

Tugas Kecil IF2211 Strategi Algoritma

**Laporan Penyelesaian Permainan Kartu 24 dengan
Algoritma *Brute Force***

Oleh:

Muhammad Bangkit Dwi Cahyono

K01 / 13521055



PROGRAM STUDIO TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

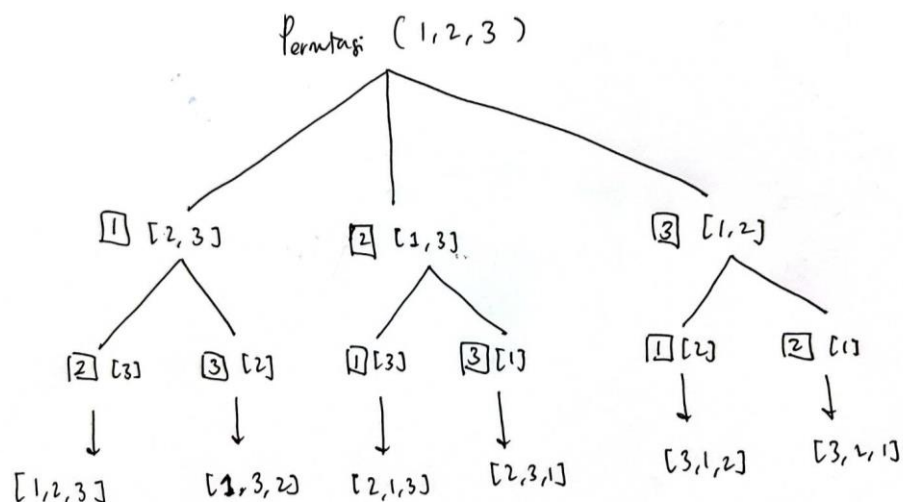
2023

A. Algoritma *Brute Force*

Algoritma *brute force* adalah sebuah pendekatan yang mudah untuk memecahkan suatu masalah, biasanya didasarkan langsung pada pernyataan masalah dan definisi konsep yang dilibatkan. Algoritma ini memecahkan masalah dengan sederhana, langsung (*straightforward*), dan dengan cara yang jelas. Algoritma ini menyelesaikan masalah dengan cara meninjau semua kasus yang muncul, lalu memilih kasus-kasus mana yang diinginkan program. Meskipun menyelesaikan suatu persoalan, tetap ada banyak cara yang berbeda dalam menyelesaikannya. Keragaman ini disebabkan pendekatan yang berbeda dalam mengerjakan persoalan tersebut, beberapa observasi yang dapat mengoptimalkan, dan membuang kasus yang tidak perlu.

Dalam menyelesaikan tugas kecil penyelesaian kartu 24 yang diberikan, digunakan algoritma *brute force*. Berikut langkah-langkah algoritma yang dibuat.

1. Pertama tentunya program akan membaca input dari pengguna untuk memilih apakah pengguna ingin memasukkan 4 kartu bebas atau 4 kartu bebas dipilih acak oleh program. Tentunya 4 kartu ini terdiri dari A, 2, 3, ..., J, Q, K, seperti halnya kartu remi. Kartu inilah yang nantinya dihitung semua kemungkinannya dengan menggunakan 4 operator yang ada (+, -, *, /) yang menghasilkan nilai 24. Tentunya input pengguna divalidasi oleh program.
2. Kemudian setelah 4 kartu sudah terpilih, saya melakukan permutasi kepada 4 angka tersebut. Algoritma yang digunakan adalah permutasi dengan rekursif sebagai berikut.



Mula-mula kita memiliki array yang berisi 4 angka, lalu kita bagi menjadi bagian-bagian kecil seperti halnya pada gambar di atas (dicontohkan 3 angka pada gambar). Diambil masing-masing elemen pada list dan sisa list yang tidak memiliki elemen tersebut. Lalu sisa list tadi direkursif ke fungsi awal, hasil dari rekursif itu kita *concat* ke elemen yang tadi di awal sudah kita pisah. Dengan demikian, didapatkan semua kemungkinan atau kombinasi dari 4 angka awal.

