

**Laporan**  
**Tugas Kecil 1 IF2211 Strategi Algoritma**  
**Penyelesaian *24 Game* dengan *Algoritma Brute Force***



oleh  
Ignatius Timothy Manullang / 13517044

PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
BANDUNG  
2019

# BAB I

## ALGORITMA *BRUTE FORCE*

### 1.1 Pengantar

Dalam 24 solver ini, digunakan algoritma brute force. Metode algoritma brute force yang digunakan adalah exhaustive search, yang dibagi menjadi tiga tahap, yaitu enumerasi, evaluasi, dan penampilan solusi.

### 1.2 Enumerasi

Dalam 24 solver ini, hal yang dienumerasi adalah setiap ekspresi aritmatika yang mungkin dibuat dengan 4 nilai angka yang diinput. Ekspresi aritmatika ini dibentuk dengan menggabungkan 4 angka, 3 operator dan 2 pasang bracket, sehingga dalam proses ini, terdapat permutasi 4 angka, permutasi 4 kemungkinan operator dalam 3 posisi operator, dan kemungkinan 2 pasang bracket yang di tengahnya terdapat 2 angka atau ekspresi aritmatika dalam kurung, dan 1 operator, yaitu

1.  $(n_0 \text{ op}_0 n_1) \text{ op}_1 (n_2 \text{ op}_2 n_3)$
2.  $((n_0 \text{ op}_1 n_1) \text{ op}_2 n_2) \text{ op}_3 n_3$
3.  $(n_0 \text{ op}_0 (n_1 \text{ op}_1 n_2)) \text{ op}_2 n_3$
4.  $n_0 \text{ op}_0 ((n_1 \text{ op}_1 n_2) \text{ op}_2 n_3)$
5.  $n_0 \text{ op}_1 (n_1 \text{ op}_2 (n_2 \text{ op}_3 n_3))$

dengan  $n$  adalah angka, dan  $op$  adalah operator, dan 0-3 adalah urutan dalam ekspresi aritmatika.

### 1.3 Evaluasi

Dalam 24 solver ini, hal yang dievaluasi adalah setiap string ekspresi aritmatika yang sudah dibentuk oleh proses enumerasi. Proses evaluasi ini menggunakan recursive descent parser.

### 1.4 Penampilan Solusi

Setelah hasilnya dievaluasi, setiap ekspresi aritmatika yang menghasilkan 24 akan ditampilkan ke layar.

## BAB II

### *SOURCE CODE PROGRAM*

#### 2.1 Source Code C++

```
3  #include <chrono> // Untuk menghitung waktu
4  #include <iostream> // Untuk input/output
5  #include <string> // Untuk proses tipe data string
6  using namespace std;
7
8  // Inisialisasi variabel
9  string str; // Variabel string untuk membentuk string ekspresi aritmatika
10 string op = "+-*/"; // Variabel untuk menampung seluruh operator aritmatika
11 int number[4]; // Variabel untuk menampung seluruh angka yang diinput
12 double result; // Variabel untuk hasil perhitungan aritmatika
13 int solution_counter = 0; // Variabel untuk menghitung jumlah solusi
14 const char* expression_string; // Variabel untuk string ekspresi aritmatika yang
15                                // akan diparse menggunakan recursive descent parser
16
17 //-----Recursive Descent Parser-----
18 char current_element()
19 // Fungsi untuk mendapatkan elemen yang diproses
20 {
21     return *expression_string;
22 }
23
24 char advance()
25 // Fungsi untuk memajukan satu elemen dari string
26 {
27     return *expression_string++;
28 }
29
30 double numbers()
31 // Fungsi untuk parsing angka
32 {
33     int result = advance() - '0';
34     while (current_element() >= '0' && current_element() <= '9')
35     {
36         result = 10*result + advance() - '0';
37     }
38     return result;
39 }
40
41 double expression();
42 //Definisi dari expression() yang akan digunakan dalam fungsi factor()
43
44 double factor()
45 //Fungsi untuk parsing angka, tanda negatif, dan tanda kurung
46 {
47     if (current_element() >= '0' && current_element() <= '9')
48     {
```

```

49     return numbers();
50 }
51 else if (current_element() == '(')
52 {
53     advance(); // skip '('
54     double result = expression();
55     advance(); // skip ')'
56     return result;
57 }
58 else if (current_element() == '-')
59 {
60     advance(); // skip '-'
61     return -factor();
62 }
63 return 0; // ketika error, langsung return
64 }
65
66 double term()
67 //Fungsi untuk parsing operator * dan /
68 {
69     double result = factor();
70     while (current_element() == '*' || current_element() == '/')
71     {
72         if (advance() == '*')
73         {
74             result *= factor();
75         }
76         else
77         {
78             result /= factor();
79         }
80     }
81     return result;
82 }
83
84 double expression()
85 // Fungsi untuk parsing operator + dan -
86 {
87     double result = term();
88     while (current_element() == '+' || current_element() == '-')
89     {
90         if (advance() == '+')
91         {
92             result += term();
93         }
94         else
95         {
96             result -= term();
97         }
98     }
99     return result;
100 }
101
102 //-----Bracket Permutation-----
103 void bracket_permutation_1(int angka[], int i, int j, int k)
104 // (n0 op0 n1) op1 (n2 op2 n3)
105 // n adalah angka, dan op adalah operator
106 // Contoh: (6 + 6) + (6 + 6)
107 {
108     str = "";
109     str.append("(");
110     str = str + to_string(angka[0]);
111     str.append(op,i,1);

