# Penerapan *Linear Congruential Generator* pada Penentuan Blok yang Jatuh dalam Permainan Tetris

Muhammad Izzat Jundy - 13523092[1]
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
[1]*izzatjundy04@gmail.com*, *13523092@std.stei.itb.ac.id*

***Abstrak*— Permainan Tetris, yang pertama kali diciptakan pada tahun 1984, mengandalkan urutan tetromino yang jatuh secara acak untuk menciptakan tantangan dan dinamika permainan. Makalah ini mengimplementasikan *Linear Congruential Generator* (LCG) untuk menentukan urutan tetromino, dengan menggabungkan hasil LCG dengan waktu saat permainan dimulai (*current time*) untuk meningkatkan keacakan dan mengurangi pola berulang. Hasil implementasi menunjukkan bahwa penggabungan kedua faktor ini menghasilkan distribusi tetromino yang lebih merata dan acak, meningkatkan tingkat kesulitan permainan. Pemilihan parameter LCG yang tepat sangat penting untuk memastikan kualitas keacakan yang dihasilkan. Dengan demikian, makalah ini menyarankan penggunaan LCG yang dipadukan dengan *current time* sebagai metode untuk menghasilkan urutan tetromino yang lebih dinamis, serta membuka peluang untuk pengembangan algoritma pengacakan yang lebih kompleks dalam permainan video.**

***Kata kunci*—Tetris, linear congruential generator, aritmetika modular, rekursi.**

## I. Pendahuluan

Permainan Tetris, yang pertama kali diciptakan oleh Alexey Pajitnov pada tahun 1984, telah menjadi salah satu permainan paling populer di dunia game. Dalam Tetris, pemain harus menata berbagai bentuk balok, yang disebut tetromino, yang jatuh dari atas layar. Setiap tetromino yang muncul memiliki bentuk yang berbeda-beda, dan pemain harus mengaturnya sedemikian rupa untuk mengisi ruang kosong di bagian bawah layar. Salah satu hal yang penting dalam permainan ini adalah urutan munculnya tetromino, yang secara langsung mempengaruhi tingkat kesulitan dan dinamika permainan.

Untuk memberikan pengalaman permainan yang menyenangkan dan menantang, urutan tetromino yang muncul harus bersifat acak. Namun, menciptakan bilangan acak dalam komputer bukanlah hal yang mudah, karena komputer pada dasarnya bekerja dengan algoritma yang deterministik. Oleh karena itu, pseudo-random number generators (PRNG) digunakan untuk menghasilkan angka yang tampak acak, meskipun sesungguhnya dihasilkan berdasarkan algoritma yang dapat diprediksi jika parameternya diketahui.

Salah satu metode yang sering digunakan dalam menghasilkan angka acak adalah Linear Congruential Generator (LCG). LCG merupakan algoritma sederhana dan efisien yang menggunakan operasi aritmetika modular untuk menghasilkan urutan angka yang tampak acak. Namun, meskipun LCG cukup populer dalam berbagai aplikasi, pola yang dihasilkan oleh LCG dapat memiliki periode terbatas dan distribusi yang tidak selalu merata.

Pada makalah ini, akan dilakukan penerapan LCG dalam penentuan urutan tetromino pada permainan Tetris, dengan menggabungkan hasil dari LCG dengan current time sebagai faktor tambahan. Penambahan waktu bertujuan untuk meningkatkan kerandoman dan mengurangi kemungkinan terjadinya pola berulang yang bisa mengurangi tantangan permainan. Dengan cara ini, setiap tetromino yang muncul akan bergantung pada dua faktor: angka yang dihasilkan oleh LCG dan nilai waktu yang terus berubah, menghasilkan urutan yang lebih dinamis dan sulit diprediksi.

Pada makalah ini, akan dianalisis apakah penggabungan hasil LCG dengan *current time* dapat menghasilkan distribusi tetromino yang lebih merata dan lebih acak, serta bagaimana hal ini memengaruhi pengalaman bermain Tetris.