3. Setelah mendapatkan semua kemungkinan tadi, saatnya kita mengevaluasi dari kemungkinan yang ada dari tiap 4 angka tadi dengan menggunakan operator. Kita memiliki 4 angka, maka terdapat 3 kombinasi operator yang bisa kita gunakan dan ada 5 kemungkinan tanda kurung “(“ dan “)” pada setiap hasil evaluasinya.
 - a. (N OP N) OP (N OP N)
 - b. ((N OP N) OP N) OP N
 - c. (N OP (N OP N)) OP N
 - d. N OP (N OP (N OP N))
 - e. N OP ((N OP N) OP N)

Dengan N adalah angka dan OP adalah operator. Algoritma yang dilakukan yaitu looping sebanyak $4 \times 4 \times 4$ (karena ada kemungkinan 4 operator dalam 3 tempat), lalu diloop ke semua kombinasi hasil permutasi awal, yakni 24 disebabkan $4! = 24$ kemungkinan. Dicoba semua kemungkinan tadi dengan 5 kasus tadi dengan “if” sebanyak 5 kali, lalu didapatkan hasil dari algoritma *brute force* tersebut.

4. Dikarenakan ada kemungkinan bahwa 4 kartu yang diinput di awal sama, misal “2 2 3 4” maka kita harus menghilangkan evaluasi dengan hasil yang sama, misal “3 * ((2 * 2) + 4)” dengan “3 * ((2 * 2) + 4)”, karena solusi tersebut sama, dibedakan dengan penempatan angka 2 saja, maka hal tersebut dianggap 1 solusi saja. Untuk itu, digunakanlah struktur data *set* agar tidak ada data yang terduplikasi.
5. Setelah semua solusi didapat, ditampilkan banyaknya solusi dan tentunya solusi itu sendiri ke layar, dan juga waktu eksekusinya.
6. Setelah semua selesai, pengguna dapat men-*save* filenya jika ingin ke dalam filename.txt.

B. Source Program

```

1  #include <iostream>
2  #include <float.h>
3
4  /* ADT and String Manipulation */
5  #include <sstream>
6  #include <string>
7  #include <vector>
8  #include <set>
9
10 /* Files */
11 #include <fstream>
12
13 /* UI and Time */
14 #include <ctime>
15 #include <windows.h>
16 #include <bits/stdc++.h>
17
18 using namespace std;
19 #define RESET "\033[0m"
20 #define BLACK "\033[1m\033[30m" /* Black */
21 #define RED "\033[1m\033[31m" /* Red */
22 #define GREEN "\033[1m\033[32m" /* Green */
23 #define WHITE "\033[1m\033[37m" /* White */
24
25 int countWords(string str) {
26     /* Menghitung banyaknya kata dalam 1 line of string (input user) */
27
28     bool space = true;
29     int count = 0;
30
31     for (auto c:str) {
32         if (isspace(c)) {
33             space = true;
34         } else {
35             if (space) {
36                 ++count;
37                 space = false;
38             }
39         }
40     }
41     return count;
42 }
43

```

```

44 bool isValid(string str) {
45     /* Mengecek input user agar masukan harus [A, 2, ..., Q, K] */
46
47     if (str.length() == 1) {
48         /* Handle 1 char */
49         if (isdigit(str[0])) {
50             if (str[0] != 'A' && str[0] != 'J' && str[0] != 'Q' && str[0] != 'K') {
51                 return false;
52             }
53         } else {
54             if (str[0] == '0' or str[0] == '1') {
55                 return false;
56             }
57         }
58     } else if (str.length() == 2) {
59         /* Handle 2 char (10) */
60         if (str[0] != '1' || str[1] != '0') {
61             return false;
62         }
63     } else {
64         return false;
65     }
66
67     return true;
68 }
69
70 bool isValid2(string s) {
71     /* Mengecek input user agar masukan terdiri dari 4 masukan */
72
73     if (countWords(s) != 4) {
74         return false;
75     }
76
77     vector<string> inputSplitted;
78     string input2;
79     stringstream ss(s);
80
81     while (getline(ss, input2, ' ')) {
82         inputSplitted.push_back(input2);
83     }
84

