

LAPORAN TUGAS KECIL I

PENYELESAIAN PERMAINAN KARTU 24 DENGAN
ALGORITMA BRUTE FORCE

Laporan dibuat untuk memenuhi salah satu tugas mata kuliah
IF2211 Strategi Algoritma



Disusun oleh:

I Putu Bakta Hari Sudewa 13521150

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023

Daftar Isi

1 Algoritma Brute Force	3
2 Source Program	4
3 Input dan Output	10
3.1 Test Case 1	10
3.2 Test Case 2	11
3.3 Test Case 3	12
3.4 Test Case 4	13
3.5 Test Case 5	14
3.6 Test Case 6	15
3.7 Test Case 7	15
4 Repotori	16
5 Checklist	16
6 Referensi	16

1 Algoritma Brute Force

Algoritma *brute force* merupakan pendekatan yang lempeng (*straightforward*) untuk memecahkan suatu persoalan. Dalam program *24 Card Game Solver* ini algoritma *brute force* digunakan untuk memperoleh semua solusi yang mungkin. Berikut adalah langkah-langkah algoritma *brute force* yang telah disebutkan,

1. Terdapat 4 buah kartu. Dari 4 buah kartu tersebut, carilah semua permutasi yang mungkin, yaitu sebanyak $4! = 24$ buah kemungkinan.
2. Untuk setiap kemungkinan permutasi kartu, carilah semua kombinasi operator (+, -, *, /), yaitu sebanyak $4^3 = 64$ buah kemungkinan.
3. Berikutnya, tentukan kemungkinan banyaknya susunan kurung yang mungkin.

(operand **operator** operand) **operator** (operand **operator** operand)
(operand **operator** (operand **operator** operand)) **operator** operand
operand **operator** ((operand **operator** operand) **operator** operand)
((operand **operator** operand) **operator** operand) **operator** operand
operand **operator** (operand **operator** (operand **operator** operand)))

Terdapat 5 buah kemungkinan susunan kurung yang mungkin.

4. Dengan menggabungkan seluruh kemungkinan yang telah diperoleh pada langkah 1 hingga 3, diperoleh total $24 \times 64 \times 5 = 7680$ kemungkinan. Selanjutnya, untuk setiap kemungkinan ini akan dilakukan operasi aritmetika dan solusi yang diinginkan adalah yang menghasilkan 24.

Kompleksitas dari algoritma *brute force* ini adalah $O(n!m^{n-1})$, dengan n adalah banyaknya kartu dan m adalah banyaknya operator.

2 Source Program

Program dibuat dengan menggunakan bahasa pemrograman C++.

```
1 #include <bits/stdc++.h>
2 #include <fstream>
3 using namespace std;
4
5 const int UNDEF = 99999;
6 const int MAX RAND = 13;
7 const int MAX_CARD_AMT = 4;
8
9 vector<string> numbers(4);
10 vector<string> permutations;
11 vector<bool> chosen(4);
12 set<string> results;
```