## II. Dasar Teori

### A. Aritmetika Modular

Aritmetika modular adalah cabang dari matematika yang mempelajari operasi aritmetika pada bilangan bulat dengan menggunakan modulus tertentu. Secara sederhana, aritmetika modular menangani bagaimana angka "berputar kembali" setelah mencapai batas tertentu. Operasi yang paling umum dalam aritmetika modular adalah operasi modulus itu sendiri, yang menghasilkan sisa pembagian dari suatu bilangan dibagi dengan bilangan lainnya. Lebih jelasnya, pada (1), ketika $a$ dimodulus dengan $m$ akan menghasilkan $r$, yang merupakan sisa dari $a$ dibagi dengan $m$, dengan $r$ memenuhi $0 \leq r \leq m$.

$$a \bmod m = r \tag{1}$$

### B. Rekursi

Rekursi adalah konsep dalam matematika dan ilmu computer yaitu suatu fungsi atau proses memanggil dirinya sendiri untuk menyelesaikan masalah. Dalam konteks algoritma, rekursi digunakan untuk memecahkan masalah yang bisa dipecah menjadi submasalah yang lebih kecil dengan struktur yang mirip

dengan masalah asli. Secara formal, suatu fungsi $f(x)$ dikatakan rekursif jika ia memanggil dirinya sendiri dalam proses perhitungannya. Fungsi rekursi memiliki dua bagian utama, yaitu basis dan rekursif. Basis adalah keadaan dasar, yaitu sebuah kondisi yang tidak memanggil fungsi lain lagi. Hal ini lah yang menghentikan proses rekursif. Sementara rekursif sendiri adalah proses pemanggilan fungsi lain. Contoh dari fungsi rekursif adalah (2).

$$f(x) = f(x-1) + 3, \quad x > 0$$
$$f(0) = 7018 \tag{2}$$

Dengan $f(0)$ sebagai basis dan $f(x)$ sebagai rekursif.

### C. Linear Congruential Generator

*Linear Congruential Generator* (LCG) adalah salah satu metode paling sederhana dan banyak digunakan untuk menghasilkan bilangan acak semu (*pseudo-random numbers*) dalam komputer. LCG didasarkan pada prinsip rekursi linier, yang mengubah bilangan ke bilangan berikutnya dengan menggunakan rumus matematika yang sangat sederhana. Algoritma ini dijabarkan dalam persamaan rekursif (3).

$$X_{n+1} = (aX_n + c) \bmod m \tag{3}$$

Dengan $X_n$ adalah bilangan acak ke-$n$, $a$ adalah faktor pengali, $c$ adalah konstanta penambah, $m$ adalah nilai modulus yang menentukan rentang nilai hasil, dan $X_0$ merupakan *seed* atau nilai awal rekurens. Persamaan rekursif tersebut menghasilkan urutan bilangan acak $X_1, X_2, X_3, \ldots$ yang dihasilkan secara rekursif. Kendatipun algoritma ini sederhana, hasil yang diperoleh bisa sangat berguna dalam banyak hal, seperti enkripsi dan dalam permainan video untuk menentukan elemen acak, seperti urutan objek yang muncul.

Untuk menghasilkan urutan angka yang optimal, pemilihan parameter $a$, $c$, dan $m$ sangat penting. Berdasarkan Teori Hull-Dobell, terdapat beberapa syarat yang harus dipenuhi untuk memastikan angka yang dihasilkan memiliki periodisitas panjang dan distribusi yang baik, yaitu

1. $c$ dan $m$ harus saling relative prima,
2. $a - 1$ harus habis dibagi setiap faktor prima dari $m$, dan
3. jika $m$ adalah kelipatan dari 4, $a - 1$ harus dibagi 4.

Dengan memilih parameter yang tepat, LCG dapat menghasilkan urutan yang sangat panjang sebelum nantinya mulai berulang, dan dengan demikian menghasilkan bilangan yang lebih acak.

### D. Tetris

Tetris adalah sebuah permainan video yang dirancang oleh Alexey Pajitnov pada tahun 1984. Permainan ini adalah salah satu permainan video paling ikonik sepanjang masa, dikenal karena mekanismenya yang sederhana tetapi menantang, membuatnya dapat dimainkan oleh berbagai kalangan pemain.