```

```

85     if (isValid(input_splitted[0]) && isValid(input_splitted[1]) && isValid(input_splitted[2]) && isValid(input_splitted[3])) {
86         return true;
87     } else {
88         return false;
89     }
90 }
91
92 int strToNum(string str) {
93     /* Mengubah string (masukan user) menjadi number */
94     /* str pasti A, J, Q, K, atau angka 2 - 10 (sudah divalidasi) */
95
96     if (str[0] == 'A') {
97         return 1;
98     } else if (str[0] == 'J') {
99         return 11;
100    } else if (str[0] == 'Q') {
101        return 12;
102    } else if (str[0] == 'K') {
103        return 13;
104    } else {
105        if (str.length() == 2 && str[0] == '1' && str[1] == '0') {
106            return 10;
107        }
108        return str[0] - '0';
109    }
110 }
111
112 string numToString(int c) {
113     /* Mengubah integer to string */
114
115     switch(c) {
116         case 1:
117             return "A";
118         case 11:
119             return "J";
120         case 12:
121             return "Q";
122         case 13:
123             return "K";
124         default:
125             return to_string(c);
126     }
127 }

```

```

129 double func(double a, char op, double b) {
130     /* Menghitung operasi 2 angka dengan menggunakan 4 macam operator */
131
132     switch(op) {
133         case '+':
134             return a + b;
135         case '-':
136             return a - b;
137         case '*':
138             return a * b;
139         default:
140             if (b != 0) {
141                 return a / b;
142             } else {
143                 return -9999;
144             }
145     }
146 }
147
148 vector<vector<int>> permute(vector<int>& nums) {
149     /* Menghasilkan vector of vector yang berisi hasil permutasi */
150
151     if (nums.size() <= 1) { // basis
152         return {nums};
153     }
154
155     vector<vector<int>> result;
156     for (int i = 0; i < nums.size(); i++) {
157         vector<int> v(nums.begin(), nums.end());
158         v.erase(v.begin() + i);
159         auto res = permute(v); // rekurens
160
161         for (int j = 0; j < res.size(); j++) {
162             vector<int> _v = res[j];
163             _v.insert(_v.begin(), nums[i]);
164
165             result.push_back(_v);
166         }
167     }
168
169     return result;
170 }

```

```

172 int opToInt(char op) {
173     /* Mengubah operator menjadi integer */
174
175     switch(op) {
176         case '+':
177             return 0;
178         case '-':
179             return 1;
180         case '*':
181             return 2;
182         default:
183             return 3;
184     }
185 }
186
187 double absolute(double x) {
188     /* Menghitung absolute suatu angka */
189     /* Handle error */
190
191     if (x < 0) {
192         return -x;
193     } else {
194         return x;
195     }
196 }
197
198 vector<int> solver1() {
199     /* Solver1 : user memasukkan input */
200
201     string input, input2;
202     vector<int> numbers;
203
204     cout << "Enter 4 cards: ";
205     getline(cin, input);
206
207     while (!isValid2(input)) {
208         cout << endl << RED << "Please enter 4 valid value [A, 2, ..., Q, K] !!!" << RESET << endl;
209         cout << "Enter 4 cards: ";
210         getline(cin, input);
211     }
212

```

```

213     stringstream ss(input);
214
215     while (getline(ss, input2, ' ')) {
216         numbers.push_back(strToInt(input2));
217     }
218
219     return numbers;
220 }
221
222 vector<int> solver2() {
223     /* Solver2 : random generated number */
224     /* Menghasilkan vektor dengan 4 random integer */
225
226     srand(time(0)); // Initialize random number generator
227     vector<int> random;
228
229     for (int i = 0; i < 4; i++) {
230         random.push_back((rand() % 13) + 1); // 1 - 13
231     }
232
233     return random;
234 }
235
236 int inputOneTwo() {
237     int n;
238
239     while (true) {
240         cout << "> ";
241         if (cin >> n && (n == 1 || n == 2)) {
242             break;
243         } else {
244             cout << RED << "Please enter a valid integer!! [1]/[2]" << RESET << endl;
245             cin.clear();
246         }
247     }
248     cin.ignore(80, '\n');
249
250     return n;
251 }
252
253 void splash() {
254     /* SPLASH SCREEN */
255