Source Code 1: Deklarasi library, variabel, dan konstanta

```
1 int main() {
2     clock_t start, end;
3     string choice;
4
5     cout << "====\n";
6     cout << "WELCOME TO 24 CARD GAME SOLVER!\n";
7     cout << "====\n\n";
8
9     while (choice != "1" && choice != "2") {
10         cout << "Options:\n";
11         cout << "1. Manual Input\n";
12         cout << "2. Randomize Input\n";
13         cout << "Your choice (1/2): ";
14         getline(cin, choice);
15         if (choice != "1" && choice != "2") {
16             cout << "Invalid input. Please try again!\n";
17         }
18         cout << '\n';
19     }
20
21     if (choice == "1") {
22         do {
23             cout << "Available Cards: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K)\n";
24             cout << "Card 1: ";
25             getline(cin, numbers[0]);
26             cout << "Card 2: ";
27             getline(cin, numbers[1]);
28             cout << "Card 3: ";
29             getline(cin, numbers[2]);
30             cout << "Card 4: ";
31             getline(cin, numbers[3]);
32         } while (!validate_input());
33     } else {
34         // Seeding with current time
35         srand(time(0));
36         for (int i = 0; i < 4; i++) {
37             numbers[i] = convert_to_s_number_ar(to_string((rand() % MAX RAND) +
38 1));
39             cout << "Card " << i + 1 << ":" << numbers[i] << '\n';
40         }
41     }
42     cout << '\n';
```

```

42     start = clock();
43     search();
44     end = clock();
45
46     cout << "====\n\n";
47     cout << results.size() << " solution(s) found.";
48     print_results();
49     cout << "\n===\n";
50
51     while (results.size() > 0 && choice != "y" && choice != "Y" &&
52            choice != "n" && choice != "N") {
53         cout << "Save to file? (y/n): ";
54         getline(cin, choice);
55     }
56
57     if (choice == "y" || choice == "Y") {
58         string file_name;
59         do {
60             cout << "File name: ";
61             getline(cin, file_name);
62         } while (!validate_file_name(file_name));
63         write_to_file(file_name);
64         cout << "Result saved to output/" << file_name << ".txt\n";
65     }
66
67     double time_taken = double(end - start) / double(CLOCKS_PER_SEC);
68     cout << "\nTotal time: " << fixed << time_taken << setprecision(5)
69     << " sec.\n";
70
71     return 0;
72 }

```

Source Code 2: Fungsi *main*

Pada Source Code 2 dilakukan proses permintaan input, pemanggilan prosedur *search*, dan proses perhitungan total waktu eksekusi algoritma *brute force*.

```

1 void search() {
2     if (permutations.size() == MAX_CARD_AMT) {
3         vector<string> optr = {"*", "+", "/", "-"};
4
5         vector<vector<pair<pair<int, int>, pair<int, int>>> brackets = {
6             // (a + b) + (c + d)
7             {{0, 1}, {0, 4}, {2, 3}, {6, 10}} ,
8             // ((a + b) + c) + d
9             {{0, 1}, {1, 5}, {2, -7}, {0, 8}, {3, 3}, {10, 10}} ,
10            // a + ((b + c) + d)
11            {{1, 2}, {3, 7}, {3, -9}, {2, 10}}, {{0, 0}, {0, 0}} ,
12            // (a + (b + c)) + d
13            {{1, 2}, {3, 7}, {0, -1}, {0, 8}, {3, 3}, {10, 10}} ,
14            // a + (b + (c + d))
15            {{2, 3}, {5, 9}, {1, -3}, {2, 10}, {0, 0}} ,
16        };
17
18        int solutions_found = 0;
19        for (auto x : brackets) {
20            vector<string> s(11);
21            for (auto y : x) {
22                if (y.first.first == y.first.second) {
23                    s[y.second.second] = permutations[y.first.first];
24                    continue;

```

```

25     }
26     s[y.second.first] = "(";
27     s[y.second.second] = ")";
28     if (y.first.second < 0) {
29         s[-y.first.second] = permutations[y.first.first];
30         continue;
31     }
32     s[y.second.first + 1] = permutations[y.first.first];
33     s[y.second.second - 1] = permutations[y.first.second];
34 }
35
36     int cnt_numbers = 0, i = 0;
37     vector<int> optr_indexs;
38     for (auto y : s) {
39         if (y != " " && y != "(" && y != ")") {
40             cnt_numbers++;
41         } else if (y == "") {
42             optr_indexs.push_back(i);
43         }
44         i++;
45     }
46
47     for (int i = 0; i < 4; i++) {
48         for (int j = 0; j < 4; j++) {
49             for (int k = 0; k < 4; k++) {
50                 s[optr_indexs[0]] = optr[i];
51                 s[optr_indexs[1]] = optr[j];
52                 s[optr_indexs[2]] = optr[k];
53                 float res = calculate_operation(s);
54                 if (cmpf(res)) {
55                     results.insert(generate_string(s));
56                     solutions_found++;
57                 }
58             }
59         }
60     }
61 }
62 } else {
63     for (int i = 0; i < 4; i++) {
64         if (chosen[i])
65             continue;
66         chosen[i] = true;
67         permutations.push_back(convert_to_s_number_ar(numbers[i]));
68         search();
69         chosen[i] = false;
70         permutations.pop_back();
71     }
72 }
73 }
```

