metode *Brute Force*, karena pengecekan yang terlalu banyak. Oleh karena itu, akan dicoba suatu metode bernama *Sieve of Eratosthenes*. Algoritma yang ditemukan oleh seorang matematikawan

bernama Eratosthenes ini, sesuai dengan namanya, melakukan 'penyaringam' terhadap suatu kumpulan bilangan menjadi kumpulan bilangan prima dengan meng-eliminasi bilangan yang bukan bilangan prima. Lebih jelasnya, metode *Sieve of Eratosthenes* digambarkan pada langkah-langkah berikut:

1. Terdapat sebuah larik bilangan dari 2 sampai N
2. Bilangan terkecil yang tidak dicoret adalah bilangan prima
3. Coret bilangan ini dan semua kelipatan bilangan ini dalam larik
4. Ulangi langkah 2 dan 3 sampai semua bilangan dalam larik telah dicoret

Kita dapat menerapkan teorema tentang bilangan komposit yang telah disebutkan sebelumnya ke dalam mentode ini untuk membuatnya lebih mangkus, sehingga langkah-langkahnya menjadi:

1. Terdapat sebuah larik bilangan dari 2 sampai N
2. Bilangan terkecil yang tidak dicoret adalah bilangan prima
3. Coret semua kelipatan bilangan ini dalam larik
4. Ulangi langkah 2 dan 3 sampai mencapai √N (pembulatan)
5. Semua bilangan yang belum dicoret adalah bilangan prima

Sebagai contoh, kita akan menentukan sekumpulan bilangan prima dari 2 sampai dengan 100, larik bilangan ini dapat kita lihat di bawah ini:

1. Terdapat sebuah larik bilangan dari 2 sampai 100

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

2. Bilangan terkecil yang tidak dicoret adalah bilangan prima, yaitu 2
3. Coret semua kelipatan bilangan 2 dalam larik

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

4. Ulangi langkah 2 dan 3 sampai mencapai √100 (pembulatan) = 10
   - 3 adalah bilangan prima, coret semua kelipatan 3 dalam larik

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

   - 5 adalah bilangan prima, coret semua kelipatan 5 dalam larik

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

   - 7 adalah bilangan prima, coret semua kelipatan 7 dalam larik

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

   - 11 sudah melebihi √100 = 10, perulangan dihentikan

5. Semua bilangan yang belum dicoret adalah bilangan prima

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Kesimpulannya, bilangan prima antara 1 sampai dengan 100 adalah 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, dan 97.

Bila diimplementasikan dalam notasi algoritmik, hasilnya dapat dilihat pada prosedur di bawah ini:

```
procedure SieveEratos(input N : integer)
{
 Mencetak semua bilangan yang kurang dari
 atau sama dengan N yang merupakan
 bilangan prima
 I. S. sembarang
 F. S. semua bilangan prima ≤ N dicetak
}
Deklarasi
       i, j : integer
       T : array [2..N] of boolean
Algoritma
       {inisialisasi larik}
       for i ← 2 to N do
              Ti ← true
       end for
       {menyaring bilangan prima}
       for i ← 2 to ⌈√N⌉ do
              if (Ti) then
                     j ← i * i
                     while j <= N do
                            Tj ← false
                            j ← j + i
                     end while
              end if
       end for
       {mencetak bilangan prima}
       for i ← 2 to N do
              if Ti then
                     output (i)
              end if
       end for
```

Cara ini tentu saja jauh lebih mangkus bila dibandingkan dengan metode *Brute Force* yang melakukan banyak pengecekan. Untuk bilangan yang besar, metode *Sieve of Eratosthenes* tetap melakukan pengecekan yang relatif sedikit.

V. KOMPLEKSITAS ALGORITMA SIEVE OF ERATOSTHENES

Kita telah mengetahui bahwa dalam menentukan sekumpulan bilangan prima, metode *Sieve of Eratosthenes* merupakan metode yang sangat mangkus bila dibandingkan dengan metode sederhana atau *Brute Force*. Hal ini akan lebih terbukti apabila kita mengecek kompleksitas masing-masing algoritma.

Pada algoritma sederhana untuk menentukan bilangan prima sampai n, terjadi pengecekan sebanyak n kali (yaitu pengecekan tiap-tiap elemen apakah merupakan bilangan prima), dan pada setiap pengecekan terjadi perulangan sebanyak (n – 1) kali untuk mencari faktor selain bilangan 1 dan bilangan itu sendiri. Ini menghasilkan kompleksitas $O(n^2)$. Sedangkan pada algoritma *Sieve of Eratosthenes*, pengecekan tidak dilakukan pada setiap n, dikarenakan 'penyaringan' dengan memanfaatkan bilangan yang telah dicek sebelumnya, yaitu dengan 'mencoret' terlebih dahulu bilangan kelipatannya sehingga tidak perlu dicek lagi. Ini menyebabkan adanya faktor logaritma pada kompleksitas *Sieve of Eratosthenes*. Pada akhirnya, algoritma ini mempunyai kompleksitas O((n log n)(log log n)). Bagaimana perbedaan kedua kompleksitas ini dapat dilihat pada tabel kelompok algoritma berdasarkan kompleksitas waktu asimptotiknya di bawah ini:

