Josephus Problem

Mohammad Ridwan Hady Arifin - 13517007
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
ridwanhady@students.itb.ac.id

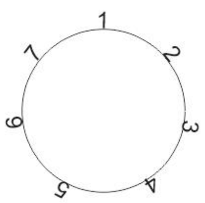
***Abstract*—Josephus problem adalah sebuah persoalan matematika pada abad pertama masehi. Permasalahan nya adalah terdapat 41 orang yang duduk melingkar, dimana orang yang mendapatkan 'undian' nomor 3 akan dipaksa untuk melakukan bunuh diri. Dikisahkan pada saat itu, tempat yang amana adalah nomor 16 dan 31. Permasalahan ini, tanpa disadari oleh orang – orang pada abad tersebut, adalah sebuah persoalan yang menyangkut rekursi. Selain menggunakan pendekatan rekursi, ternyata masalah josephus juga dapat diselesaikan dengan pendekatan lain, seperti pendekatan bilangan biner. Selain itu, untuk pemenang dari n yang sangat besar, diperlukan bantuan dari komputer, sehingga program untuk menghitung pemenang dari permasalahan josephus akan dibuat dalam bentuk program.**

***Keywords*—Josephus Problem, Rekursi, Bilangan Biner, Algoritma, Induksi Matematika.**

I. Pendahuluan

"Josephus Problem" atau Masalah Josephus adalah sebuah persoalan matematika yang diperkenalkan oleh seorang sejarawan Yahudi, Flavius Josephus pada abad pertama dalam bukunya *"The Jewish War"*. Diceritakan bahwa suatu ketika, Josephus beserta 40 pasukan, terkepung oleh tentara romawi. Dalam proses pengepungan itu, mereka terjebak dalam sebuah gua. Mereka lebih memilih bunuh diri daripada harus tertangkap tentara Romawi. Mereka akhirnya memutuskan untuk duduk melingkar, dan mulai berhitung dari 1 sampai 3, kemudian, jika mendapat urutan nomor 3, maka dia harus bunuh diri di tempat, hingga hanya tersisa satu orang (atau setidaknya ada 2 orang dalam kisah ini). Dikisahkan bahwa Josephus lebih memilih untuk ditangkap daripada bunuh diri, sehingga dia mulai melakukan perhitungan agar dia dan temannya tidak terbunuh. [1].

Dalam makalah ini, yang akan dibahas adalah apabila ada sejumlah orang, n misalnya, duduk melingkar. Kemudian setiap orang kedua akan dibunuh hingga tersisa satu orang, maka akan ditentukan siapa orang terakhir yang tetap hidup. Misalnya jika n = 7, maka kita dapat membuat lingkaran sebagai berikut.



Gambar 1. Ilustrasi Josephus problem dengan 7 orang.

Jadi, urutan eliminasi yang berjalan adalah 2,4,6,1,5. Dan hanya tersisa orang ke-7, sehingga orang ke-7 memenangkan undian. Namun, yang akan kita bahas bukanlah permasalahan dengan 7 orang, melainkan dengan n orang. Maka dari itu, kita akan mencoba merumuskan masalah josephus untuk n orang atau $J(n)$.

II. Landasan Teori

Untuk menyelesaikan dan memahami masalah ini, diperlukan beberapa teori – teori dasar yang menunjang. Teori – teori tersebut adalah sebagai berikut.

A. Rekursi

Adakalanya sesuatu yang ada di dunia tidak bisa didefinisikan dengan hal lain kecuali dirinya sendiri, hal inilah yang disebut sebagai rekursi. Rekursi banyak terjadi di sekitar kehidupan manusia, contoh termudahnya adalah pada bunga.