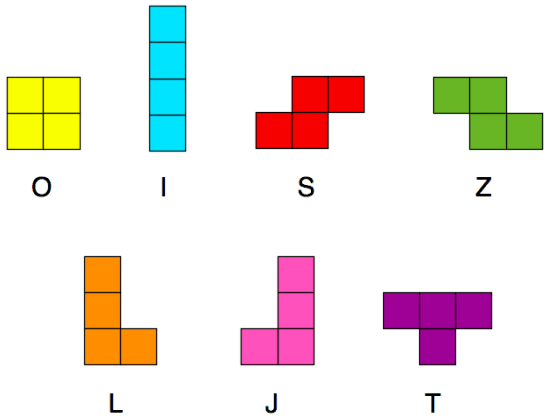
#### D.1. Mekanisme Permainan

Dalam Tetris, pemain mengendalikan blok-blok berbentuk yang disebut tetromino. Tetromino, sesuai namanya yang mengandung "tetra" yang berarti empat, adalah bentuk yang terdiri dari empat kotak kecil yang disusun sedemikian rupa agar beragam. Blok-blok ini jatuh secara vertikal dari bagian atas layar, dan pemain harus memutar atau menggeser blok untuk menyusunnya di bagian bawah layar. Ketika satu baris horizontal penuh terisi tanpa celah, baris tersebut akan terhapus, memberikan poin kepada pemain. Untuk setiap penghapusan baris, semakin banyak jumlah baris yang dihapus dalam sekali penghapusan, semakin banyak pula poin yang akan didapatkan oleh pemain. Penghapusan empat baris sekaligus sering disebut "Tetris".

Tujuan dari permainan ini adalah untuk mencegah blok-blok memenuhi layar hingga ke bagian atas, yang akan menyebabkan permainan berakhir, serta mendapatkan poin sebanyak-banyaknya sebelum hal tersebut terjadi.

#### D.2. Jenis-Jenis Tetromino

Terdapat tujuh jenis tetramino yang ada pada tetris, yaitu I, O, T, L, J, S, dan Z. Bentuknya dapat dilihat pada gambar 1.




Gambar 1. Jenis-jenis tetromino dalam permainan Tetris

Sumber: https://learnopencv.com/wp-content/uploads/2020/11/tetris-pieces.png

Pada D.1. disampaikan bahwa poin maksimal akan didapat ketika mendapatkan penghapusan empat baris sekaligus dalam sekali penghapusan. Hal tersebut hanya bisa tercapai apabila menggunakan tetromino I sebagai "pelengkap terakhir" dari keempat baris tersebut.

## III. Implementasi

### A. Tujuan Implementasi dan Pemilihan Bahasa Pemrograman

Implementasi dilakukan untuk menyimulasikan algoritma pengacakan menggunakan *Linear Congurential Generator* (LCG) dikombinasikan dengan *current time* untuk penentuan blok yang jatuh dalam permainan Tetris. Implementasi ini hanya akan menampilkan blok apa saja yang akan turun. Pada teorinya, terdapat variabel *current time* yang akan variatif tergantung blok yang jatuh ditaruh pada ketinggian berapa. Semakin tinggi letak ditaruhnya blok, semakin singkat durasi jatuhnya. Namun, karena keterbatasan kemampuan penulis dan keterbatasan waktu yang ada, permainan Tetris tidak benar-benar diimplementasikan pada makalah ini. Variabel *current time*

yang seharusnya bervariasi tergantung bagaimana pemain memainkan permainannya, menjadi tidak masuk dalam variabel yang digunakan.

Implementasi ini menggunakan bahasa pemrograman C, yang dipilih karena implementasi yang dilakukan tidak memerlukan bahasa yang terlalu tinggi, dan karena tersedianya library "time" yang sangat digunakan pada implementasi ini. Bahasa pemrograman C juga merupakan bahasa yang sangat cepat untuk di-*compile*, sehingga mempersingkat waktu implementasi. Bahasa pemrograman ini juga menyenangkan untuk digunakan karena tingkat kesulitannya yang tidak terlalu sulit, tetapi juga tidak terlalu mudah.