```

```

112     str = str + to_string(angka[1]);
113     str.append("(");
114     str.append(op,j,1);
115     str.append("(");
116     str = str + to_string(angka[2]);
117     str.append(op,k,1);
118     str = str + to_string(angka[3]);
119     str.append(")");
120     expression_string = str.c_str();
121     result = expression();
122     // Evaluasikan permutasi ekspresi aritmatika yang ada, yang diambil hanya yang
    hasilnya 24
123     if ((result > 23.999999999) && (result < 24.000000001))
124     {
125         solution_counter++;
126         cout << solution_counter << " " << str << endl;
127     }
128 }
129
130 void bracket_permutation_2(int angka[], int i, int j, int k)
131 // ((n0 op1 n1) op2 n2) op3 n3
132 // n adalah angka, dan op adalah operator
133 // Contoh: ((6 + 6) + 6) + 6
134 {
135     str = "";
136     str.append("(");
137     str.append("(");
138     str = str + to_string(angka[0]);
139     str.append(op,i,1);
140     str = str + to_string(angka[1]);
141     str.append(")");
142     str.append(op,j,1);
143     str = str + to_string(angka[2]);
144     str.append(")");
145     str.append(op,k,1);
146     str = str + to_string(angka[3]);
147     expression_string = str.c_str();
148     result = expression();
149     // Evaluasikan permutasi ekspresi aritmatika yang ada, yang diambil hanya yang
    hasilnya 24
150     if ((result > 23.999999999) && (result < 24.000000001))
151     {
152         solution_counter++;
153         cout << solution_counter << " " << str << endl;
154     }
155 }
156
157 void bracket_permutation_3(int angka[], int i, int j, int k)
158 // (n0 op0 (n1 op1 n2)) op2 n3
159 // n adalah angka, dan op adalah operator
160 // Contoh: (6 + (6 + 6)) + 6
161 {
162     str = "";
163     str.append("(");
164     str = str + to_string(angka[0]);
165     str.append(op,i,1);
166     str.append("(");
167     str = str + to_string(angka[1]);
168     str.append(op,j,1);
169     str = str + to_string(angka[2]);
170     str.append(")");
171     str.append(")");
172     str.append(op,k,1);

```

```

173     str = str + to_string(angka[3]);
174     expression_string = str.c_str();
175     result = expression();
176     // Evaluasikan permutasi ekspresi aritmatika yang ada, yang diambil hanya yang
    hasilnya 24
177     if ((result > 23.999999999) && (result < 24.000000001))
178     {
179         solution_counter++;
180         cout << solution_counter << " " << str << endl;
181     }
182 }
183
184 void bracket_permutation_4(int angka[], int i, int j, int k)
185 // n0 op0 ((n1 op1 n2) op2 n3)
186 // n adalah angka, dan op adalah operator
187 // Contoh: 6 + ((6 + 6) + 6)
188 {
189     str = "";
190     str = str + to_string(angka[0]);
191     str.append(op,i,1);
192     str.append("(");
193     str.append("(");
194     str = str + to_string(angka[1]);
195     str.append(op,j,1);
196     str = str + to_string(angka[2]);
197     str.append(")");
198     str.append(op,k,1);
199     str = str + to_string(angka[3]);
200     str.append(")");
201     expression_string = str.c_str();
202     result = expression();
203     // Evaluasikan permutasi ekspresi aritmatika yang ada, yang diambil hanya yang
    hasilnya 24
204     if ((result > 23.999999999) && (result < 24.000000001))
205     {
206         solution_counter++;
207         cout << solution_counter << " " << str << endl;
208     }
209 }
210
211 void bracket_permutation_5(int angka[], int i, int j, int k)
212 // n0 op1 (n1 op (n2 op3 n3))
213 // n adalah angka, dan op adalah operator
214 // Contoh: 6 + (6 + (6 + 6))
215 {
216     str = "";
217     str = str + to_string(angka[0]);
218     str.append(op,i,1);
219     str.append("(");
220     str = str + to_string(angka[1]);
221     str.append(op,j,1);
222     str.append("(");
223     str = str + to_string(angka[2]);
224     str.append(op,k,1);
225     str = str + to_string(angka[3]);
226     str.append(")");
227     str.append(")");
228     expression_string = str.c_str();
229     result = expression();
230     // Evaluasikan permutasi ekspresi aritmatika yang ada, yang diambil hanya yang
    hasilnya 24
231     if ((result > 23.999999999) && (result < 24.000000001))
232     {

```

```

233         solution_counter++;
234         cout << solution_counter << " " << str << endl;
235     }
236 }
237
238 //-----Enumerasikan Permutasi-----
239 bool mustSwap(int string[], int begin, int current_string_element)
240 // Jika elemen yang sedang diproses tidak sama dengan elemen-elemen setelah awal
    string, maka harus ditukar (return 1)
241 // Jika sama, tidak ditukar (return 0)
242 {
243     for (int x = begin; x < current_string_element; x++)
244     {
245         if (string[x] == string[current_string_element])
246         {
247             return 0;
248         }
249     }
250     return 1;
251 }
252
253 void print_permutations(int string[], int current_index, int length_of_string)
254 // Print seluruh permutasi string yang mungkin
255 {
256     if (current_index >= length_of_string)
257     {
258         for (int x = 0; x <= 3; x++)
259         {
260             for (int y = 0; y <= 3; y++)
261             {
262                 for (int z = 0; z <= 3; z++)
263                 {
264                     bracket_permutation_1(string, x, y, z);
265                     bracket_permutation_2(string, x, y, z);
266                     bracket_permutation_3(string, x, y, z);
267                     bracket_permutation_4(string, x, y, z);
268                     bracket_permutation_5(string, x, y, z);
269                 }
270             }
271         }
272         return;
273     }
274
275     for (int i = current_index; i < length_of_string; i++)
276     {
277         // Pengecekan setiap elemen string secara maju, ditentukan elemen harus ditukar
atau tidak
278         bool check = mustSwap(string, current_index, i);
279         if (check)
280         {
281             swap(string[current_index], string[i]);
282             print_permutations(string, current_index + 1, length_of_string);
283             swap(string[current_index], string[i]);
284         }
285     }
286 }
287
288 int main()
289 {
290     cout << "
    _____
    " << endl;