```



```

423     case 0:
424         MyFile << "(" << numToString(x[1]) << " " << operators[x[2]] << " " << numToString(x[3]) << ")" << operators[x[4]] << "(" << numToSt
425         break;
426     case 1:
427         MyFile << "(" << numToString(x[1]) << " " << operators[x[2]] << " " << numToString(x[3]) << ")" << operators[x[4]] << " " << numToSt
428         break;
429     case 2:
430         MyFile << "(" << numToString(x[1]) << " " << operators[x[2]] << " (" << numToString(x[3]) << " " << operators[x[4]] << " " << numToSt
431         break;
432     case 3:
433         MyFile << numToString(x[1]) << " " << operators[x[2]] << " " << "(" << numToString(x[3]) << " " << operators[x[4]] << "(" << numToSt
434         break;
435     case 4:
436         MyFile << numToString(x[1]) << " " << operators[x[2]] << " (" << numToString(x[3]) << " " << operators[x[4]] << " " << numToString(x[3]
437         break;
438     }
439 }
440
441 // close file
442 MyFile.close();
443
444 cout << GREEN << "File saved succesfully" << RESET << endl;
445 }
446 }
447 }
448
449 int main () {
450     /* Main Program */
451
452     int startInput;
453
454     splash();
455     cout << "Welcome to 24 Cards Solver >" << endl;
456     command1();
457
458     startInput = inputOneTwo();
459
460     while (startInput == 1) {
461         app();
462
463         cout << endl << GREEN << "Do you want to try again?" << RESET << endl << endl;
464         command1();
465         startInput = inputOneTwo();
466     }
467
468     cout << endl << GREEN << "Thank You >" << RESET << endl;
469     exit(0);
470
471     return 0;
472 }

```

1. Testcase 1 (A J Q K)

[illegible]

32 Solutions Found
(A * Q) * (K - J)
(Q * A) * (K - J)
(Q / A) * (A * Q)
(K - J) * (A * Q)
(K - J) * (Q * A)
(K - J) * (Q / A)
(K - J) / (A / Q)
((A * K) - J) * Q
((K - J) * A) * Q
((K - J) * Q) * A
((K - J) * Q) / A
((K - J) / A) * Q
((K - J) - J) * Q
((K / A) - J) * Q
(A * (K - J)) * Q
Q * (K - J)) * A
Q * (K - J)) / A
Q - (A * J)) * Q
Q - (J * A)) * Q
(K - (J / A)) * Q
A * (Q * (K - J))
Q * (A * (K - J))
Q * (K - (A * J))
Q * (K - (J * A))
Q / (A / (K - J))
A * (K - J) * Q
Q * (A * K) - J
Q * (K - J) / A
Q * (K * A) - J
Q * (K / A) - J

(Hasil tetscase1.txt)