### *B. Pemilihan Parameter dalam Linear Congruential Generator*

Terdapat beberapa parameter yang perlu ditentukan di awal dalam penggunaan *Linear Congruential Generator* (LCG), yaitu $a$, $c$, dan $m$. Nilai $a$ yang dipilih adalah 1103515245, nilai $c$ yang dipilih adalah 12345, dan nilai $m$ yang dipilih adalah 2147483648 yang senilai dengan $2^{31}$. Nilai $a$ tersebut dipilih karena telah dianjurkan oleh berbagai referensi untuk implementasi LCG dalam C. Sementara nilai $m$ diperoleh dari batas nilai positif dari *integer* di C. Nilai $c$, mengikuti nilai $m$, merupakan nilai yang relatif prima terhadap nilai $m$ ($GCD(c, m) = 1$).

```c
unsigned long a = 1103515245;
unsigned long c = 12345;
unsigned long m = 2147483648;

// Linear Congruential Generator
unsigned long nextRandom(unsigned long seed) {
    return (a * seed + c) % m;
}
```

Gambar 2. Deklarasi variabel $a$, $c$, dan $m$ diikuti deklarasi dan implementasi fungsi *linear congruential generator*

### *C. Pemilihan Seed Sebagai Basis dari Fungsi Rekursif Linear Congruential Generator*

*Seed* yang digunakan pada implementasi ini adalah waktu terkini (*current time*) saat program dijalankan. Pada teorinya, *seed* ini didasari oleh waktu tepat saat dimulainya permainan, bukan saat program dijalankan. Namun, karena ini hanya implementasi sederhana, *seed* yang dipilih adalah *current time* saat program di-*run*.

```c
unsigned long seed = (unsigned long)time(NULL);
```

Gambar 3. Implementasi deklarasi *seed* sebagai *current time*

### *D. Penggabungan Hasil Linear Congruential Generator dengan Current Time untuk Setiap Blok*

*Seed* yang digunakan pada fungsi *Linear Congruential Generator* (LCG) menggunakan hasil dari LCG dengan *seed* sebelumnya, sebagaimana konsep rekursi. Hasil LCG tersebut, sebelum digunakan untuk *seed* pemanggilan fungsi LCG berikutnya, digunakan terlebih dahulu pada penentuan blok yang akan jatuh selanjutnya. Penggunaannya dipadu dengan *current time* dengan persamaan (4).

$$Indeks\ Tetromino = (Hasil\ LCG + Current\ Time)\ mod\ 7 \quad (4)$$

Modulo dengan 7 dilakukan untuk memastikan indeks tetromino berada pada rentang 0-6 inklusif, sesuai dengan jumlah jenis tetromino yang ada yaitu sebanyak 7.

```c
unsigned long lcg = seed;
printf("Urutan Tetromino yang Jatuh:\n");
for (int i = 0; i < 10; i++) {
    lcg = nextRandom(lcg);
    unsigned long current_time = (unsigned long)time(NULL);

    unsigned long tetromino = (lcg + current_time) % 7;

    switch (tetromino) {
        case 0: printf("I\n"); break;
        case 1: printf("O\n"); break;
        case 2: printf("T\n"); break;
        case 3: printf("L\n"); break;
        case 4: printf("J\n"); break;
        case 5: printf("S\n"); break;
        case 6: printf("Z\n"); break;
    }
}
```

Gambar 4. Implementasi penggabungan hasil LCG dengan *current time*

### *E. Menampilkan Hasil*

Setiap jenis tetromino diberikan indeks unik dari 0 hingga 6. Dimulai dari 0, urutan pemberian indeksnya adalah I, O, T, L, J, S, dan terakhir Z. Program menampilkan nama jenis tetromino sesuai dengan nilai yang didapat dari hasil pengacakan.

```c
switch (tetromino) {
    case 0: printf("I\n"); break;
    case 1: printf("O\n"); break;
    case 2: printf("T\n"); break;
    case 3: printf("L\n"); break;
    case 4: printf("J\n"); break;
    case 5: printf("S\n"); break;
    case 6: printf("Z\n"); break;
}
```

Gambar 5. Implementasi menampilkan hasil dari pengacakan

### *F. Contoh Keluaran*

Contoh keluaran dari implementasi ini dapat dilihat pada gambar 6.

```
Urutan Tetromino yang Jatuh:
J
Z
J
I
Z
O
Z
O
T
J
```

