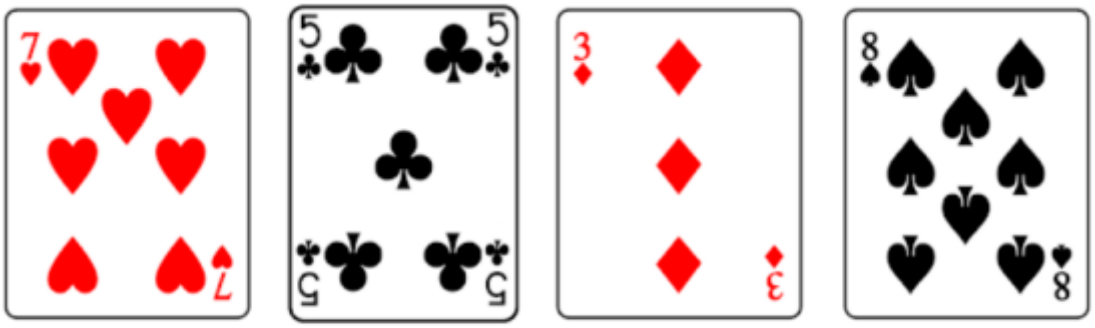
IF2211 - Strategi Algoritma
Laporan Tugas Kecil 1

Nama	NIM
Nicholas Liem	13521135

Institut Teknologi Bandung
Sekolah Teknik Elektro dan Informatika
Tahun Ajaran 2022/2023

Daftar Isi

1 Deskripsi Masalah



MAKE IT 24

1.1 Deskripsi Umum Persoalan

Permainan kartu 24 adalah permainan kartu aritmatika dengan tujuan mencari cara untuk mengubah 4 buah angka random sehingga mendapatkan hasil akhir sejumlah 24. Permainan ini menarik cukup banyak peminat dikarenakan dapat meningkatkan kemampuan berhitung serta mengasah otak agar dapat berpikir dengan cepat dan akurat. Permainan Kartu 24 biasa dimainkan dengan menggunakan kartu remi. Kartu remi terdiri dari 52 kartu yang terbagi menjadi empat suit (sekop, hati, keriting, dan wajik) yang masing-masing terdiri dari 13 kartu (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). Yang perlu diperhatikan hanyalah nilai kartu yang didapat (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). As bernilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, sedangkan kartu bilangan memiliki nilai dari bilangan itu sendiri.

Pada awal permainan moderator atau salah satu pemain mengambil 4 kartu dari dek yang sudah dikocok secara random. Permainan berakhir ketika pemain berhasil menemukan solusi untuk membuat kumpulan nilainya menjadi 24. Pengubahan nilai tersebut dapat dilakukan menggunakan operasi dasar matematika penjumlahan (+), pengurangan (-), perkalian (×), divisi (/) dan tanda kurung (()). Tiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas.

2 Metode Penyelesaian

2.1 Definisi Umum Algoritma Brute Force

Algoritma brute force pada hakekatnya adalah algoritma yang menyelesaikan suatu permasalahan secara langsung tanpa ada konstruksi atau membutuhkan kemampuan berpikir yang lebih.

2.2 Penerapan Algoritma Brute Force untuk Permainan 24

Langkah-langkah brute force dalam permainan 24 ini dapat dibagi menjadi beberapa tahap. Kita tahu bahwa penyelesaian dari permainan 24 adalah masalah kombinatorial antara angka dan operasi sehingga menghasilkan nilai evaluasi sama dengan 24. Berikut adalah analisis dasar dari permainan 24.