2. Testcase 2 (7 Q 6 K)

```
=====
=                               List Command                               =
=====
[1] Input 4 Cards [A, 2, ..., Q, K]
[2] Generate 4 Random Cards
> 1
Enter 4 cards: 7 Q 6 K

92 Solutions Found
(6 + Q) + (K - 7)
(6 + Q) - (7 - K)
(6 + K) + (Q - 7)
(6 + K) - (7 - Q)
(6 - 7) + (Q + K)
(6 - 7) + (K + Q)
(Q + 6) + (K - 7)
(Q + 6) - (7 - K)
(Q + K) + (6 - 7)
(Q + K) - (7 - 6)
(Q - 7) + (6 + K)
(Q - 7) + (K + 6)
(K + 6) + (Q - 7)
(K + 6) - (7 - Q)
(K + Q) + (6 - 7)
(K + Q) - (7 - 6)
(K - 7) + (6 + Q)
(K - 7) + (Q + 6)
((6 + Q) + K) - 7
((6 + Q) - 7) + K
((6 + K) + Q) - 7
((6 + K) - 7) + Q
((6 - 7) + Q) + K
((6 - 7) + K) + Q
((Q + 6) + K) - 7
((Q + 6) - 7) + K
((Q + K) + 6) - 7
((Q + K) - 7) + 6
((Q - 7) + 6) + K
((Q - 7) + K) + 6
((K + 6) + Q) - 7
((K + 6) - 7) + Q
((K + Q) + 6) - 7
((K + Q) - 7) + 6
((K - 7) + 6) + Q
((K - 7) + Q) + 6
((K - 7) * 6) - 7
(6 + (Q + K)) - 7
(6 + (Q - 7)) + K
(6 + (K + Q)) - 7
(6 + (K - 7)) + Q
(6 - (7 - Q)) + K
(6 - (7 - K)) + Q
(6 * (K - 7)) - Q
(Q + (6 + K)) - 7
(Q + (6 - 7)) + K
(Q + (K + 6)) - 7
(Q + (K - 7)) + 6
(Q - (7 - 6)) + K
(Q - (7 - K)) + 6
(Q + (K - 7)) - 7
Q + (K - (7 - 6))
Q + (K - (7 - 6))
Q - (7 - (6 + K))
Q - (7 - (K + 6))
K + (6 + (Q - 7))
K + (6 - (7 - 6))
K + ((Q + 6) - 7)
K + ((Q - 7) + 6)
K - ((7 - 6) - Q)
K - ((7 - Q) - 6)
Execution Time: 0.001 seconds
```

```

=====
=          Do you want to save file?          =
=====
[1] Yes
[2] No
> 1

Input Filename: testcase2
File saved succesfully

```

<pre> 92 Solutions Found (6 + Q) + (K - 7) (6 + Q) - (7 - K) (6 + K) + (Q - 7) (6 + K) - (7 - Q) (6 - 7) + (Q + K) (6 - 7) + (K + Q) (Q + 6) + (K - 7) (Q + 6) - (7 - K) (Q + K) + (6 - 7) (Q + K) - (7 - 6) (Q - 7) + (6 + K) (Q - 7) + (K + 6) (K + 6) + (Q - 7) (K + 6) - (7 - Q) (K + Q) + (6 - 7) (K + Q) - (7 - 6) (K - 7) + (6 + Q) (K - 7) + (Q + 6) ((6 + Q) + K) - 7 ((6 + Q) - 7) + K ((6 + K) + Q) - 7 ((6 + K) - 7) + Q ((6 - 7) + Q) + K ((6 - 7) + K) + Q ((Q + 6) + K) - 7 ((Q + 6) - 7) + K ((Q + K) + 6) - 7 ((Q + K) - 7) + 6 ((Q - 7) + 6) + K ((Q - 7) + K) + 6 ((K + 6) + Q) - 7 ((K + 6) - 7) + Q </pre>	<pre> ((K + Q) + 6) - 7 ((K + Q) - 7) + 6 ((K - 7) + 6) + Q ((K - 7) + Q) + 6 ((K - 7) * 6) - Q (6 + (Q + K)) - 7 (6 + (Q - 7)) + K (6 + (K + Q)) - 7 (6 + (K - 7)) + Q (6 - (7 - Q)) + K (6 - (7 - K)) + Q (6 * (K - 7)) - Q (Q + (6 + K)) - 7 (Q + (6 - 7)) + K (Q + (K + 6)) - 7 (Q + (K - 7)) + 6 (Q - (7 - 6)) + K (Q - (7 - K)) + 6 (K + (6 + Q)) - 7 (K + (6 - 7)) + Q (K + (Q + 6)) - 7 (K + (Q - 7)) + 6 (K - (7 - 6)) + Q (K - (7 - Q)) + 6 6 + (Q + (K - 7)) 6 + (Q - (7 - K)) 6 + (K + (Q - 7)) 6 + (K - (7 - Q)) 6 - (7 - (Q + K)) 6 - (7 - (K + Q)) Q + (6 + (K - 7)) Q + (6 - (7 - K)) Q + (K + (6 - 7)) </pre>	<pre> Q + (K - (7 - 6)) Q - (7 - (6 + K)) Q - (7 - (K + 6)) K + (6 + (Q - 7)) K + (6 - (7 - Q)) K + (Q + (6 - 7)) K + (Q - (7 - 6)) K - (7 - (6 + Q)) K - (7 - (Q + 6)) 6 + ((Q + K) - 7) 6 + ((Q - 7) + K) 6 + ((K + Q) - 7) 6 + ((K - 7) + Q) 6 - ((7 - Q) - K) 6 - ((7 - K) - Q) Q + ((6 + K) - 7) Q + ((6 - 7) + K) Q + ((K + 6) - 7) Q + ((K - 7) + 6) Q - ((7 - 6) - K) Q - ((7 - K) - 6) K + ((6 + Q) - 7) K + ((6 - 7) + Q) K + ((Q + 6) - 7) K + ((Q - 7) + 6) K - ((7 - 6) - Q) K - ((7 - Q) - 6) </pre>
---	--	--