Gambar 6. Contoh keluaran dari implementasi

## IV. KESIMPULAN

Permainan Tetris, yang pertama kali diciptakan oleh Alexey Pajitnov pada tahun 1984, membutuhkan urutan tetromino yang jatuh secara acak untuk menciptakan dinamika permainan yang menantang. Dalam makalah ini, diterapkan *Linear Congruential Generator* (LCG) untuk menghasilkan urutan acak tersebut, dengan menggabungkannya dengan waktu saat permainan dimulai (*current time*) untuk meningkatkan keacakan.

Hasil dari implementasi ini menunjukkan bahwa penggabungan LCG dengan *current time* memberikan distribusi tetromino yang lebih acak dan merata. Dengan memanfaatkan waktu saat permainan dimulai sebagai *seed* dan menghitung urutan tetromino dengan operasi modulus, urutan tetromino yang dihasilkan menjadi lebih dinamis dan sulit diprediksi, yang meningkatkan tantangan permainan.

Selain itu, pemilihan parameter yang tepat dalam LCG, seperti nilai $a$, $c$, dan $m$, sangat memengaruhi kualitas keacakan yang dihasilkan. Meskipun LCG memiliki keterbatasan dalam hal periode dan distribusi acak, penggabungannya dengan *current time* membantu memperbaiki hal tersebut.

Kesimpulannya, penggunaan LCG yang dikombinasikan dengan waktu sistem dapat menghasilkan urutan tetromino yang lebih acak, sehingga membuat permainan Tetris lebih menyenangkan dengan memberikan tantangan yang lebih dinamis dan sulit diprediksi.

## V. SARAN

Untuk pengembangan lebih lanjut, disarankan untuk menggunakan algoritma pengacakan yang lebih canggih, seperti Mersenne Twister atau Xorshift, untuk mengatasi keterbatasan distribusi dan periode yang ada pada *Linear Congruential Generator* (LCG). Selain itu, pemilihan parameter LCG yang lebih optimal dapat dilakukan melalui eksperimen untuk menghasilkan urutan acak yang lebih baik. Penggunaan waktu yang lebih dinamis, seperti interval antara jatuhnya tetromino atau waktu permainan yang sebenarnya, dapat meningkatkan keacakan. Untuk meningkatkan pengalaman bermain, disarankan juga untuk mengimplementasikan grafis permainan Tetris yang lengkap. Disarankan juga untuk mempertimbangkan pengembangan algoritma pengacakan yang lebih maksimal untuk permainan lainnya juga.

## VI. APPENDIX

Appendixes, if needed, appear before the acknowledgment.

## VII. ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ... .” Sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page.

## REFERENSI

[1] Arsenault, D. (2017). *Game design patterns for player immersion: The Tetris effect revisited*. In Proceedings of the DiGRA 2017 Conference. Diambil dari http://www.digra.org/digital-library/

[2] Bogost, I. (2007). *Persuasive Games: The Expressive Power of Videogames*. The MIT Press.

[3] Hatfield, D. (2007). *History of Tetris*. IGN. Diambil dari https://www.ign.com/articles/2007/05/17/history-of-tetris

[4] Knuth, D. E. (1997). *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (3rd ed.). Addison-Wesley.

[5] Knuth, D. E. (1997). *The Art of Computer Programming, Volume 2: Seminumerical algorithms* (3rd ed.). Addison-Wesley.

[6] Levin, I., & Munro, P. (2013). Random number generation: The linear congruential generator. *Journal of Statistical Software, 52*(5), 1–25. https://doi.org/10.18637/jss.v052.i05

[7] Parker, J. (1995). *The C programming language* (2nd ed.). Prentice Hall.

[8] Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes: The art of scientific computing* (3rd ed.). Cambridge University Press.

[9] Rosen, K. H. (2012). *Discrete Mathematics and Its Applications* (7th ed.). McGraw-Hill.

[10] Thompson, C. (2009). *Tetris: The games people play*. Wired Magazine. Diab https://www.wired.com/2009/05/ff-tetris/

## 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, 8 Januari 2025

Muhammad Izzat Jundy (13523092)