- Setiap baris yang akan dievaluasi memiliki tepat 3 buah operasi (ada 4 operasi, tambah, kurang, kali, bagi). Jadi, total banyaknya permutasi operasi pada 3 operasi (boleh berulang) adalah $4 \times 4 \times 4$ atau sama dengan 64.
- Setiap baris yang akan dievaluasi memiliki tepat 4 buah angka. Jadi, total banyaknya permutasi angka tanpa berulang pada 4 angka adalah sebanyak 4! atau 24.
- Ada 5 kemungkinan bentukan tanda kurung untuk setiap baris yang akan dievaluasi. (a, b, c, d adalah angka dan op adalah operasi)
    1. (a op b) op (c op d)
    2. (a op (b op c)) op d
    3. a op (b op (c op d))
    4. a op ((b op c) op d)
    5. ((a op b) op c) op d
- Total maksimal jumlah evaluasi perbaris adalah $64 \times 24 \times 5 = 7680$ banyak operasi.

Oleh sebab itu, ada beberapa hal yang perlu dipersiapkan sebelum langkah algoritma brute force dilakukan.

1. Persiapkan suatu array yang berisi semua permutasi dari 3 operasi, contohnya [+, *, -]. Jadi, sebuah multidimensional array yang setiap elemennya adalah sebuah array yang berisi 3 buah karakter operasi (Jumlahnya ada 64).
2. Persiapkan suatu array yang berisi semua permutasi dari 4 bilangan, contohnya [1, 3, 4, 2]. Jadi, sebuah multidimensional array yang setiap elemennya adalah sebuah array yang berisi 4 buah bilangan (Jumlahnya ada 24).
3. Untuk setiap permutasi operasi kita lakukan operasi tersebut untuk setiap permutasi array 4 bilangan.

4. Setiap operasi pada no.4, dilakukan untuk 5 kali jumlah kemungkinan susunan tanda kurung, jadi total banyaknya operasi yang akan dilakukan adalah

5. Evaluasi permutasi operasi dilakukan untuk setiap permutasi bilangan dilakukan untuk setiap susunan kurung dan cek apakah nilai evaluasinya sama dengan 24, jika ya simpan pada sebuah array penampung.

6. Tambahan untuk menu mengurangi solusi double dengan cara "menghapus" nilai duplikat pada array penampung 4 kartu angka tersebut.

Contoh pelaksanaannya adalah sebagai berikut.

1. Dipilih satu dari sekian permutasi operasi [+, +, *]

2. Dipilih satu dari sekian pemutasi bilangan [1, 2, 3, 4]

3. Pada susunan kurung pertama (a op (b op c)) op d.

4. Terbentuklah (1+(2+3))*4, hasil evaluasinya adalah 24. Kita dapat mengambil ini sebagai solusi.

2.3 Permutasi Operasi dan Bilangan

1. Untuk operasi semua permutasi operasi: [+++], [++-], [++*], [++/], [+-+], [+–], [+-*], [+-/], [+*+], [+*-], [+**], [+*/], [+/+], [+/-], [+/*], [+//], [-++], [-+-], [-+*], [-+/], [–+], [—], [–*], [–/], [-*+], [-*-], [-**], [-*/], [-/+], [-/-], [-/*], [-//], [*++], [*+-], [*+*], [*+/], [*-+], [*–], [*-*], [*-/], [**+], [**-], [***], [**/], [*/+], [*/-], [*/*], [*//], [/++], [/+-], [/+*], [/+/], [/-+], [/–], [/-*], [/-/], [/*+], [/*-], [/**], [/*/], [//+], [//-], [//*], [///]

2. Untuk bilangan 1, 2, 3, dan 4: [1, 2, 3, 4], [1, 2, 4, 3], [1, 3, 2, 4], [1, 3, 4, 2], [1, 4, 2, 3], [1, 4, 3, 2], [2, 1, 3, 4], [2, 1, 4, 3], [2, 3, 1, 4], [2, 3, 4, 1], [2, 4, 1, 3], [2, 4, 3, 1], [3, 1, 2, 4], [3, 1, 4, 2], [3, 2, 1, 4], [3, 2, 4, 1], [3, 4, 1, 2], [3, 4, 2, 1], [4, 1, 2, 3], [4, 1, 3, 2], [4, 2, 1, 3], [4, 2, 3, 1], [4, 3, 1, 2], [4, 3, 2, 1]