Gambar 2 . Bunga Rekursif
(Sumber : Flickr.com)

Dalam dunia matematika, rekursi didefisinikan sebagai sebuah fungsi yang memanggil dirinya sendiri. Terdapat 2 komponen penting dalam membuat sebuah fungsi rekursi, yaitu:

1. Basis,
   Basis sendiri dapat kita sebut sebagai sebuah 'Penghenti' dari jalannya rekursi, saat rekursi sudah menemui basis, maka fungsi tersebut akan berhenti memanggil fungsi tersebut kembali.
2. Rekurens
   Rekurens dapat kita sebut sebagai bagian dari fungsi rekursi yang berperan untuk melakukan pemanggilan pada fungsi itu -sendiri.

Sebagai contoh adalah fungsi Fibonacci berikut.

$f(n) = 1 \qquad n = 0 \qquad (1)$

$f(n) = 1 \qquad n = 1 \qquad (2)$

$$f(n) = f(n-1) + f(n-2) \quad (3)$$

Dalam kasus rumus rekurens untuk bilangan barisan Fibonacci kita dapat menyebut persamaan (1) dan (2) sebagai basis, dan persamaan (3) sebagai rekurensnya.

B. Bilangan Biner dan Operasinya

Bilangan yang sering kita temui dalam kehidupan sehari hari baik akademik maupun non-akademik biasanya didasarkan pada bilangan basis 10. Bilangan basis 10 sendiri adalah bilangan yang tersusun dari angka angka 0,1,2,3,4,5,6,7,8, atau 9. Contoh bilangan basis 10 adalah 19, 20009, 12345678, dll.

Bilangan biner sendiri, adalah sebuah bilangan berbasis 2, yaitu sebuah bilangan yang hanya tersusun dari 0 atau 1. Sistem bilangan biner modern pertama kali diperkenalkan oleh seorang filsuf dari jerman pada abad ke 17 yaitu Gottfried Wilhem Leibniz. Sistem bilangan biner kini telah berkembang dan menjadi basis dari setiap hal hal yang berbau digital. Bilangan biner sendiri sering kali dituliskan dalam bentuk Hexadecimal.

Cara mengkonversi bilangan basis 10 menjadi biner relatif mudah, yaitu dengan cara mengubah bilangan desimal menjadi penjumlahan dari bilangan 2 pangkat. Sebagai contoh

$$10 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

Maka bilangan biner yang mewakili angka 10 adalah 1010. Setiap digit dalam bilangan biner disebut sebagai bit, dan 4 bit dalam bilangan biner disebut Byte.

Bilangan biner, selain berlaku operasi biasa seperti penjumlahan, pengurangan, perkalian dan pembagian, juga memiliki operasi khusus yang disebut sebagai Operasi Bitwise. Operasi bitwise pada dasarnya kebanyakan didasarkan oleh operasi yang ada pada logika.

Dikutip dari bahan ajar Organisasi dan Arsitektur Komputer Institut Teknologi Bandung, berikut adalah beberapa operasi bitwise tersebut.

1. And (&)

   Hasil dari operasi And sesuai dengan tabel berikut.

   Tabel 1. Tabel Operasi And.

&	0	1
0	0	0
1	0	1

2. Or (|)

   Hasil dari operasi Or sesuai dengan tabel berikut.

   Tabel 2. Tabel Operasi Or.

|	0	1
0	0	1
1	1	1

3. Xor (^)

   Tabel 3. Tabel operasi Xor sesuai tabel berikut.

^	0	1
0	0	1
1	1	0

4. Negasi (~)

   Tabel 4. Tabel operasi Negasi sesuai tabel berikut.

~	
0	1
1	0

5. Left Shift (x << y)

   Left shift menggeser semua bit ke kiri dan membuang 'bit ekstra' di sebelah kiri. Mengisi dengan '0' pada di kanan.

6. Right Shift (x >> y)

   Right shift menggeser semua bit ke kanan, terdapat 2 jenis, Logical shift, dan Arithmetic shift.

   Untuk lebih mengerti terhadap shift, akan ditunjukan oleh tabel berikut.

