МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №8.2

по дисциплине
«Структуры и алгоритмы обработки данных»

Тема. ОТДЕЛЬНЫЕ ВОПРОСЫ АЛГОРИТМИЗАЦИИ
Реализация алгоритмов на основе сокращения числа переборов

Выполнил студент группы ИКБО-43-23 Кощеев М. И.

Принял старший преподаватель Рысин М.Л.

Москва 2024

СОДЕРЖАНИЕ

1 ЦЕЛЬ 3
2 ХОД РАБОТЫ 4
4 ВЫВОДЫ 8
5 ИСПОЛЬЗУЕМЫЙ ИСТОЧНИК 9

1 ЦЕЛЬ

Освоить приёмы реализации алгоритмов на основе сокращения числа переборов

2 ХОД РАБОТЫ

Разработать алгоритм решения задачи с применением метода, указанного в варианте и реализовать программу.
Оценить количество переборов при решении задачи стратегией «в лоб» - грубой силы. Сравнить с числом переборов при применении метода.
Вариант 1:

№_	Задача	Метод
1	Посчитать число последовательностей нулей и единиц длины *n*, в которых не встречаются две идущие подряд единицы.	Динамическое программирование

Алгоритм (main)

1) Ввод с клавиатуры длинны n
2) Вывод результата, используя динамическое программирование
3) Вывод результата, используя метод грубой силы (брутфорс)

Код программы

```
int main() {
    int n;
    cout << "Enter the length of the sequence n: ";
    cin >> n;
    if (n <= 0) {
        cout << "Error" << endl;
        return -1;
    }

    int resultDP = countSequencesDP(n);
    cout << "The number of sequences of length " << n << " without two consecutive units (dynamic programming): " << resultDP << endl;
    int resultBruteForce = countSequencesBruteForce(n);
    cout << "The number of sequences of length " << n << " without two consecutive units (bruteforce): " << resultBruteForce << endl;
    return 0;
}
```

Алгоритм (countSequencesDP):

1) Если n равен 1, то вывод 2
2) Создание вектора dp (количество последовательностей длинны) размером (n+1)*2
3) Установка dp[1][0] = 1 (одна последовательность длины 1, заканчивающаяся на 0)
4) Установка dp[1][1] = 1 (одна последовательность длины 1, заканчивающаяся на 1)
5) Для каждого i от 2 до n:

a. Вычисление количества последовательности длинны i, заканчивающихся на 1
b. Вычисление количества последовательности длинны i, заканчивающихся на 0

6) Вывод общего количества последовательностей длинны n

Код программы:

```
int countSequencesDP(int n) {
    if (n == 1)
        return 2;

    vector<vector<int>> dp(n + 1, value:vector<int>(n:2, value:0));
    dp[1][0] = 1;
    dp[1][1] = 1;

    for (int i = 2; i <= n; ++i) {
        dp[i][0] = dp[i - 1][0] + dp[i - 1][1];
        dp[i][1] = dp[i - 1][0];
    }

    return dp[n][0] + dp[n][1];
}

bool isValidSequence(const string& seq) {
    for (size_t i = 1; i < seq.size(); ++i)
        if (seq[i] == '1' && seq[i - 1] == '1')
            return false;
    return true;
}

```

Алгоритм (countSequencesBruteForce)

1) Инициализация переменной count со значением 0
2) Инициализация переменной limit со значением 2^n
3) Для каждого числа от 0 до n
    a. Инициализация строки seq
    b. Присваивание num значение i
    c. Для каждого числа в num проверяется, является ли текущая цифра в двоичной записи числом 0 или 1 (если 0, добавляем в начало seq 0, если 1, то 1)
4) Деление num на 2, чтобы перейти к следующей цифре в двоичной записи

5) Проверка, seq через isValidSequence. Если да, увеличиваем count на 1
6) Возврат значения count

Код

```
int countSequencesBruteForce(int n) {
    int count = 0;
    int limit = pow(x:2, y:n);

    for (int i = 0; i < limit; i++) {
        string seq;
        int num = i;
        for (int j = 0; j < n; j++) {
            if (num % 2 == 0)
                seq = '0' + seq;
            else
                seq = '1' + seq;
            num /= 2;
        }
        if (isValidSequence(seq)) {
            ++count;
        }
    }
    return count;
}
```

Алгоритм (isValidSequence)

1) Проверка по 2 числа в строке
2) Если встречается 11, вывод false, если 11 не встречается – вывод true

Код

```
bool isValidSequence(const string& seq) {
    for (size_t i = 1; i < seq.size(); ++i)
        if (seq[i] == '1' && seq[i - 1] == '1')
            return false;
    return true;
}
```

Тестирование

Входные данные	Ожидаемый результат для ДП	Фактический результат для ДП	Ожидаемый результат для БФ	Фактический результат для БФ
5	13	13	13	13
8	55	55	55	55
10	144	144	144	144

```
Enter the length of the sequence n:5
 The number of sequences of length 5 without two consecutive units (dynamic programmi
ng): 13
The number of sequences of length 5 without two consecutive units (bruteforce): 13
```

```
Enter the length of the sequence n:8
 The number of sequences of length 8 without two consecutive units (dynamic programmi
ng): 55
The number of sequences of length 8 without two consecutive units (bruteforce): 55
```

```
Enter the length of the sequence n:10
 The number of sequences of length 10 without two consecutive units (dynamic programm
ing): 144
The number of sequences of length 10 without two consecutive units (bruteforce): 144
```

4 ВЫВОДЫ

В результате проделанной работы были освоены приёмы реализации алгоритмов на основе сокращения числа переборов (динамическое программирование и брутфорс)

5 ИСПОЛЬЗУЕМЫЙ ИСТОЧНИК

1. М.Л. РЫСИН, М.В. САРТАКОВ, М.Б. ТУМАНОВА Учебно-методическое пособие СиАОД часть 2