3 Eksperimen

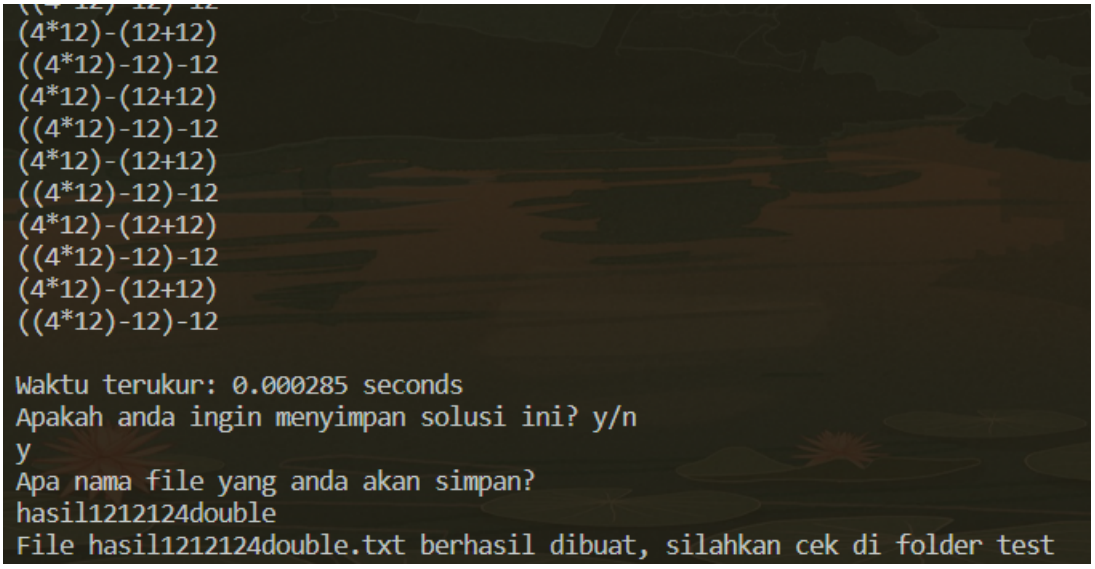
3.1 Kasus Input Sendiri dan Save File

```
Pilih menu yang anda inginkan:
1. Random Card | Kartu yang dihasilkan akan digenerate secara random
2. Isi Sendiri | Masukkan nilai kartu sendiri
3. Keluar Program | Exit
2
Silahkan input 4 kartu anda:
1 2 3 4
Apakah anda mau untuk menghitung tanpa solusi double? (jika ada angka sama) (y/n)
n
Ada sebanyak 242 solusi
((1+2)+3)*4
(1+(2+3))*4
(1*2)*(3*4)
((1*2)*3)*4
(1*(2*3))*4
1*((2*3)*4)
```