   Tabel 5. Contoh operasi Shift

X	01100010	10100010
<< 3	00010000	00010000
Log. >> 2	00011000	00101000
Arith. >> 2	00011000	11101000

   Logical shift mengisi 0 disebelah kiri. Sedangkan untuk arithmetic shift mengisi bit pada sebelah kiri, dengan bit terkiri.

C. Induksi Matematika

Dikutip dari buku Matematika Diskrit, yang ditulis oleh Rinaldi Munir, dijelaskan bahwa Induksi matematika adalah suatu metode pembuktian proposisi bilangan bulat. Induksi memiliki 2 jenis prinsip yaitu prinsip induksi sederhana dan induksi kuat.

Disebutkan pula dalam buku tersebut bahwa prinsip induksi sederhana berbunyi :

Misalkan $p(n)$ adalah proposisi perihal bilangan bulat positif dan kita ingin membuktikan bahwa $p(n)$ benar untuk semua bilangan bulat positif n. Untuk membuktikan proposisi ini, kita hanya perlu menunjukkan bahwa:

1. $p(1)$ benar, dan

2. jika $p(n)$ benar, maka $p(n + 1)$ juga benar untuk setiap $n \geq 1$.

sehingga $p(n)$ benar untuk semua bilangan bulat positif n.

Langkah 1 kemudian disebut sebagai basis induksi sedangkan langkah 2 disebut langkah induksi.

Sedangkan langkah induksi kuat berbunyi :

Misalkan $p(n)$ adalah pernyataan perihal bilangan bu/at dan kita ingin membuktikan

bahwa $p(n)$ benar untuk semua bilangan bulat $n \geq n_0$ Untuk membuktikan ini,

kita hanya perlu menunjukkan bahwa:

1. $p(n_0)$ benar, dan

2. jika $p(no), p(no + 1), \ldots, p(n)$ benar, maka $p(n + 1)$ juga benar untuk setiap bilangan bulat $n \geq n_0$,

sehingga $p(n)$ benar untuk semua bilangan bulat $n \geq n_0$.

D. Algoritma dan Kompleksitasnya

Dikutip dari buku Matematika Diskrit yang ditulis oleh Rinaldi Munir, dijelaskan bahwa algoritma adalah suatu urutan sistematis untuk menyelesaikan suatu permasalahan. Algoritma tidak hanya harus benar, tetapi juga harus mangkus.

Untuk setiap persoalan, pasti terdapat sejumlah algoritma yang mampu menyelesaikannya, namun algoritma yang bagus adalah algoritma yang memiliki kompleksitas terkecil dibanding yang lain. Kompleksitas algoritma dinotasikan oleh huruf O besar yang kemudian disebut sebagai big-O. Berikut adalah gambar urutan kemangkusan suatu algoritma.

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

Gambar 3. Urutan Big-O

Sumber tabel : Buku Matematika Diskrit, Rinaldi Munir.

Dapat dilihat bahwa O(1) merupakan kompleksitas terbaik dan tercepat dari semua yang ada. Maka dari itu, programmer berlomba lomba untuk menyelesaikan suatu permasalahan dengan algoritma terbaik.

III. PEMBAHASAN

A. *Pendekatan Rekursif*

Untuk menyelesaikan persoalan ini kita dapat melakukan pendekatan sederhana dengan menuliskan angka mulai dari 1 hingga 41 yang membentuk sebuah lingkaran. Kemudian kita dapat melakukan pencoretan saat kita mendapati hitungan 3. Hal ini kita lakukan secara konstan hingga hanya tersisa 1 orang. Permodelan yang dimaksud akan ditunjukan oleh tabel berikut.