Source Code 3: Prosedur *search*

Pada Source Code 3 dilakukan proses pencarian solusi permainan kartu 24 dengan mencoba berbagai kombinasi yang mungkin antara operator dan operand. Terdapat sebuah variabel bernama *brackets* (line 5) yang menyimpan indeks posisi dari masing-masing operator dan operand.

```
1 bool cmpf(float a, float b = 24) { return fabs(a - b) < 0.000005f; }
```

Source Code 4: Fungsi *cmpf*

Fungsi *cmpf* pada Source Code 4 menghasilkan sebuah boolean bernilai true jika float a = float b, false jika sebaliknya.

```

1 float calculate_basic_arithmetic(float a, float b, string s) {
2     if (s == "+") {
3         return a + b;
4     }
5     if (s == "-") {
6         return a - b;
7     }
8     if (s == "*") {
9         return a * b;
10    }
11    if (s == "/" && !cmpf(b, 0)) {
12        return a / (b + 0.0);
13    }
14    return UNDEF;
15 }
```

Source Code 5: Fungsi *calculate_basic_arithmetic*

Fungsi *calculate_basic_arithmetic* pada Source Code 5 menghasilkan sebuah float yang merupakan hasil operasi aritmetika antara float a dan float b, atau UNDEF jika terdapat pembagian oleh nol.

```

1 float stack_calculation(stack<float> &oprд, stack<string> &optr) {
2     float second_number = oprд.top();
3     oprд.pop();
4     float first_number = oprд.top();
5     oprд.pop();
6
7     float result =
8         calculate_basic_arithmetic(first_number, second_number, optr.top());
9     if (result == UNDEF)
10        return UNDEF;
11
12     oprд.push(result);
13
14     optr.pop();
15
16     return result;
17 }
```

Source Code 6: Fungsi *stack_calculation*

Fungsi *stack_calculation* pada Source Code 6 menghasilkan sebuah float hasil operasi aritmetika dua buah float teratas pada stack *oprд*, atau menghasilkan UNDEF jika terdapat pembagian oleh nol.

```

1 float calculate_operation(vector<string> s) {
2     stack<float> oprд;
3     stack<string> optr;
4
5     for (auto x : s) {
6         if (x == "") {
7             continue;
8         if (x != "(" && x != ")" && x != "*" && x != "/" && x != "-" && x != "+")
9         {
10            oprд.push(stoi(x));
11        } else {
12            if (optr.empty() || x == "*" || x == "/" || optr.top() == "(" ||
13                x == "(") {
14                optr.push(x);
15            } else if (x == ")") || ((x == "-") || (x == "+")) && oprд.size() >= 2))
16            {
17                if (stack_calculation(opрд, optr) == UNDEF)
```

```

16         return UNDEF;
17     if (x == ")")
18         optr.pop();
19     }
20 }
21 }
22
23 while (!optr.empty()) {
24     if (stack_calculation(oprd, optr) == UNDEF)
25         return UNDEF;
26 }
27
28 return oprd.top();
29 }
```

Source Code 7: Fungsi *calculate_operation*

Fungsi *calculate_operation* pada Source Code 7 menghasilkan sebuah float hasil dari operasi yang direpresentasikan oleh vector of string *s*. Jika operasi tidak valid akan menghasilkan UNDEF.

```

1 bool validate_input() {
2     for (auto number : numbers) {
3         bool is_valid = false;
4         vector<string> valid_input = {"A", "2", "3", "4", "5", "6", "7",
5             "8", "9", "10", "J", "Q", "K"};
6         for (auto input : valid_input) {
7             if (input == number) {
8                 is_valid = true;
9                 break;
10            }
11        }
12        if (!is_valid) {
13            cout << "Invalid input. Please try again!\n\n";
14            return false;
15        }
16    }
17    return true;
18 }
```