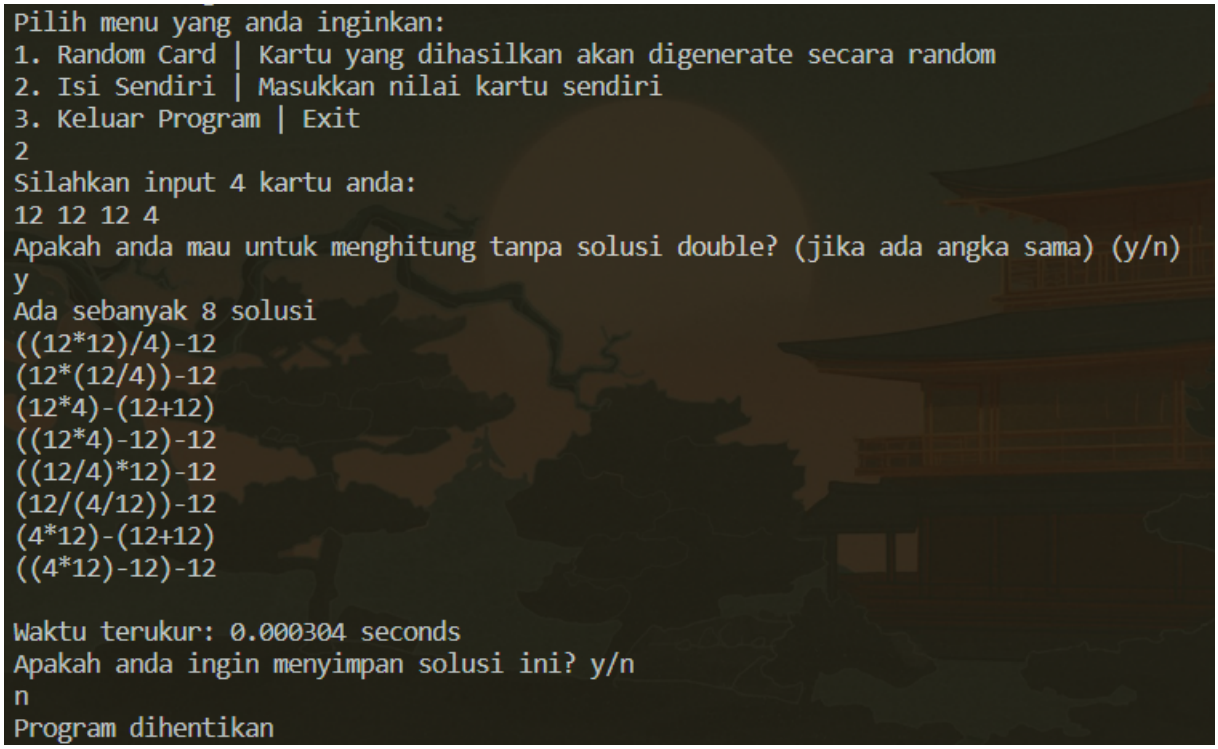
```
4*((3*2)*1)
4*(3*(2*1))
(4*3)*(2/1)
((4*3)*2)/1
(4*(3*2))/1
4*((3*2)/1)
4*(3*(2/1))

Waktu terukur: 0.000388 seconds
Apakah anda ingin menyimpan solusi ini? y/n
y
Apa nama file yang anda akan simpan?
hasil1234
File hasil1234.txt berhasil dibuat, silahkan cek di folder test
```