(Hasil testcase2.txt)

3. Testcase 3 (A A A A)

```

=====
=          List Command          =
=====
[1] Input 4 Cards [A, 2, ..., Q, K]
[2] Generate 4 Random Cards
> 1
Enter 4 cards: A A A A

No Solutions Found
Execution Time: 0.001 seconds

```

4. Testcase 4 (Random) (2 2 5 A)

```
[2] Generate 4 Random Cards
> 2

Your cards are: 2 2 5 A

16 Solutions Found
(A + 5) * (2 + 2)
(A + 5) * (2 * 2)
(2 + 2) * (A + 5)
(2 + 2) * (5 + A)
(2 * 2) * (A + 5)
(2 * 2) * (5 + A)
(5 + A) * (2 + 2)
(5 + A) * (2 * 2)
((A + 5) * 2) * 2
((5 + A) * 2) * 2
(2 * (A + 5)) * 2
(2 * (5 + A)) * 2
2 * (2 * (A + 5))
2 * (2 * (5 + A))
2 * ((A + 5) * 2)
2 * ((5 + A) * 2)
Execution Time: 0.001 seconds
```

```
=====
=          Do you want to save file?          =
=====
[1] Yes
[2] No
> 1

Input Filename: testcase4
File saved succesfully
```

```
16 Solutions Found
(A + 5) * (2 + 2)
(A + 5) * (2 * 2)
(2 + 2) * (A + 5)
(2 + 2) * (5 + A)
(2 * 2) * (A + 5)
(2 * 2) * (5 + A)
(5 + A) * (2 + 2)
(5 + A) * (2 * 2)
((A + 5) * 2) * 2
((5 + A) * 2) * 2
(2 * (A + 5)) * 2
(2 * (5 + A)) * 2
2 * (2 * (A + 5))
2 * (2 * (5 + A))
2 * ((A + 5) * 2)
2 * ((5 + A) * 2)
```

(Hasil testcase4.txt)

5. Testcase 5 (Random) (10 10 K 7)

```
=====
=          List Command          =
=====
[1] Input 4 Cards [A, 2, ..., Q, K]
[2] Generate 4 Random Cards
> 2

Your cards are: 10 10 K 7

No Solutions Found
Execution Time: 0 seconds
```

(Terkadang execution time 0 seconds)