Source Code 8: Fungsi *validate_input*

Fungsi *validate_input* pada Source Code 8 menghasilkan sebuah boolean bernilai true jika input kartu yang dimasukkan sesuai dengan ketentuan yang diberikan, false jika sebaliknya.

```

1 string convert_to_s_number_ar(string data) {
2     if (data == "A")
3         data = "1";
4     else if (data == "1")
5         data = "A";
6     else if (data == "J")
7         data = "11";
8     else if (data == "11")
9         data = "J";
10    else if (data == "Q")
11        data = "12";
12    else if (data == "12")
13        data = "Q";
14    else if (data == "K")
15        data = "13";
16    else if (data == "13")
17        data = "K";
18    return data;
```

19 }

Source Code 9: Fungsi *convert_to_s_number_ar*

Fungsi *convert_to_s_number_ar* pada Source Code 9 menghasilkan sebuah string hasil konversi dari simbol kartu menjadi simbol angka dan sebaliknya.

```
1 string generate_string(vector<string> data) {  
2     string result = "";  
3     for (auto s : data) {  
4         result += convert_to_s_number_ar(s) + " ";  
5     }  
6     return result;  
7 }
```

Source Code 10: Fungsi *generate_string*

Fungsi *generate_string* pada Source Code 10 menghasilkan sebuah string dari parameter vector of string *data*.

```
1 void write_to_file(string file_name) {  
2     ofstream new_file("output/" + file_name + ".txt");  
3     new_file << "Cards: "  
4     for (auto card : numbers) {  
5         new_file << card << ' '  
6     }  
7     new_file << '\n' << results.size() << " solution(s) found.\n";  
8     for (auto result : results) {  
9         new_file << result << '\n';  
10    }  
11    new_file.close();  
12 }
```

Source Code 11: Prosedur *write_to_file*

Prosedur *write_to_file* pada Source Code 11 melakukan proses penulisan file dengan nama file sesuai parameter *file_name* yang diberikan.

```
1 bool validate_file_name(string file_name) {  
2     if (file_name.size() > 100) {  
3         cout << "File name maximum 100 characters!\n";  
4         return false;  
5     }  
6     return true;  
7 }
```

Source Code 12: Fungsi *validate_file_name*

Fungsi *validate_file_name* pada Source Code 12 menghasilkan sebuah boolean bernilai true jika *file_name* yang dimasukkan oleh user tidak melebihi 100 karakter, false jika sebaliknya.

```
1 void print_results() {  
2     cout << '\n';  
3     for (auto result : results) {  
4         cout << result << '\n';  
5     }  
6 }
```

Source Code 13: Prosedur *print_results*

Prosedur *print_results* pada Source Code 13 mencetak semua solusi yang ditemukan ke layar.

3 Input dan Output

3.1 Test Case 1

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 3
Invalid input. Please try again!

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 1

Available Cards: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K)
Card 1: 100
Card 2: 2
Card 3: 3
Card 4: 4
Invalid input. Please try again!

Available Cards: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K)
Card 1: K
Card 2: J
Card 3: 7
Card 4: 10
```

Gambar 1: Test Case 1 - Input manual, validasi input pilihan, dan validasi input kartu

```
=====
6 solution(s) found.
( ( K - J ) * 7 ) + 10
( 7 * ( K - J ) ) + 10
10 + ( ( K - J ) * 7 )
10 + ( 7 * ( K - J ) )
10 - ( ( J - K ) * 7 )
10 - ( 7 * ( J - K ) )

=====
Save to file? (y/n): y
File name: test1
Result saved to output/test1.txt

Total time: 0.037728 sec.
```

Gambar 2: Test Case 1 - Output, simpan ke file, dan total waktu eksekusi

3.2 Test Case 2

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 1

Available Cards: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K)
Card 1: J
Card 2: K
Card 3: Q
Card 4: 2
```