```

```

291     cout << " \\_____ \\ / | | / _____ / _____ / _____
| | _____" << endl;
292     cout << " / _____ // | | / \\ _____ \\ / \\ /
\\ \\ _____ \\ / - \\ | \\ \\ \\ / _____ \\ " << endl;
293     cout << "/ \\ / \\ / ^ / \\ \\ \\ / _____ Y
Y \\ _____ ( <_> ) |_\\ \\ / _____ / \\ /" << endl;
294     cout << "\\ _____ \\ _____ | \\ _____ ( _____ /_ |_ /\\ _____ >
/ _____ /\\ _____ /_ \\ / \\ / \\ _____ >_ | " << endl;
295     cout << " \\ / \\ / |_ | \\ / \\ / \\ / \\ / \\ / \\ /
\\ / " << endl;

296
297     //Input 4 angka
298     cout << "By Ignatius Timothy Manullang/13517044" << endl << "Input 4 numbers, one
at a time" << endl;
299     for(int number_counter = 0; number_counter <= 3; number_counter++)
300     {
301         cout << "Input number " << number_counter+1 << ": ";
302         cin >> number[number_counter];
303     }
304     auto start = chrono::high_resolution_clock::now(); // Set up waktu awal proses
305     print_permutations(number, 0, 4); // Cari permutasi dan tampilkan
306     //Tampilkan jumlah solusi 24 game yang dapat ditemukan dari input 4 angka
307     if (solution_counter == 0)
308     {
309         cout << "NO SOLUTIONS FOUND" << endl;
310     }
311     else
312     {
313         cout << "Solutions found: " << solution_counter << endl;
314     }
315     auto finish = chrono::high_resolution_clock::now(); // Set up waktu akhir proses
316     chrono::duration<double> elapsed = finish - start; // Hitung selisih waktu akhir
dan awal proses
317     cout << "Elapsed time: " << elapsed.count() << " s\\n"; // Tampilkan waktu yang
dipakai
318     return 0;
319 }

```

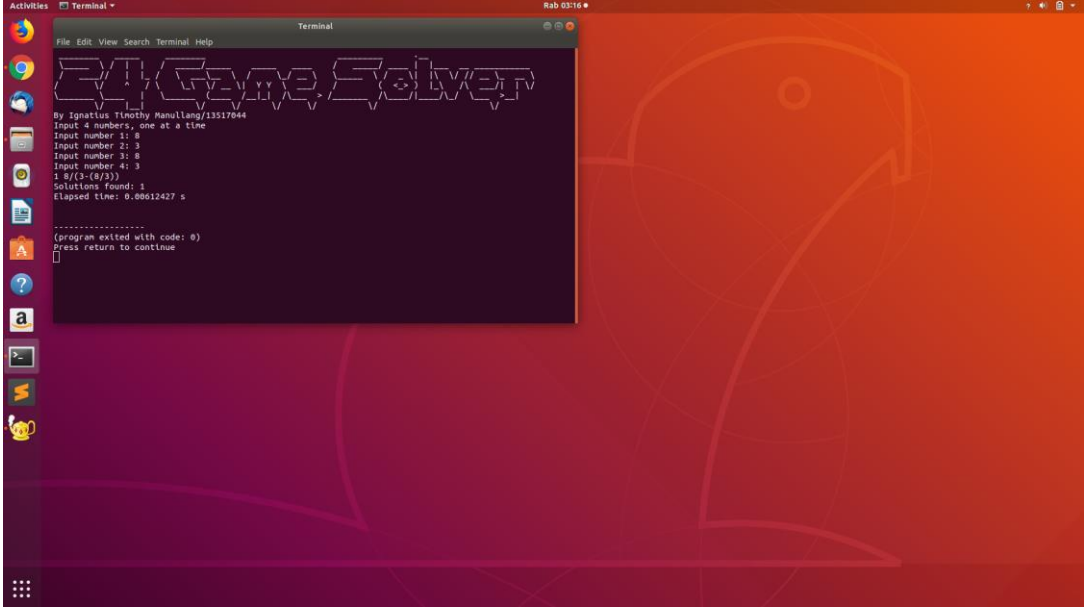


## BAB III

### EKSPERIMEN

Dalam bab ini akan terdapat gambar hasil solusi eksperimen, dan gambar solusi di 24 solver yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/> untuk masukan yang sama sebagai perbandingan.

1



Activities Terminal ▾ Bab 03/16 ●

File Edit View Search Terminal Help

24 Game Solver

By Ignatius Timothy Manullang/13517044

Input 4 numbers, one at a time

Input number 1: 8

Input number 2: 3

Input number 3: 8

Input number 4: 3

1 8/(3-(8/3))

Solutions found: 1


Elapsed time: 0.00612427 s

-----

(program exited with code: 0)

Press return to continue

Gambar 1.1 Aplikasi memberikan 1 solusi untuk masukan 8, 3, 8, 3, jumlah solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program



Menu 24 Solver ["8", "3", "8", "3"] X +

< > ↺ Not secure 24solver.us-west-2.elasticbeanstalk.com

AliExpress Booking.com Agoda.com Tokopedia

## Welcome to 24 Game Solver

Enter your 4 numbers below, then click "Solve" to see every solution that equals 24.

8

3

8

3

Solve

Clear

0 solutions found

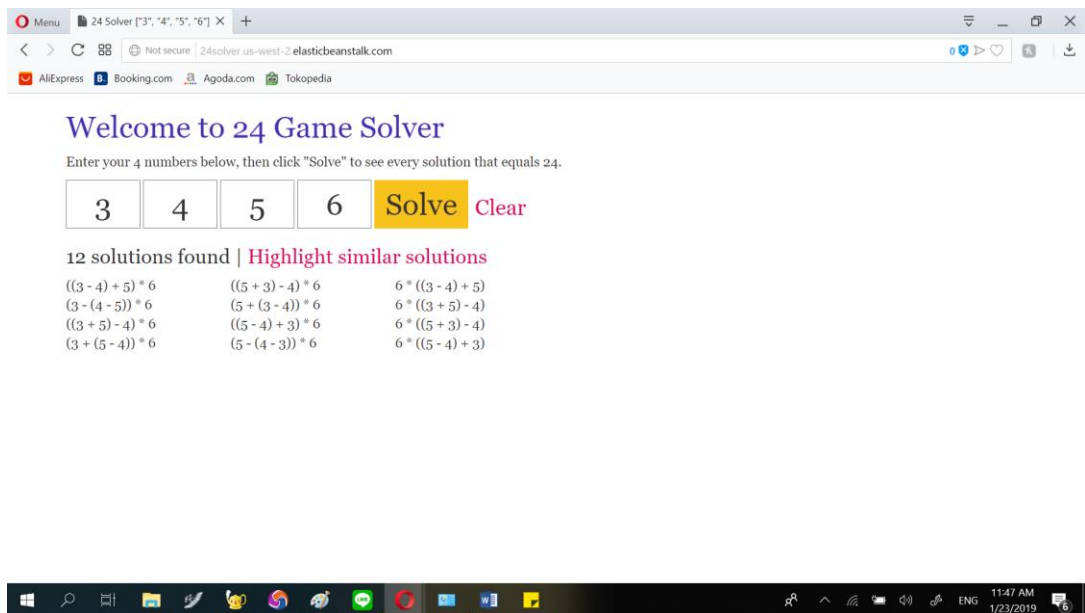
Windows taskbar: 11:51 AM 1/23/2019

Gambar 1.2 Solusi untuk masukan 8, 3, 8, 3 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

2

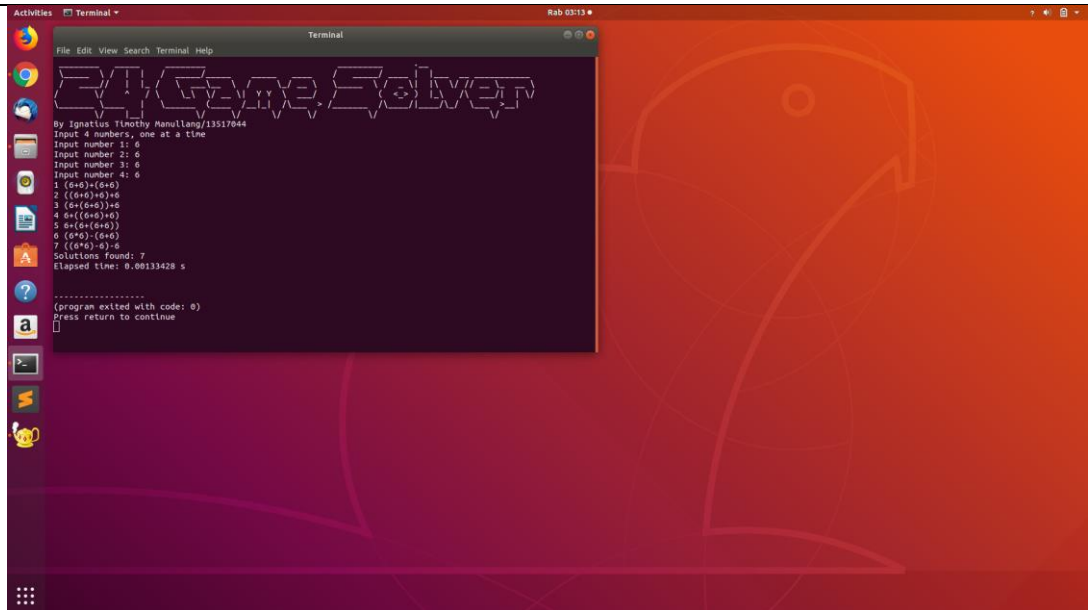
```
24 Game Solver
By Ignatius Timothy Manuilaing/13517044
Input 4 numbers, one at a time
Input number 1: 3
Input number 2: 4
Input number 3: 5
Input number 4: 6
1 ((3-4)+5)*6
2 ((3-4)+5)*6
3 ((3+5)-4)*6
4 ((3+5)-4)*6
5 ((5-3)+4)*6
6 ((5-3)+4)*6
7 ((5+3)-4)*6
8 ((5+3)-4)*6
9 6*((5-4)+3)
10 6*((5-4)+3)
11 6*((5+3)-4)
12 6*((5+3)-4)
13 6*((5+3)-4)
14 6*((5+3)-4)
15 6*((5+3)-4)
16 6*((5+3)-4)
Solutions found: 16
Elapsed time: 0.0130696 s
(program exited with code: 0)
Press return to continue
```

Gambar 2.1 Aplikasi memberikan 16 solusi untuk masukan 3, 4, 5, 6, jumlah solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program



Gambar 2.2 Solusi untuk masukan 3, 4, 5, 6 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

3

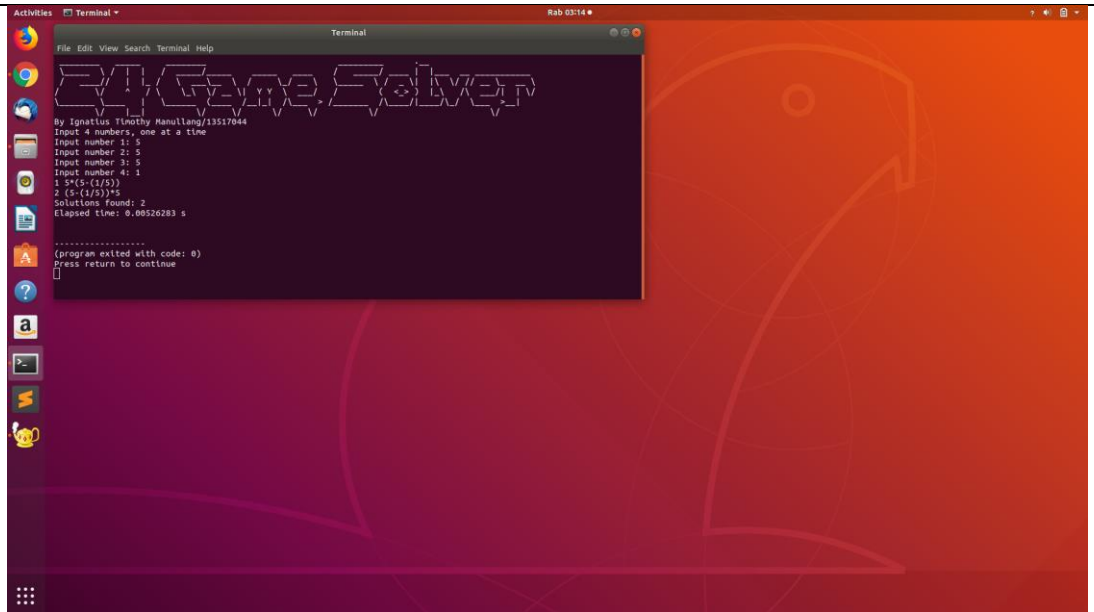


Gambar 3.1 Aplikasi memberikan 7 solusi untuk masukan 6, 6, 6, 6, jumlah solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program



Gambar 3.2 Solusi untuk masukan 6, 6, 6, 6 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

4

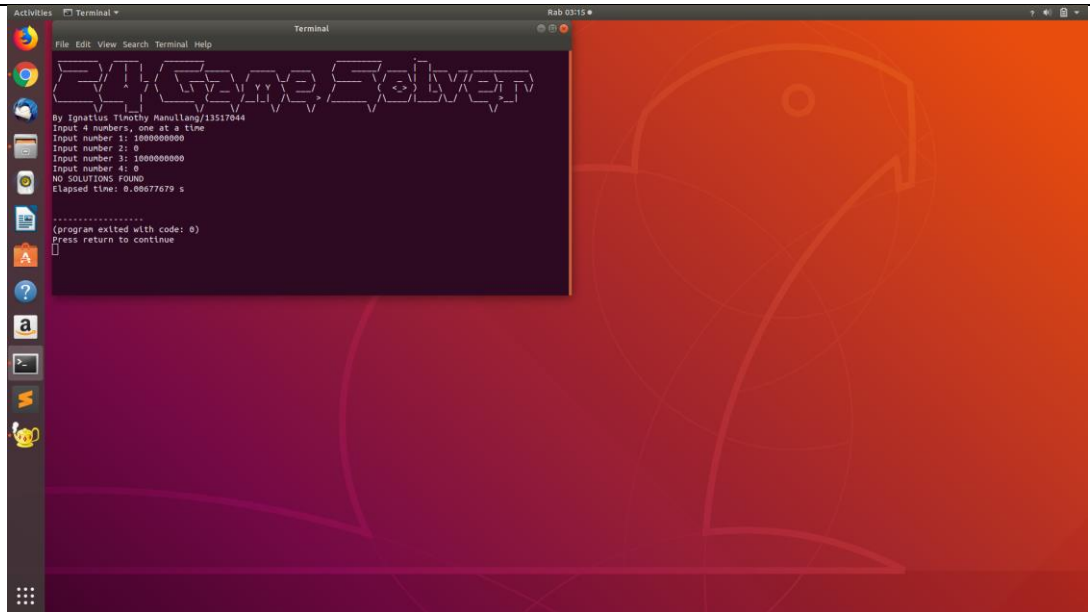


Gambar 4.1 Aplikasi memberikan 2 solusi untuk masukan 5, 5, 5, 1, jumlah solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program



Gambar 4.2 Solusi untuk masukan 5, 5, 5, 1 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

5



Gambar 5.1 Aplikasi tidak bisa memberikan solusi untuk masukan 1000000000, 0, 1000000000, 0, menampilkan pesan bahwa tidak ada solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program



Gambar 5.2 Solusi untuk masukan 1000000000, 0, 1000000000, 0 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

6

```

File Edit View Search Terminal Help

Euler Game Solver

By Ignatius Timothy Manullang/13517044

Input 4 numbers, one at a time
Input number 1: 1
Input number 2: 2
Input number 3: 3
Input number 4: 4

1 ((1+2)+3)*4
2 ((1+2)+3)*4
3 ((1+2)+3)*4
4 ((1+2)+3)*4
5 ((1+2)+3)*4
6 1*((2+3)+4)
7 1*((2+3)+4)
8 ((1+2)*(4+3))
9 ((1+2)*(4+3))
10 ((1+2)*(4+3))
11 1*((2+4)+3)
12 1*((2+4)+3)
13 ((1+3)+2)*4
14 ((1+3)+2)*4
15 ((1+3)+2)*4
16 ((1+3)+2)*4
17 ((1+3)+2)*4
18 ((1+3)+2)*4
19 1*((3+2)+4)
20 1*((3+2)+4)
21 ((1+3)*(4+2))
22 ((1+3)*(4+2))
23 ((1+3)*(4+2))
24 ((1+3)*(4+2))
25 1*((3+4)+2)
26 1*((3+4)+2)
27 ((1+4)*(3+2))
28 ((1+4)*(3+2))
29 ((1+4)*(3+2))
30 1*((4+3)+2)
31 1*((4+3)+2)
32 ((1+4)*(2+3))
33 ((1+4)*(2+3))
34 ((1+4)*(2+3))
35 1*((4+2)+3)
36 1*((4+2)+3)
37 ((2+1)+3)*4
38 ((2+1)+3)*4
39 ((2+1)+3)*4
40 ((2+1)+3)*4
41 ((2+1)+3)*4

```

Gambar 6.1 Aplikasi menampilkan solusi 1-41 untuk masukan 1, 2, 3, 4

```

File Edit View Search Terminal Help

42 2*((1+3)+4)
43 2*((1+3)+4)
44 ((2/1)+3)*4
45 ((2/1)+3)*4
46 ((2/1)+3)*4
47 ((2/1)+3)*4
48 2/((1/3)/4)
49 ((2+1)*(4+3))
50 ((2+1)*(4+3))
51 ((2+1)*(4+3))
52 2*((1+4)+3)
53 2*((1+4)+3)
54 ((2/1)*(4+3))
55 ((2/1)*(4+3))
56 ((2/1)*(4+3))
57 2/((1/4)+3)
58 2/((1/4)+3)
59 ((2+3)+1)*4
60 ((2+3)+1)*4
61 ((2+3)+1)*4
62 ((2+3)+1)*4
63 ((2+3)+1)*4
64 ((2+3)+1)*4
65 2*((3+1)+4)
66 ((2+3)/1)*4
67 ((2+3)/1)*4
68 2*((3/1)+4)
69 ((2+3)/(1/4))
70 2*((3/1/4))
71 ((2+3)*(4+1))
72 ((2+3)*(4+1))
73 ((2+3)*(4+1))
74 2*((3+4)+1)
75 2*((3+4)+1)
76 ((2+3)*(4/1))
77 ((2+3)*(4/1))
78 ((2+3)*(4/1))
79 2*((3+4)/1)
80 2*((3+4)/1)
81 ((2+4)*(3+1))
82 ((2+4)*(3+1))
83 ((2+4)*(3+1))
84 ((2+4)*(3+1))
85 2*((4+3)+1)
86 2*((4+3)+1)
87 ((2+4)*(3/1))
88 ((2+4)*(3/1))
89 ((2+4)*(3/1))
90 2*((4+3)/1)
91 2*((4+3)/1)
92 ((2+4)*(1+3))
93 ((2+4)*(1+3))
94 ((2+4)*(1+3))

```

Gambar 6.2 Aplikasi menampilkan solusi 42- 94 untuk masukan 1, 2, 3, 4

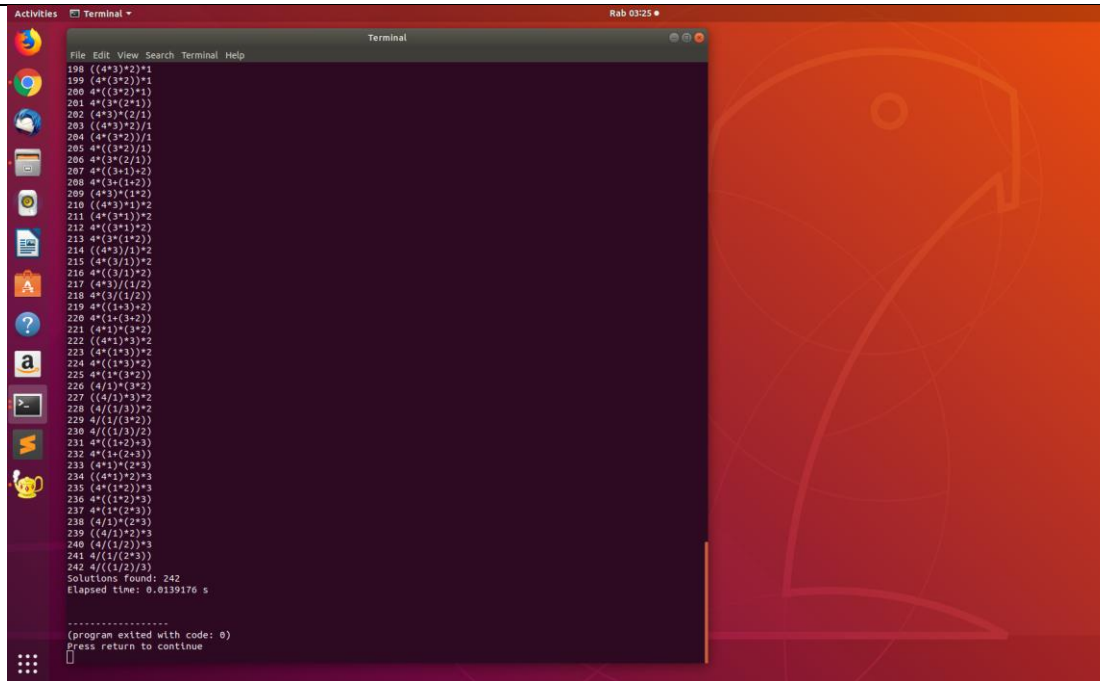
```
95 2*(4+1)*3
96 2*(4+1)*3
97 2*(4+1)*3
98 (2+4)/(1+3)
99 (2+4)/(1+3)
100 2*(4/1)*3
101 (2+4)/(1/3)
102 2*(4/(1/3))
103 ((3+2)+1)*4
104 (3+(2+1))*4
105 (3+2)*(1+4)
106 ((3+2)+1)*4
107 (3*(2+1))*4
108 3*((2+1)+4)
109 3*(2+(1+4))
110 ((3+2)/1)*4
111 (3*(2/1))*4
112 3*((2/1)+4)
113 (3*2)/(1/4)
114 3*(2/(1/4))
115 (3+2)*(4+1)
116 ((3+2)+4)*1
117 (3*(2+4))*1
118 3*(2+4)*1
119 3*(2+(4+1))
120 (3+2)*(4/1)
121 ((3+2)+4)/1
122 3*(2+4)/1
123 3*((2+4)/1)
124 3*(2*(4/1))
125 ((3+1)+2)*4
126 (3+(1+2))*4
127 (3+1)*(2+4)
128 (3+1)*(2+4)
129 ((3+1)+2)*4
130 (3*(1+2))*4
131 3*((1+2)+4)
132 3*(1+(2+4))
133 (3/1)*(2+4)
134 ((3/1)+2)*4
135 3/((1/2)+4)
136 3/(1/(2+4))
137 3/((1/2)/4)
138 (3+1)*(4+2)
139 (3+1)*(4+2)
140 ((3+1)+4)*2
141 (3*(1+4))*2
142 3*((1+4)+2)
143 3*(1+(4+2))
144 (3/1)*(4+2)
145 ((3/1)+4)*2
146 3/((1/4)+2)
147 3/(1/(4+2))
```

Gambar 6.3 Aplikasi menampilkan solusi 95-147 untuk masukan 1, 2, 3, 4

```
148 3/((1/4)/2)
149 (3+4)*(1+2)
150 ((3+4)+1)*2
151 (3*(4+1))*2
152 3*((4+1)+2)
153 3*(4+(1+2))
154 ((3+4)/1)*2
155 (3*(4/1))*2
156 3*((4/1)+2)
157 (3+4)/(1/2)
158 3*(4/(1/2))
159 (3+4)*(2+1)
160 ((3+4)+2)*1
161 (3*(4+2))*1
162 3*((4+2)+1)
163 3*(4+(2+1))
164 (3+4)*(2/1)
165 ((3+4)+2)/1
166 (3*(4+2))/1
167 3*((4+2)/1)
168 3*(4*(2/1))
169 (4+2)*(3+1)
170 4*((2+3)+1)
171 4*(2+(3+1))
172 (4+2)*(3+1)
173 ((4+2)+3)*1
174 (4*(2+3))*1
175 4*((2+3)+1)
176 4*(2*(3+1))
177 (4+2)*(3/1)
178 ((4+2)+3)/1
179 4*(2+3)/1
180 4*((2+3)/1)
181 4*(2*(3/1))
182 (4+2)*(1+3)
183 4*((2+1)+3)
184 4*(2+(1+3))
185 (4+2)*(1+3)
186 ((4+2)+1)*3
187 (4*(2+1))*3
188 4*((2+1)+3)
189 4*(2*(1+3))
190 ((4+2)/1)*3
191 (4*(2/1))*3
192 4*((2/1)+3)
193 (4+2)/(1/3)
194 4*(2/(1/3))
195 4*((3+2)+1)
196 4*(3+(2+1))
197 (4+3)*(2+1)
198 ((4+3)+2)*1
199 4*(3*(2+1))
200 4*((3+2)+1)
```

Gambar 6.4 Aplikasi menampilkan solusi 148-200 untuk masukan 1, 2, 3, 4





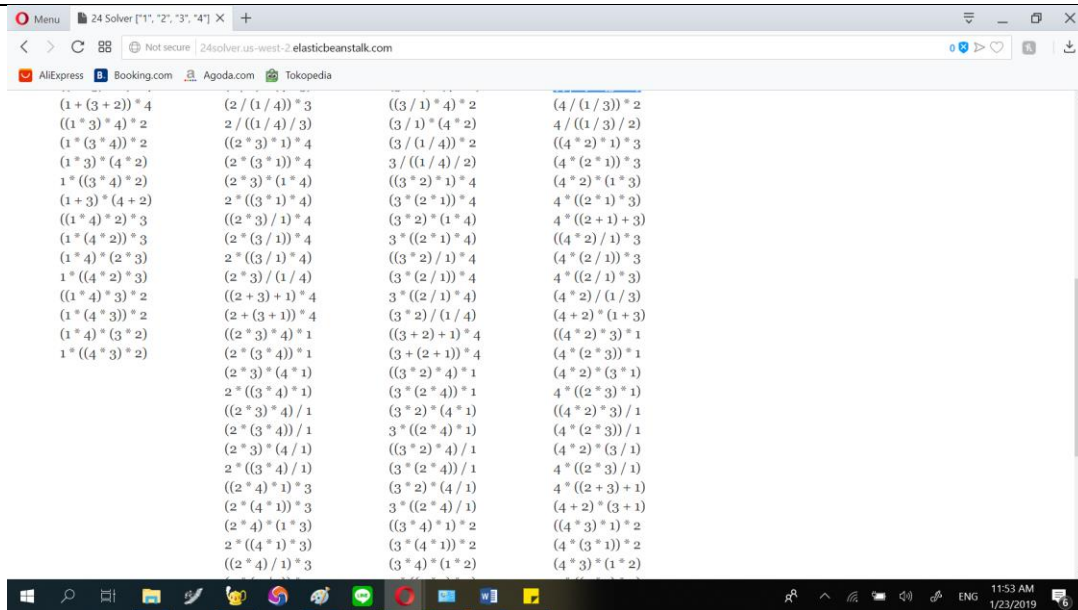
```
198 ((4+3)*2)+1
199 4*(3+2))+1
200 4*((3+2)+1)
201 4*(3*(2+1))
202 (4+3)*(2/1)
203 ((4+3)*2)/1
204 4*(3+2)/1
205 4*((3+2)/1)
206 4*(3*(2/1))
207 4*((3+1)+2)
208 4*(3+(1+2))
209 (4+3)*(1+2)
210 ((4+3)+1)*2
211 4*(3+1))*2
212 4*((3+1)+2)
213 4*(3*(1+2))
214 ((4+3)/1)*2
215 4*((3/1)*2)
216 4*((3/1)+2)
217 (4+3)/(1/2)
218 4*(3/(1/2))
219 4*((3+3)+2)
220 4*(1+(3+2))
221 4*(1*(3+2))
222 ((4+3)+2)*2
223 4*(1+3))*2
224 4*((1+3)+2)
225 4*((1+3)/2)
226 (4/1)*(3+2)
227 ((4/1)+3)*2
228 4/(1/3))*2
229 4/(1/(3+2))
230 4/((1/3)/2)
231 4*((1+2)+3)
232 4*(1+(2+3))
233 4*(1*(2+3))
234 ((4+1)+2)*3
235 4*((1+2)+3)
236 4*((1+2)+3)
237 4*(1*(2+3))
238 (4/1)*(2+3)
239 ((4/1)+2)*2
240 4/(1/2))*3
241 4/(1/(2+3))
242 4/(1/(2/3))
Solutions found: 242
Elapsed time: 0.0139176 s
(program exited with code: 0)
Press return to continue
```

Gambar 6.5 Aplikasi menampilkan solusi 198-247 untuk masukan 1, 2, 3, 4, jumlah solusi yang ditemukan, dan waktu yang digunakan untuk mengeksekusi program

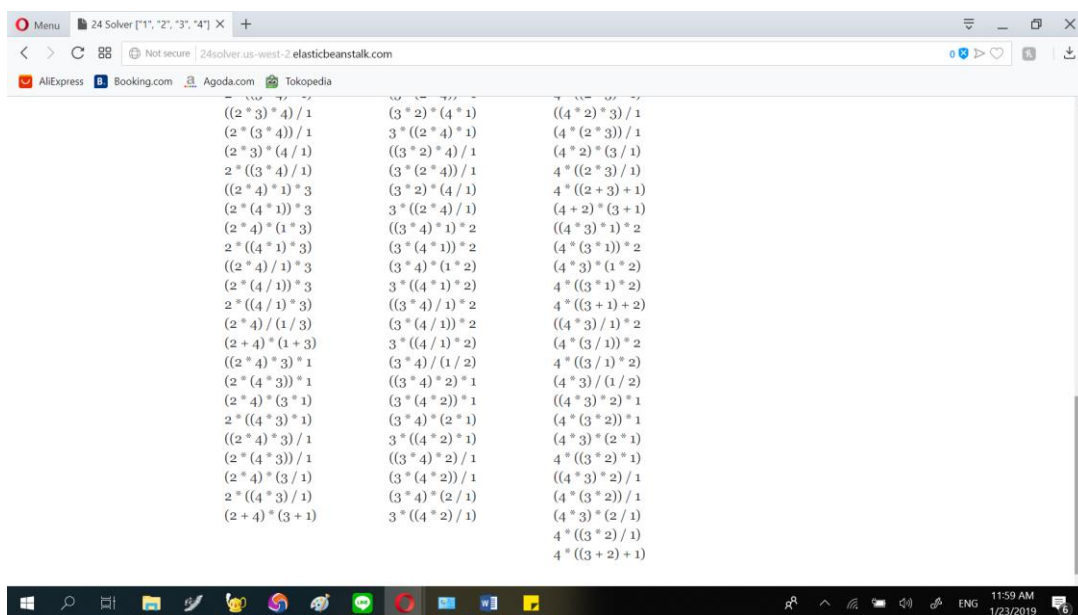


Gambar 6.6 Solusi 1-64 untuk masukan 1, 2, 3, 4 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>





Gambar 6.7 Solusi 65-153 untuk masukan 1, 2, 3, 4 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>



Gambar 6.8 Solusi 127-194 untuk masukan 1, 2, 3, 4 yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/>

## Keterangan Tambahan

Aplikasi dijalankan di Asus Zenbook Flip 14 UX461UN dengan processor Intel® Core™ i7-8550 CPU @ 1.80 GHz 1.99 GHz, dengan display device Intel® UHD Graphics 620 dan render device NVIDIA GeForce MX150, menggunakan operating system Ubuntu 18.04 LTS.

24 Game Solver yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/> dijalankan di Asus Zenbook Flip 14 UX461UN dengan processor Intel® Core™ i7-8550 CPU @ 1.80 GHz 1.99 GHz, dengan display device Intel® UHD Graphics 620 dan render device NVIDIA GeForce MX150, menggunakan operating system Windows 10 dan browser Opera.

Asumsi permutasi grouping tanda kurung yang mungkin berisi 2 angka atau ekspresi aritmatika dalam kurung, dan 1 operator, yaitu

6.  $(n_0 \text{ op}_0 n_1) \text{ op}_1 (n_2 \text{ op}_2 n_3)$
7.  $((n_0 \text{ op}_1 n_1) \text{ op}_2 n_2) \text{ op}_3 n_3$
8.  $(n_0 \text{ op}_0 (n_1 \text{ op}_1 n_2)) \text{ op}_2 n_3$
9.  $n_0 \text{ op}_0 ((n_1 \text{ op}_1 n_2) \text{ op}_2 n_3)$
10.  $n_0 \text{ op}_1 (n_1 \text{ op} (n_2 \text{ op}_3 n_3))$

dengan  $n$  adalah angka, dan  $\text{op}$  adalah operator, , dan 0-3 adalah urutan dalam ekspresi aritmatika.

Hal tersebut berbeda dengan 24 solver yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/> yang memiliki permutasi grouping tanda kurung hanya:

1.  $(n_0 \text{ op}_0 n_1) \text{ op}_1 (n_2 \text{ op}_2 n_3)$
2.  $((n_0 \text{ op}_1 n_1) \text{ op}_2 n_2) \text{ op}_3 n_3$
3.  $(n_0 \text{ op}_0 (n_1 \text{ op}_1 n_2)) \text{ op}_2 n_3$
4.  $n_0 \text{ op}_0 ((n_1 \text{ op}_1 n_2) \text{ op}_2 n_3)$

Selain itu, di dalam beberapa ekspresi aritmatika yang dibentuk, terdapat pecahan, namun dalam perhitungan dengan komputer diubah menjadi decimal sehingga terdapat kehilangan keakuratan. Oleh karena itu, hasil yang seharusnya dicek dengan 24, dicek dengan range antara 23.9999999999 dan 24.0000000001. Salah satu contoh dari ini adalah pada eksperimen 1 (Gambar 1),

dengan input 8 3 8 3. Hal tersebut berbeda dengan 24 solver yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/> yang tidak memperhitungkan hal tersebut, contohnya dengan input 8 3 8 3.

Kedua hal tersebut menyebabkan solusi yang ada di <http://24solver.us-west-2.elasticbeanstalk.com/> lebih sedikit dibandingkan yang terdapat di aplikasi.

## **BAB IV**

### **REFERENSI**

#### **Sumber Tertulis:**

1. Munir, R. (2014). Algoritma Brute Force. Diakses Januari 23, 2019, dari [http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Brute-Force-\(2016\).pdf](http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Brute-Force-(2016).pdf)
2. Welcome to 24 Game Solver. (n.d.). Retrieved January 23, 2019, dari <http://24solver.us-west-2.elasticbeanstalk.com/>

Poin	Ya	Tidak
1. Program berhasil dikompilasi	✓	
2. Program berhasil dieksekusi	✓	
3. Solusi 24 Game benar untuk semua data tes. Bandingkanlah hasilnya dengan aplikasi 24solver yang sudah ada.	✓	