Kelompok Algoritma	Nama
O(1)	konstan
O(log n)	logaritmik
O(n)	lanjar
O(n log n)	n log n
$O(n^2)$	kuadratik
$O(n^3)$	kubik
$O(2^n)$	eksponensial
O(n!)	faktorial

Berikut pula perbandingan laju n^2 dan (n log n)(log log n):

n	n^2	(n log n)(log log n)
200000	4E+10	767971,095
300000	9E+10	1213542,241
400000	1,6E+11	1676915,608
500000	2,5E+11	2153630,424
600000	3,6E+11	2641038,911
700000	4,9E+11	3137381,273
800000	6,4E+11	3641400,886
900000	8,1E+11	4152154,497
1000000	1E+12	4668907,502
1100000	1,21E+12	5191071,328
1200000	1,44E+12	5718163,658
1300000	1,69E+12	6249781,929
1400000	1,96E+12	6785584,991
1500000	2,25E+12	7325279,999
1600000	2,56E+12	7868612,792

1700000	2,89E+12	8415360,68
1800000	3,24E+12	8965326,916
1900000	3,61E+12	9518336,398
2000000	4E+12	10074232,27

Dilihat dari kedua tabel di atas jelas bahwa O((n log n)(log log n)) jauh lebih mangkus daripada O(n^2). Telah terbukti, bahwa metode *Sieve of Eratosthenes* merupakan jawaban dari permasalahan di mana kita ingin menentukan sekumpulan bilangan prima yang besar.

VII. KESIMPULAN

Kesimpulan yang dapat diambil dari pembahasan pada makalah ini adalah bahwa kita dapat menentukan suatu bilangan prima melalui beberapa metode atau algoritma. Pada cara yang sederhana, kita dapat mengecek satu per satu suatu kumpulan bilangan apakah memenuhi syarat bilangan prima, untuk menentukan bilangan prima, akan tetapi cara ini sangat tidak mangkus untuk angka yang besar. Hal ini dapat diselesaikan dengan mengembangkan Algoritma *Brute Force*, yaitu dengan membatasi pengecekan pembagian bilangan hanya sampai akar dari bilangan tersebut. Namun, metode ini tetap tidak mangkus untuk mencari sekumpulan bilangan prima yang besar. Metode *Sieve of Eratosthenes* merupakan jawaban dari permasalahan ini. Metode ini menyaring bilangan-bilangan bukan prima dari sekumpulan bilangan tertentu sehingga dihasilkan kumpulan bilangan yang merupakan bilangan prima. Cara ini sangat mangkus dibandingkan dengan cara yang sederhana terutama untuk bilangan yang besar, dengan kompleksitas O((n log n)(log log n)).

DAFTAR REFERENSI

[1] Munir, Rinaldi. 2008. *Diktat Kuliah IF2091 Struktur Diskrit*. Bandung: Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. Hal. X-1 – X-28.

[2] Wikipedia. http://en.wikipedia.org/wiki/Eratosthenes. Tanggal akses: 13 Desember 2010.

[3] Wikipedia. http://en.wikipedia.org/wiki/Sieve_of_Eratosthenes. Tanggal akses: 13 Desember 2010

[4] Wikipedia. http://en.wikipedia.org/wiki/Prime_number. Tanggal akses: 13 Desember 2010

[5] Yahoo!Answers. http://id.answers.yahoo.com/question/index?qid=20100317190417AAaAaYQ. Tanggal akses: 15 Desember 2010

[6] The Largest Known Primes. http://primes.utm.edu/largest.html. Tanggal akses: 15 Desember 2010

[7] GIMPS Home. www.mersenne.org. Tanggal akses: 15 Desember 2010

[8] Sejarah Bilangan Prima. http://rudyks3-majalengka.blogspot.com/2009/01/sejarah-perkembangan-bilangan-prima-drs.html. Tanggal akses: 15 Desember 2010

[9] Eratosthenes dan Matematika Sederhana Menentukan Keliling Bumi. http://blog.unsri.ac.id/fitrianarahmawati/matematika/eratosthenes-dan-matematika-sederhana-menentukan-keliling-bumi/mrdetail/15247. Tanggal akses: 15 Desember 2010.

[10] Time complexity of Sieve of Eratosthenes algorithm. http://stackoverflow.com/questions/2582732/time-complexity-of-sieve-of-eratosthenes-algorithm. Tanggal akses: 15 Desember 2010.

[11] Cazelais, Gilles. Sieve of Eratosthenes. http://pages.pacificcoast.net/~cazelais/222/sieve.pdf. Tanggal akses: 16 Desember 2010.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 16 Desember 2010

M. R. Alghazali
NIM. 13509068