Gambar 3: Test Case 2 - Input manual

```
=====
20 solution(s) found.
( ( J + K ) - Q ) * 2
( ( J + K ) / 2 ) + Q
( ( J - Q ) + K ) * 2
( ( K + J ) - Q ) * 2
( ( K + J ) / 2 ) + Q
( ( K - Q ) + J ) * 2
( J + ( K - Q ) ) * 2
( J - ( Q - K ) ) * 2
( K + ( J - Q ) ) * 2
( K - ( Q - J ) ) * 2
2 * ( ( J + K ) - Q )
2 * ( ( J - Q ) + K )
2 * ( ( K + J ) - Q )
2 * ( ( K - Q ) + J )
2 * ( J + ( K - Q ) )
2 * ( J - ( Q - K ) )
2 * ( K + ( J - Q ) )
2 * ( K - ( Q - J ) )
Q + ( ( J + K ) / 2 )
Q + ( ( K + J ) / 2 )

=====
Save to file? (y/n): n

Total time: 0.037803 sec.
```

Gambar 4: Test Case 2 - Output dan total waktu eksekusi

3.3 Test Case 3

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 1

Available Cards: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K)
Card 1: 5
Card 2: 7
Card 3: 4
Card 4: 2
```

Gambar 5: Test Case 3 - Input manual

```
=====
20 solution(s) found.
( ( 5 + 7 ) * 4 ) / 2
( ( 5 + 7 ) / 2 ) * 4
( ( 7 + 5 ) * 4 ) / 2
( ( 7 + 5 ) / 2 ) * 4
( 4 * ( 5 + 7 ) ) / 2
( 4 * ( 7 + 5 ) ) / 2
( 4 - 2 ) * ( 5 + 7 )
( 4 - 2 ) * ( 7 + 5 )
( 4 / 2 ) * ( 5 + 7 )
( 4 / 2 ) * ( 7 + 5 )
( 5 + 7 ) * ( 4 - 2 )
( 5 + 7 ) * ( 4 / 2 )
( 5 + 7 ) / ( 2 / 4 )
( 7 + 5 ) * ( 4 - 2 )
( 7 + 5 ) * ( 4 / 2 )
( 7 + 5 ) / ( 2 / 4 )
4 * ( ( 5 + 7 ) / 2 )
4 * ( ( 7 + 5 ) / 2 )
4 / ( 2 / ( 5 + 7 ) )
4 / ( 2 / ( 7 + 5 ) )

=====
```

Gambar 6: Test Case 3 - Output

```

Save to file? (y/n): y
File name: kljflsadjl jflkas flkajfl jaslf lkf lakjlf jlkfdsaj lfjs alfjlda flksaj flaskjf laskjf lajf dlcadjf lakjf lakjlf ajf ldkajl dfajlkf lk
ajf lakjf dlakjf lakjf lkafj lkafj lasjf lasjf afjla flka lfdkaj flajlkf dljflkjaijreajrlj flasdj lasjf oaijrejrajlkjfld ajflkaf lakjfl
akjflka fjlkj flakjfld alkfjooijrej ldjflaj fijrlj lj asjo saflajf ajrwel jlkadj flaljk jalkfj lafjla fjoierjwljl ajfdlka flajfierjlja lfjla f
lkajf lajlfkdj ajflkajfrekj lkfj lfajlkf dlaj fa0jrelkj lajfkj ajfiajlkjr leajlja lfjda lfj ajrljae lkjf lakjf lajfejrlakj lfjdlajf lakjfalk f
as
File name maximum 100 characters!
File name: test2
Result saved to output/test2.txt
Total time: 0.031861 sec.

```

Gambar 7: Test Case 3 - Validasi nama file dan total waktu eksekusi

3.4 Test Case 4

```

=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 2

Card 1: A
Card 2: 6
Card 3: 4
Card 4: J

```

Gambar 8: Test Case 4 - Input random