3.2 Kasus Input Angka Sama dan Save File

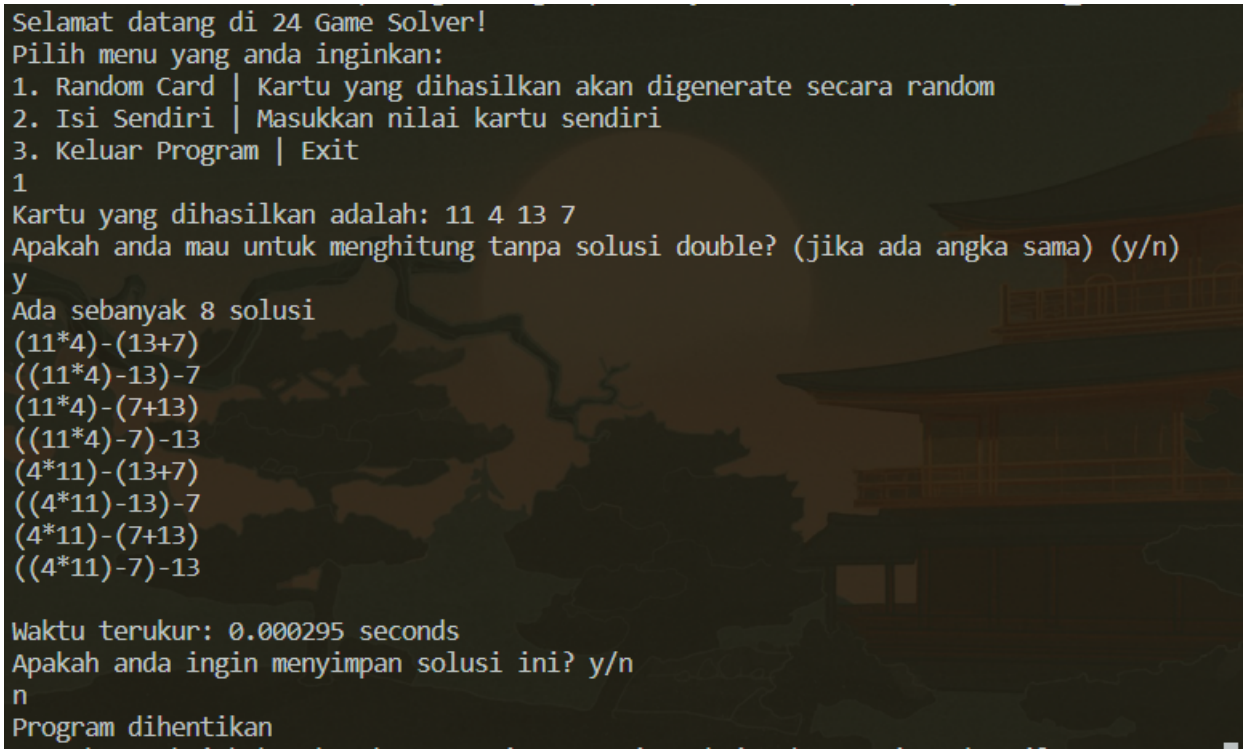
```
Selamat datang di 24 Game Solver!
Pilih menu yang anda inginkan:
1. Random Card | Kartu yang dihasilkan akan digenerate secara random
2. Isi Sendiri | Masukkan nilai kartu sendiri
3. Keluar Program | Exit
2
Silahkan input 4 kartu anda:
12 12 12 4
Apakah anda mau untuk menghitung tanpa solusi double? (jika ada angka sama) (y/n)
n
Ada sebanyak 48 solusi
((12*12)/4)-12
(12*(12/4))-12
((12*12)/4)-12
```



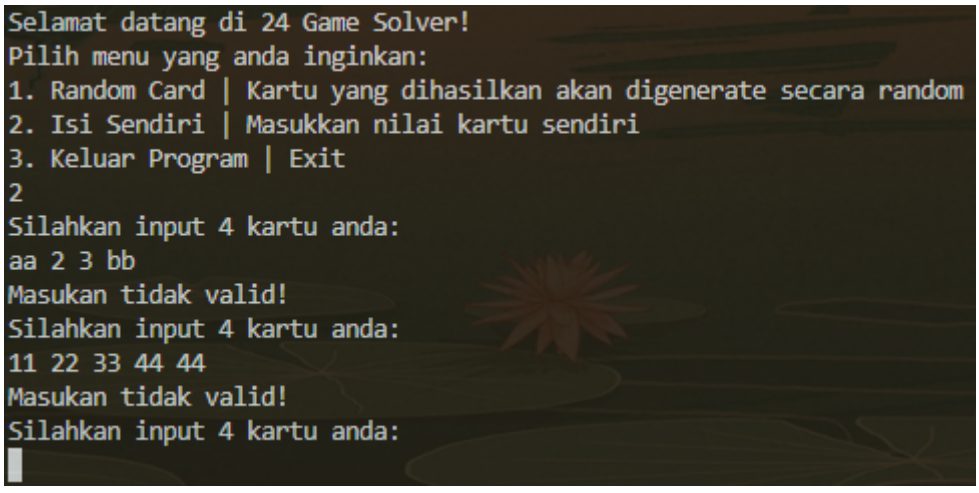
3.3 Kasus Menghilangkan Solusi Double dan Tidak Save File



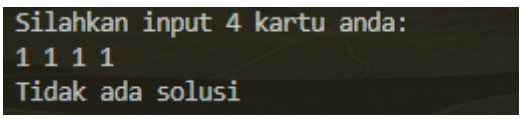
3.4 Kasus Input Random dan Tidak Save File



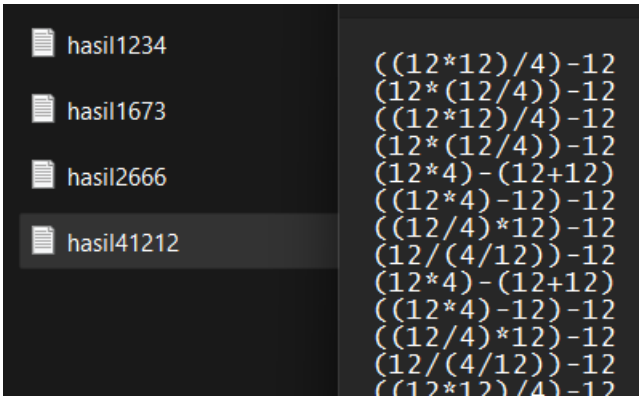
3.5 Kasus Input Tidak Valid



3.6 Kasus Tidak Ada Solusi



3.7 File Tersimpan



3.8 Eksperimen Tambahan

Eksperimen ini menggunakan menu tanpa solusi double untuk kasus angka yang sama.

No. TC	Kartu Dek	Banyak Hasil Solusi	File
1	K 7 K Q	Tidak ada solusi	-
2	9 8 9 K	Tidak ada solusi	-
3	4 K 6 4	2 Solusi	test/tc3.txt
4	As J K Q	32 Solusi	test/tc4.txt
5	K 5 8 4	18 Solusi	test/tc5.txt
6	4 6 8 10	12 Solusi	test/tc6.txt

4 Source Code (C++)

```
#include <iostream>
#include <sstream>
#include <string>
#include <string.h>
#include <cstdlib>
#include <fstream>
#include <chrono>
using namespace std;

bool correctInput(string *num){
    if (*num == "A" || *num == "J" || *num == "Q" || *num == "K" || *num == "2" || *num == "3" || *num == "4"
    || *num == "5" || *num == "6" || *num == "7" || *num == "8" || *num == "9" || *num == "10" || *num == "1"
    || *num == "11" || *num == "12" || *num == "13"){
        return 1;
    } else {
        return 0;
    }
}

int convertStrToInt(string* num){
    if (*num == "A"){
        return 1;
    } else if (*num == "J") {
        return 11;
    } else if (*num == "Q") {
        return 12;
    } else if (*num == "K") {
        return 13;
    } else {
        return stoi(*num);
    }
}

float calculate(float num1, float num2, char op){
    switch(op){
        case '+':
            return (num1 + num2);
        case '-':
            return (num1 - num2);
        case '/':
            return num1/num2;
        default:
            return (num1 * num2);
    }
}

void randomizeCard(int* card){
    int i;
    unsigned seed = chrono::system_clock::now().time_since_epoch().count();
    srand(seed);
```

```
51      for(i = 0; i < 4; i++){
52          card[i] = 1 + (rand() % 13);
53      }
54  }
55
56  int menuMode(){
57      int choice;
58      cout << "Selamat datang di 24 Game Solver!" << endl;
59      while(true){
60          cout << "Pilih menu yang anda inginkan: " << endl;
61          cout << "1. Random Card | Kartu yang dihasilkan akan
        digenerate secara random" << endl;
62          cout << "2. Isi Sendiri | Masukkan nilai kartu sendiri" <<
        endl;
63          cout << "3. Keluar Program | Exit" << endl;
64          cin >> choice;
65          if (choice == 1 || choice == 2 || choice == 3){
66              break;
67          }
68          cout << "Masukan tidak valid!" << endl;
69      }
70      return choice;
71  }
72
73  void clearDuplicate(int cards[24][4]){
74      int count = 0;
75      for(int i = 0; i < 24; i++){
76          for(int j = i + 1; j < 24 ; j++){
77
78              if (cards[j][0] == -1){
79                  continue;
80              } else {
81                  count = 0;
82                  for(int k = 0; k < 4; k++){
83                      if (cards[i][k] == cards[j][k]){
84                          count++;
85                      } else {
86                          break;
87                      }
88                  }
89                  if (count == 4){
90                      for(int k = 0; k < 4; k++){
91                          cards[j][k] = -1;
92                      }
93                  }
94              }
95          }
96      }
97  }
98
99  int main() {
100
101     string n1, n2, n3, n4;
102     int i, j, k, l;
103     int a, b, c, d;
104     char op1, op2, op3;
```

```
105
106     char operators[4] = {'+', '-', '*', '/'};
107     char permOp[64][3];
108     int intCards[4];
109     int permCards[24][4];
110
111     int choice = menuMode();
112
113     if (choice == 1){
114         randomizeCard(intCards);
115         cout << "Kartu yang dihasilkan adalah: " << intCards[0] <<
    " " << intCards[1] << " " << intCards[2] << " " << intCards[3]
    << endl;
116     } else if (choice == 2) {
117         while(true){
118             cout << "Silahkan input 4 kartu anda: " << endl;
119             cin >> n1 >> n2 >> n3 >> n4;
120             if (correctInput(&n1) && correctInput(&n2) &&
    correctInput(&n3) && correctInput(&n4)){
121                 intCards[0] = convertStrToInt(&n1);
122                 intCards[1] = convertStrToInt(&n2);
123                 intCards[2] = convertStrToInt(&n3);
124                 intCards[3] = convertStrToInt(&n4);
125                 break;
126             } else {
127                 cout << "Masukan tidak valid!" << endl;
128             }
129         }
130     } else {
131         cout << "Program dihentikan" << endl;
132         exit(0);
133     }
134
135     char saveChoice;
136     cout << "Apakah anda mau untuk menghitung tanpa solusi double?
    (jika ada angka sama) (y/n)" << endl;
137     cin >> saveChoice;
138
139     /* Begin time measurement */
140     auto begin = std::chrono::high_resolution_clock::now();
141
142     // Membuat permutasi operator
143     int countOp = 0;
144     for(i = 0; i < 4; i++){
145         for(j = 0; j < 4; j++){
146             for(k = 0; k < 4; k++){
147                     permOp[countOp][0] = operators[i];
148                     permOp[countOp][1] = operators[j];
149                     permOp[countOp][2] = operators[k];
150                     countOp++;
151             }
152         }
153     }
154
155     // Membuat permutasi angka
156     int countNum = 0;
```

```
157     for(i = 0; i < 4; i++){
158         for(j = 0; j < 4; j++){
159             if(i == j){
160                 continue;
161             }
162             for(k = 0; k < 4; k++){
163                 if (k == j || k == i){
164                     continue;
165                 }
166                 for(l = 0; l < 4; l++){
167                     if (l == i || l == j || l == k){
168                         continue;
169                     }
170                     permCards[countNum][0] = intCards[i];
171                     permCards[countNum][1] = intCards[j];
172                     permCards[countNum][2] = intCards[k];
173                     permCards[countNum][3] = intCards[l];
174                     countNum++;
175                 }
176             }
177         }
178     }
179 
180     if (saveChoice == 'y' || saveChoice == 'Y'){
181         clearDuplicate(permCards);
182     }
183 
184     float total;
185     int totalVariation = 0;
186     std::stringstream buffer;
187     for(i = 0; i < countNum; i++){
188         for (j = 0; j < countOp; j++){
189 
190             a = (float) permCards[i][0];
191             b = (float) permCards[i][1];
192             c = (float) permCards[i][2];
193             d = (float) permCards[i][3];
194 
195             op1 = permOp[j][0];
196             op2 = permOp[j][1];
197             op3 = permOp[j][2];
198 
199             // Posisi Kurung 1 (a op b) op (c op d)
200             total = calculate(calculate(a, b, op1), calculate(c, d,
    op3), op2);
201             if (abs(total-24) < 0.00001){
202                 buffer << "(" << a << op1 << b << ")" << op2 << "("
    << c << op3 << d << ")" << endl;
203                 totalVariation++;
204             }
205 
206             // Posisi Kurung 2 ((a op b) op c) op d
207             total = calculate(calculate(calculate(a, b, op1), c,
   op2), d, op3);
208             if (abs(total-24) < 0.00001){
```

```
209                 buffer << "((" << a << op1 << b << ")" << op2 << c
    << ")" << op3 << d << endl;
210                 totalVariation++;
211             }
212 
213             //Posisi Kurung 3 (a op (b op c)) op d
214             total = calculate(calculate(a, calculate(b, c, op2),
    op1), d, op3);
215             if (abs(total-24) < 0.00001){
216                 buffer << "(" << a << op1 << "(" << b << op2 << c
    << "))" << op3 << d << endl;
217                 totalVariation++;
218             }
219 
220             // Posisi Kurung 4 a op ((b op c) op d)
221             total = calculate(a, calculate(calculate(b, c, op2), d,
     op3), op1);
222             if (abs(total-24) < 0.00001){
223                 buffer << a << op1 << "((" << b << op2 << c << ")"
    << op3  << d << ")" << endl;
224                 totalVariation++;
225             }
226 
227             // Posisi Kurung 5 a op (b op (c op d))
228             total = calculate(a, calculate(b, calculate(c, d, op3),
     op2), op1);
229             if (abs(total-24) < 0.00001){
230                 buffer << a << op1 << "(" << b << op2 << "(" << c
    << op3 << d << "))" << endl;
231                 totalVariation++;
232             }
233         }
234     }
235 
236     auto end = std::chrono::high_resolution_clock::now();
237     auto elapsed = std::chrono::duration_cast<std::chrono::
    nanoseconds>(end - begin);
238 
239     if (totalVariation > 0){
240         cout << "Ada sebanyak " << totalVariation << " solusi" <<
    endl;
241     } else {
242         cout << "Tidak ada solusi" << endl;
243         exit(0);
244     }
245 
246     cout << buffer.str() << endl;
247     cout << "Waktu terukur: " << elapsed.count() * 1e-9 << "
    seconds" << endl;
248 
249     // Mekanisme saving file
250     while(true && totalVariation > 0){
251         cout << "Apakah anda ingin menyimpan solusi ini? y/n" <<
    endl;
252         cin >> saveChoice;
253         if(saveChoice == 'y' || saveChoice == 'Y'){
```

```
254             string namaFile;
255             cout << "Apa nama file yang anda akan simpan?" << endl;
256             cin >> namaFile;
257             ofstream output;
258             output.open("test/" + namaFile + ".txt");
259             output << buffer.str();
260             output.close();
261             cout << "File " << namaFile << ".txt berhasil dibuat,
    silahkan cek di folder test" << endl;
262             break;
263         } else if (saveChoice == 'n' || saveChoice == 'N'){
264             cout << "Program dihentikan" << endl;
265             break;
266         }
267     }
268
269     return 0;
270 }
```

5 Checklist

Poin	Judul Fitur	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	✓	

6 Lampiran

6.1 Link Repository GitHub

Repo GitHub [Click Me!] or https://github.com/NicholasLiem/Tucil1_13521135