6. Testcase 6 (Random) (Q A 7 5)

```
[1] Input 4 Cards [A, 2, ..., Q, K]
[2] Generate 4 Random Cards
> 2
```

Your cards are: Q A 7 5

230 Solutions Found

```
(A - Q) + (5 * 7)
(A - Q) + (7 * 5)
(A * 5) + (7 + Q)
(A * 5) + (Q + 7)
(A * 7) + (5 + Q)
(A * 7) + (Q + 5)
(A * Q) + (5 + 7)
(A * Q) + (7 + 5)
(A * Q) * (7 - 5)
(5 + 7) + (A * Q)
(5 + 7) + (Q * A)
(5 + 7) + (Q / A)
(5 + Q) + (A * 7)
(5 + Q) + (7 * A)
(5 + Q) + (7 / A)
(5 * A) + (7 + Q)
(5 * A) + (Q + 7)
(5 * 7) + (A - Q)
(5 * 7) - (Q - A)
(5 / A) + (7 + Q)
(5 / A) + (Q + 7)
(7 + 5) + (A * Q)
```

```
((7 + 5) + (Q * A))
((7 + 5) + (Q / A))
((7 + Q) + (A * 5))
((7 + Q) + (5 * A))
((7 + Q) + (5 / A))
((7 - 5) * (A * Q))
((7 - 5) * (Q * A))
((7 - 5) * (Q / A))
((7 - 5) / (A / Q))
((7 * A) + (5 + Q))
((7 * A) + (Q + 5))
((7 * 5) + (A - Q))
((7 * 5) - (Q - A))
((7 / A) + (5 + Q))
((7 / A) + (Q + 5))
((Q + 5) + (A * 7))
((Q + 5) + (7 * A))
((Q + 5) + (7 / A))
((Q + 7) + (A * 5))
((Q + 7) + (5 * A))
((Q + 7) + (5 / A))
((Q * A) + (5 + 7))
((Q * A) + (7 + 5))
((Q * A) * (7 - 5))
((Q / A) + (5 + 7))
((Q / A) + (7 + 5))
((Q / A) * (7 - 5))
```

```
((A * 5) + 7) + Q
((A * 5) + Q) + 7
((A * 7) + 5) + Q
((A * 7) + Q) + 5
((A * 7) - 5) * Q
((A * Q) + 5) + 7
((A * Q) + 7) + 5
((5 + 7) + Q) * A
((5 + 7) + Q) / A
((5 + 7) * A) + Q
((5 + 7) / A) + Q
((5 + Q) + 7) * A
((5 + Q) + 7) / A
((5 * A) + 7) + Q
((5 * A) + Q) + 7
((5 * 7) + A) - Q
((5 * 7) - Q) + A
((5 / A) + 7) + Q
((5 / A) + Q) + 7
((7 + 5) + Q) * A
((7 + 5) + Q) / A
((7 + 5) * A) + Q
((7 + 5) / A) + Q
((7 + Q) + 5) * A
((7 + Q) + 5) / A
((7 + Q) * A) + 5
```

```
((7 + Q) / A) + 5
((7 - 5) * A) * Q
((7 - 5) * Q) * A
((7 - 5) * Q) / A
((7 - 5) / A) * Q
((7 * A) + 5) + Q
((7 * A) + Q) + 5
((7 * A) - 5) * Q
((7 * 5) + A) - Q
((7 * 5) - Q) + A
((7 / A) + 5) + Q
((7 / A) + Q) + 5
((7 / A) - 5) * Q
((Q + 5) + 7) * A
((Q + 5) + 7) / A
((Q + 5) * A) + 7
((Q + 5) * A) / A
((Q + 7) + 5) / A
((Q + 7) * A) + 5
((Q + 7) / A) + 5
((Q - 7) * 5) - A
((Q * A) + 5) + 7
((Q * A) + 7) + 5
((Q / A) + 5) + 7
((Q / A) + 7) + 5
((A + (5 * 7)) - Q)
```

```
(A + (7 * 5)) - Q
(A * (5 + 7)) + Q
(A * (5 + Q)) + 7
(A * (7 + 5)) + Q
(A * (7 + Q)) + 5
(A * (7 - 5)) * Q
(A * (Q + 5)) + 7
(A * (Q + 7)) + 5
(5 + (A * 7)) + Q
(5 + (A * Q)) + 7
(5 + (7 + Q)) * A
(5 + (7 + Q)) / A
(5 + (7 * A)) + Q
(5 + (7 / A)) + Q
(5 + (Q + 7)) * A
(5 + (Q + 7)) / A
(5 + (Q * A)) + 7
(5 + (Q / A)) + 7
(5 * (Q - 7)) - A
(7 + (A * 5)) + Q
(7 + (A * Q)) + 5
(7 + (5 + Q)) * A
(7 + (5 + Q)) / A
(7 + (5 * A)) + Q
(7 + (5 / A)) + Q
(7 + (Q + 5)) * A
(7 + (Q + 5)) / A
```

```
((7 + (Q * A)) + 5)
((7 + (Q / A)) + 5)
((7 - (A * 5)) * Q)
((7 - (5 * A)) * Q)
((7 - (5 / A)) * Q)
((Q + (A * 5)) + 7)
((Q + (A * 7)) + 5)
((Q + (5 + 7)) * A)
((Q + (5 + 7)) / A)
((Q + (5 * A)) + 7)
((Q + (5 / A)) + 7)
((Q + (7 + 5)) * A)
((Q + (7 + 5)) / A)
((Q + (7 * A)) + 5)
((Q + (7 / A)) + 5)
((Q * (7 - 5)) * A)
((Q * (7 - 5)) / A)
A - (Q - (5 * 7))
A - (Q - (7 * 5))
A * (5 + (7 + Q))
A * (5 + (Q + 7))
A * (7 + (5 + Q))
A * (7 + (Q + 5))
A * (Q + (5 + 7))
A * (Q + (7 + 5))
A * (Q * (7 - 5))
A * (A * (7 + Q))
```

```
5 + (A * (Q + 7))
5 + (7 + (A * Q))
5 + (7 + (Q * A))
5 + (7 + (Q / A))
5 + (Q + (A * 7))
5 + (Q + (7 * A))
5 + (Q + (7 / A))
7 + (A * (5 + Q))
7 + (A * (Q + 5))
7 + (5 + (A * Q))
7 + (5 + (Q * A))
7 + (5 + (Q / A))
7 + (Q + (A * 5))
7 + (Q + (5 * A))
7 + (Q + (5 / A))
Q + (A * (5 + 7))
Q + (A * (7 + 5))
Q + (5 + (A * 7))
Q + (5 + (7 * A))
Q + (5 + (7 / A))
Q + (7 + (A * 5))
Q + (7 + (5 * A))
Q + (7 + (5 / A))
Q * (A * (7 - 5))
Q * (7 - (A * 5))
Q * (7 - (5 * A))
Q * (7 - (5 / A))
```

```
Q / (A / (7 - 5))
A + ((5 * 7) - Q)
A + ((7 * 5) - Q)
A * ((5 + 7) + Q)
A * ((5 + Q) + 7)
A * ((7 + 5) + Q)
A * ((7 + Q) + 5)
A * ((7 - 5) * Q)
A * ((Q + 5) + 7)
A * ((Q + 7) + 5)
5 + ((A * 7) + Q)
5 + ((A * Q) + 7)
5 + ((7 + Q) * A)
5 + ((7 + Q) / A)
5 + ((7 * A) + Q)
5 + ((7 / A) + Q)
5 + ((Q + 7) * A)
5 + ((Q + 7) / A)
5 + ((Q * A) + 7)
5 + ((Q / A) + 7)
7 + ((A * 5) + Q)
7 + ((A * Q) + 5)
7 + ((5 + Q) * A)
7 + ((5 + Q) / A)
7 + ((5 * A) + Q)
7 + ((5 / A) + Q)
7 + ((Q + 5) * A)
```



```

7 + ((Q + 5) * A)
7 + ((Q + 5) / A)
7 + ((Q * A) + 5)
7 + ((Q / A) + 5)
Q + ((A * 5) + 7)
Q + ((A * 7) + 5)
Q + ((5 + 7) * A)
Q + ((5 + 7) / A)
Q + ((5 * A) + 7)
Q + ((5 / A) + 7)
Q + ((7 + 5) * A)
Q + ((7 + 5) / A)
Q + ((7 * A) + 5)
Q + ((7 / A) + 5)
Q * ((A * 7) - 5)
Q * ((7 - 5) * A)
Q * ((7 - 5) / A)
Q * ((7 * A) - 5)
Q * ((7 / A) - 5)
Execution Time: 0.001 seconds

=====
=          Do you want to save file?          =
=====
[1] Yes
[2] No
> 1

Input Filename: testcase6
File saved succesfully

```

(Hasil testcase6.txt)

D. Link To Repository

<https://github.com/bangkitdc/Tucil1-STIMA-2022>

E. Table

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	✓
2. Program berhasil running	✓	✓
3. Program dapat membaca input / generate sendiri dan memberikan luaran	✓	✓
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	✓
5. Program dapat menyimpan solusi dalam file teks	✓	✓