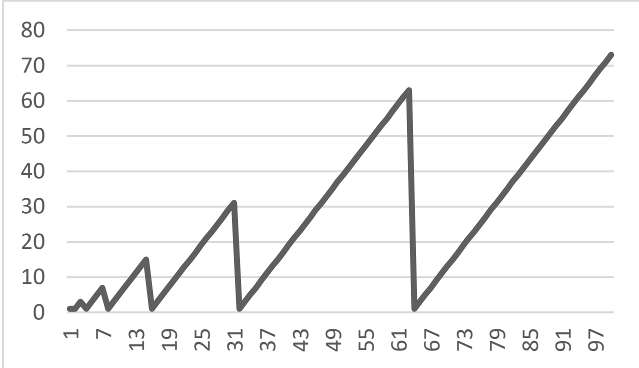
Tabel 6. Penyelesaian permasalahan asli josephus.

Putaran	Hasil
1	1 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
2	~~1~~ 2 4 ~~5~~ 7 8 ~~10~~ 11 13 ~~14~~ 16 17 ~~19~~ 20 22 ~~23~~ 25 26 ~~28~~ 29 31 ~~32~~ 34 35 ~~37~~ 38 40 ~~41~~
3	2 4 ~~7~~ 8 11 ~~13~~ 16 17 ~~20~~ 22 25 ~~26~~ 29 31 ~~34~~ 35 38 ~~40~~
4	2 4 ~~8~~ 11 16 ~~17~~ 22 25 ~~29~~ 31 35 ~~38~~
5	2 4 ~~11~~ 16 22 ~~25~~ 31 35
6	~~2~~ 4 16 ~~22~~ 31 35
7	~~4~~ 16 31 ~~35~~

Dari tabel dapat dilihat bahwa untuk tetap hidup, josephus dan temannya harus duduk di urutan ke -16 dan 31.

Sesuai dengan yang telah disebut pada pendahuluan, yang akan dibahas dalam makalah ini adalah permasalahan josephus yang sudah dimodifikasi. Modifikasi yang dimaksud adalah setiap orang yang mendapat nomor 2 akan mengakhiri hidupnya.

Untuk lebih memahami persoalan ini, kita dapat mulai mencoba melakukan permodelan sendiri diatas kertas. Pertama, untuk n = 1, maka sudah dipastikan bahwa satu orang tersebut yang akan tetap hidup. Selanjutnya untuk n = 2, dengan peraturan bahwa orang pertama akan membunuh orang setelahnya, maka orang pertama juga yang akan memenangkan permainan. Dengan mengulangi langkah ini hingga didapat n yang cukup besar, disini penulis mencoba hingga n = 100, didapat data dalam grafik 1 sebagai berikut.



Grafik 1. Grafik n terhadap pemenang.

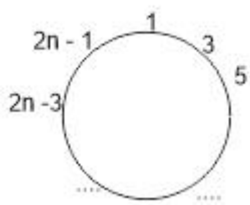
Dimana jika data tersebut diambil hanya dari 1 hingga 16 dan diletakan dalam tabel, maka akan terlihat lebih jelas hasilnya sebagai berikut.

Tabel 7. Data Percobaan Masalah Josephus.

N	Pemenang
1	1
2	1
3	3
4	1
5	3
6	5
7	7
8	1
9	3
10	5
11	7
12	9
13	11
14	13
15	15
16	1

Dari data tabel tersebut, kita dapat mengetahui bahwa pemenang selalu berada posisi bernomor ganjil. Alasannya tentu saja karena pada putaran pertama, semua yang bernomor genap pasti akan tereliminasi. Semisal kita mengambil sembarang n , dimana n adalah bilangan genap, maka perhitungan akan ke kembali ke awal. Bedanya adalah, pada putaran kedua, hanya tersisa setengah dari putaran pertama.

Maka dari itu, asumsikan kita memiliki sejumlah 2n orang yang duduk secara melingkar. Maka setelah melewati 1 putaran penuh. Orang yang masih hidup pada putaran kedua ditunjukan oleh ilustrasi berikut.



Gambar 4. Ilustrasi putaran kedua pada n genap.

Dari hal ini kita mendapatkan kesimpulan pertama, yaitu:

$$J(2n) = 2J(n) - 1 \qquad (1)$$

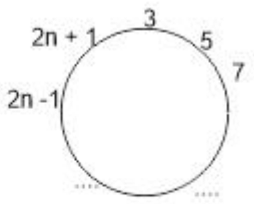
Lalu, kita akan mencoba rumus tersebut terhadap sebuah angka pada tabel untuk mengetahui hasilnya. Misalnya kita pilih n = 14, maka

$$J(14) = 2J(7) - 1 = 2.7 - 1 = 13$$

Dan rumus tersebut menghasilkan angka yang sama dengan tabel yang telah kita buat.

Sayangnya, rumus tersebut hanya berlaku untuk n yang merupakan bilangan genap, sedangkan untuk bilangan ganjil, kita harus mencari solusi lainnya.

Asumsikan kita memiliki 2n + 1 orang yang duduk melingkar, maka setelah putaran pertama usai, orang pertama pada lingkaran akan terbunuh, dan akan menyisakan orang sebagaimana ilustrasi berikut.



Gambar 5. ilustrasi pada putaran kedua pada n ganjil.

Putaran kedua pada bilangan ganjil hampir mirip dengan putaran kedua pada bilangan genap, hanya saja, pada bilangan ganjil, orang pertama merupakan bilangan pertama dari putaran sebelumnya ditambah dengan 2. Sehingga kita mendapat sebuah persamaan baru untuk kasus dimana n adalah bilangan ganjil sebagai berikut.

$$J(2n+1) = 2J(n) + 1 \qquad (2)$$

Selanjutnya kita akan menguji kebenaran persamaan (2) terhadap tabel yang telah dibuat sebelumnya, misalnya pilih n = 15 maka didapatkan

$$J(15) = 2J(7) + 1 = 2.7 + 1 = 15$$

Hasil ini sama persis dengan yang ada pada tabel yang telah dibuat sebelumnya.

Dengan menggabungkan persamaan (1) dan persamaan (2) serta dengan persamaan $J(1) = 1$. didapat sebuah persamaan umum yaitu:

$$\begin{aligned} &J(1) = 1 \\ &J(2n) = 2J(n) - 1 \; untuk \; n \ge 1 \\ &J(2n+1) = 2J(n) + 1 \; untuk \; n \ge 1 \qquad (3) \end{aligned}$$

Dengan ini, persamaan untuk menyelesaikan Josephus problem melalui rekursi sudah selesai. Tapi kita masih akan mencoba mencari bentuk yang lebih sederhana lagi dari persamaan yang akan menyelesaikan permasalahan kita.

Tabel 8. Data Percobaan Masalah Josephus.

N	Pemenang
1	1
2	1
3	3
4	1
5	3
6	5
7	7
8	1
9	3
10	5
11	7
12	9
13	11
14	13
15	15
16	1

Dari tabel 8 yang sudah kita modifikasi, kita dapat melihat bahwa pada nomor 1, 2, 8, 16 pemenang yang keluar selalu peserta dengan nomor urut 1. Dari beberapa bilangan tersebut kita dapat langsung menarik kesimpulan bahwa bilangan – bilangan tersebut adalah bilangan kelipatan dua. Maka untuk tiap nilai n kita dapat tulis

$$n = 2^a + b \qquad (4)$$

Dimana a adalah sembarang bilangan bulat positif dan b adalah sisa dari dari pembagian n terhadap bilangan kelipatan 2 yang terbesar darinya. Maka dapat disimpulkan ($a \ge 0; 0 \le b < 2^a$). Selanjutnya dapat kita simpulkan sebuah persamaan yaitu:

$$J(n) = J(2^a + b) = 2.b + 1 \qquad (5)$$

Rumus ini akan kita buktikan dengan menggunakan prinsip Induksi kuat. Pertama untuk basis, kita pilih $n = 1$, maka:

$$J(1) = J(2^0 + 0) = 2.0 + 1 = 1$$

Dengan mencocokan data pada tabel maka kita dapat menyimpulkan bahwa basisnya benar.

Untuk bagian Induksi, terdapat 2 kasus, dimana n adalah bilangan genap dan n adalah bilangan ganjil. Untuk bilangan genap, maka kita pilih b_1 dan a_1 dimana $\frac{n}{2} = 2^{a_1} + b_1$ dan $0 \le b_1 < a_1$. perlu digaris bawahi bahwa $b_1 = \frac{b}{2}$.

Bukti :

$$\frac{n}{2} = \frac{2^a + b}{2} = 2^{a-1} + \frac{b}{2} = 2^{a_1} + b_1 \qquad (6)$$

Maka sesuai dengan persamaan (3) di awal tadi:

$$J(n) = 2J\left(\frac{n}{2}\right) - 1 = 2(2.b_1 + 1) - 1 = 2b + 1$$

Maka induksi untuk n merupakan bilangan genap benar.

Untuk n merupakan bilangan ganjil, pilih b_1 dan a_1 dimana $\frac{n-1}{2} = 2^{a_1} + b_1$ dan $0 \le b_1 < a_1$. perlu digaris bawahi bahwa $b_1 = \frac{b-1}{2}$.

Bukti :

$$\frac{n-1}{2} = \frac{2^a + b - 1}{2} = 2^{a-1} + \frac{b-1}{2} = 2^{a_1} + b_1 \qquad (7)$$

Maka sesuai dengan persamaan 3 diatas :

$$J(n) = 2J\left(\frac{n-1}{2}\right) + 1 = 2(2.b_1 + 1) + 1 = 2b + 1$$

Maka induksi untuk n merupakan bilangan ganjil terbukti benar. Dengan ini, induksi kuat telah selesai, dan persamaan (5) terbukti benar.

Sehingga kita dapat menarik kesimpulan dari bahwa rumus lain dari pendekatan rekursif adalah

$$J(n) = J(2^a + b) = 2.b + 1$$

B. Pendekatan Bilangan Biner

Penyelesaian masalah ini dengan menggunakan pendekatan biner didasari oleh persamaan (5) pada bagian sebelumnya. Dasar yang dimaksud adalah adanya bilangan dengan kelipatan 2 yang menjadi "basis" nya. Untuk itu, penulis telah membuat tabel baru dari tabel 1 dengan bilangan biner 4 bit yang mewakili angka tersebut.

Tabel 9. Pemenang dari josephus problem dengan bilangan biner

N	N dalam Biner	Pemenang	Pemenang dalam biner
1	1	1	1
2	10	1	1
3	11	3	11
4	100	1	1
5	101	3	11
6	110	5	101
7	111	7	111
8	1000	1	1
9	1001	3	11
10	1010	5	101
11	1011	7	111
12	1100	9	1001
13	1101	11	1011
14	1110	13	1101
15	1111	15	1111
16	10000	16	1

Dari tabel 9, kita dapat melihat sebuah pola yang cukup menarik untuk diamati, pola tersebut adalah kita hanya perlu mengambil bit '1' terkiri dari representasi bitnya, kemudian kita letakan pada posisi terkanan representasi bit tersebut.

Sebagai contoh, misalnya 6, representasi bit dari 6 adalah '110', jika kita ambil 1 terkiri maka akan menyisakan '10' kemudian kita letakan 1 tersebut di tempat paling kanan, maka akan menghasilkan '101'. '101' sendiri merupakan representasi biner dari angka 5, dimana pernyataan ini benar sesuai dengan tabel yang kita buat sebelumnya.

C. Program

Program pertama terkait dengan penyelesaian permasalahan yang didasarkan pada pendekatan rekursi. Berikut adalah potongan program yang bersangkutan.

```
program josephus(n : integer) -> integer

Kamus Lokal

Algoritma
if n is 1
{*Basis*}
    -> 1
else if n is even
{*Rekurens untuk n, merupakan bil. genap*}
    -> 2*josephus(n/2) - 1
else
{*Rekurens untuk n, merupakan bil. ganjil*}
    -> 2*josephus((n-1)/2) + 1
```

Gambar 6 Potongan program josephus variasi 1.

Kemudian kita akan menganalisa kompleksitasnya. Pada program tersebut, kasus terbaik adalah:

$$Josephus_{min}(n) = 1$$

Sedangkan untuk kasus terburuknya adalah:

$$Josephus_{max}(n) = \log_2 n + 1$$

Sehingga dapat kita tarik kesimpulan bahwa Kompleksitas dari program Josephus

$$O(logn)$$

Program kedua terkait dengan penyelesaian permasalahan yang didasarkan pada pendekatan dengan bilangan biner. Berikut adalah program yang bersangkutan.

```
program josephus_bit( input/output n : integer)

Kamus lokal
j : integer
pos : integer
{* pos adalah sebuah program untuk mendapatkan posisi
dari biner '1' terbesar *}

Algoritma
{* Gunakan j untuk menyimpan bit '1' terkiri *}
j <- 1 << (pos - 1)

{* Ubah bit terkiri dari n dari '1' menjadi '0'
    contoh : 10001^10000 = 00001 *}
n <- n ^ j

{* Left Shift untuk menambah 1 buah angka 0 di paling kanan
    contoh : 00001<<1 = 00010 *}
n = n << 1;

{* Ubah bit '0' terkanan menjadi '1'
    contoh : 00010 | 1 = 00011 *}
n = n | 1
```

Gambar 7. Potongan program josephus variasi 2.

Sama seperti potongan pertama, kita akan melakukan analisasi terhadap kompleksitasnya. Untuk kasus terbaik:

$$JosephusBit_{min}(n) = 1$$

Sedangkan untuk kasus terburuknya adalah:

$$Josephus_{max}(n) = \log_2 n + 1$$

*Dari jumlah pencarian posisi 1 terkiri.

Jadi, kompleksitas dari operasi bit ini adalah :

$$O(logn)$$

Dari kedua program tersebut dapat ditarik kesimpulan bahwa keduanya memiliki kompleksitas yang sama. Analisa nya adalah karena pendekatan bit diturunkan dari bentuk lain rekursi

IV. KESIMPULAN

Penyelesaian 'Josephus Problem' dapat dilakukan dengan 2 metode pendekatan, yaitu dengan rekursi, dan pendekatan biner. Rumus untuk menyelesaikan dengan pendekatan rekusif adalah

$J(1) = 1$

$J(2n) = 2J(n) - 1\ untuk\ n \geq 1$

$J(2n + 1) = 2J(n) + 1\ untuk\ n \geq 1$

Atau

$J(n) = J(2^a + b) = 2.b + 1$

Untuk pendekatan biner, yang dilakukan adalah cukup memindahkan bit 1 terkiri dan diletakan di posisi terkanan.Dalam bentuk program, baik penggunaan pendekatan rekursif maupun bit memiliki kompleksitas yang sama yaitu: $O(logn)$

V. UCAPAN TERIMA KASIH

Ucapan terima kasih penulis haturkan sebesar – besar nya kepada pihak – pihak yang telah membantu pengerjaan makalah ini baik secara langsung maupun tidak langsung. Terutama kepada Allah SWT yang telah memberi segala nikmat untuk mengerjakan makalah.

REFERENSI

[1] Graham – Knuth – Patashnik, Concrete Mathematics 2nd ed., Addison-Wesley, 1988. hal. 9.

[2] Munir, Rinaldi, Matematika Diskrit Revisi Keempat. Bandung: Informatika Bandung, 2005.

[3] https://www.geeksforgeeks.org/josephus-problem-using-bit-magic/ diakses pada 9 Desember 2018.

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, 3 Desember 2017

Mohammad Ridwan Hady Arifin