```

=====
20 solution(s) found.
( ( A + J ) - 6 ) * 4
( ( A - 6 ) + J ) * 4
( ( J + A ) - 6 ) * 4
( ( J - 6 ) + A ) * 4
( 6 - 4 ) * ( A + J )
( 6 - 4 ) * ( J + A )
( A + ( J - 6 ) ) * 4
( A + J ) * ( 6 - 4 )
( A - ( 6 - J ) ) * 4
( J + ( A - 6 ) ) * 4
( J + A ) * ( 6 - 4 )
( J - ( 6 - A ) ) * 4
4 * ( ( A + J ) - 6 )
4 * ( ( A - 6 ) + J )
4 * ( ( J + A ) - 6 )
4 * ( ( J - 6 ) + A )
4 * ( A + ( J - 6 ) )
4 * ( A - ( 6 - J ) )
4 * ( J + ( A - 6 ) )
4 * ( J - ( 6 - A ) )

=====
Save to file? (y/n): y
File name: random1
Result saved to output/random1.txt

Total time: 0.035122 sec.

```

Gambar 9: Test Case 4 - Ouput dan total waktu eksekusi

3.5 Test Case 5

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 2

Card 1: 8
Card 2: 7
Card 3: Q
Card 4: Q
```

Gambar 10: Test Case 5 - Input random

```
=====

12 solution(s) found.
( ( 8 - 7 ) * Q ) + Q
( 8 - 7 ) * ( Q + Q )
( Q * ( 8 - 7 ) ) + Q
( Q + Q ) * ( 8 - 7 )
( Q + Q ) / ( 8 - 7 )
( Q / ( 8 - 7 ) ) + Q
Q + ( ( 8 - 7 ) * Q )
Q + ( Q * ( 8 - 7 ) )
Q + ( Q / ( 8 - 7 ) )
Q - ( ( 7 - 8 ) * Q )
Q - ( Q * ( 7 - 8 ) )
Q - ( Q / ( 7 - 8 ) )

=====

Save to file? (y/n): y
File name: random2
Result saved to output/random2.txt

Total time: 0.034759 sec.
```

Gambar 11: Test Case 5 - Output dan total waktu eksekusi

3.6 Test Case 6

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 2

Card 1: 7
Card 2: 9
Card 3: 5
Card 4: 6

=====

6 solution(s) found.
( ( 7 - 5 ) * 9 ) + 6
( 9 * ( 7 - 5 ) ) + 6
6 + ( ( 7 - 5 ) * 9 )
6 + ( 9 * ( 7 - 5 ) )
6 - ( ( 5 - 7 ) * 9 )
6 - ( 9 * ( 5 - 7 ) )

=====

Save to file? (y/n): tidak
Save to file? (y/n): y
File name: random3
Result saved to output/random3.txt

Total time: 0.034843 sec.
```

Gambar 12: Test Case 6 - Input random, output, validasi pilihan *save to file* dan total waktu eksekusi

3.7 Test Case 7

```
=====
WELCOME TO 24 CARD GAME SOLVER!
=====

Options:
1. Manual Input
2. Randomize Input
Your choice (1/2): 2

Card 1: 10
Card 2: 9
Card 3: 3
Card 4: 4

=====

0 solution(s) found.

=====

Total time: 0.033975 sec.
```

Gambar 13: Test Case 7 - Input random, output tanpa solusi, dan total waktu eksekusi

4 Repository

https://github.com/sozyGithub/Tucil1_13521150

5 Checklist

Poin	Ya	Tidak
Program berhasil dikompilasi tanpa kesalahan	✓	
Program berhasil <i>running</i>	✓	
Program dapat membaca input / generate sendiri dan memberikan luaran	✓	
Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	
Program dapat menyimpan solusi dalam file teks	✓	

6 Referensi

- R. Munir, (2023). *Algoritma Brute Force (bagian 1)*[Online]. Tersedia: [https://info.rmatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-\(2022\)-Bag1.pdf](https://info.rmatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf)