Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)
КАФЕДРА «Информационная безопасность» (ИУ8)

Домашнее задание
По дисциплине «Алгоритмы и структуры данных»
Вариант №20

Выполнил: Никитин А.М., группа ИУ8-52

Преподаватель: Чесноков В.О.

_____ _____
(оценка) (подпись, дата)

Москва, 2017

Теоретическая часть

Постановка задачи:

Реализуйте программу для сборки кубика Рубика. На вход подается текущее состояние кубика, выходом является последовательность поворотов.

Программа будет разработана на языке C++

Ку́бик Ру́бика — механическая головоломка, изобретённая в 1974 году (и запатентованная в 1975 году) венгерским скульптором и преподавателем архитектуры Эрнё Рубиком.

Головоломка представляет собой пластмассовый куб (размер в первоначальном варианте — 3×3×3). Его видимые элементы снаружи выглядят как 26 малых кубиков с 54 видимыми цветными гранями, составляющих один большой куб. Грани большого куба способны вращаться вокруг 3 внутренних осей куба. Каждая из шести граней состоит из девяти квадратов и окрашена в один из шести цветов, в одном из распространённых вариантов окраски, расположенных парами друг напротив друга: красный — оранжевый, белый — жёлтый, синий — зелёный. Повороты граней позволяют переупорядочить цветные квадраты множеством различных способов. Задача игрока заключается в том, чтобы «собрать кубик Рубика»: поворачивая грани куба, вернуть его в первоначальное состояние, когда каждая из граней состоит из квадратов одного цвета.

Кубик Рубика может рассматриваться как пример математической группы.

Каждый из шести поворотов граней кубика Рубика может рассматриваться как элемент симметрической группы множества 48 этикеток кубика Рубика, не являющихся центрами граней. Более конкретно, можно пометить все 48 этикеток числами от 1 до 48 и сопоставить каждому из {F, B, U, D, L, R} элемент симметрической группы.

Тогда группа кубика Рубика G определяется как подгруппа S48, порождаемая шестью поворотами граней: G=< F, B, U, D, L, R> (F – front – фронтальная сторона; B – back – задняя сторона; L – left – левая сторона; R – right – правая сторона; U – up – верхняя сторона; D – down – нижняя сторона).

Порядок группы G равен

|G|=8! 12! 3^8 2^12/3 2 2=43252003274489856000

Определения:

Поворот стороны по часовой стрелке шифруется как заглавная буква, обозначающая эту сторону. Поворот стороны против часовой стрелки шифруется как строчная буква, обозначающая эту сторону.

Начальная сторона — сторона, с которой начинается сборка. Центральные элементы расположены по центру кубика на каждой из его сторон в окружении других восьми элементов. Каждый такой элемент невозможно переместить, и он имеет лишь один цвет.

Угловые элементы расположены на углах кубика. Каждый элемент имеет три разных цвета. Боковые элементы расположены между угловыми элементами. Каждый такой элемент имеет два разных цвета.

Выделяется 4 основных алгоритма сборки кубика Рубика.

1) Метод Коцимбы.

Позволяет находить «почти оптимальные» алгоритмы, не позволяя доказать их минимальность. Сначала составляется таблица, в которой для всех 20 миллиардов «избранных» конфигураций хранится число ходов, необходимых для сборки кубика. Потом для каждой выбранной конфигурации рассматривается минимальный путь перевести её в одну из избранных. Суммарная длина двух участков пути и даёт полную длину пути от заданного состояния к финальному. Тот из них, что окажется самым коротким, и берётся за «почти оптимальный».

2) Алгоритм Рокицкого.

Для каждой конфигурации ищутся все алгоритмы определенной длины, переводящие её в одну из «избранных». Поскольку все разрешённые конфигурации переводятся друг в друга определённым набором поворотов, то рано или поздно исчерпываются все «избранные». Если записать для каждой из них длину предложенного решения, то максимальная из них даст максимальную длину не только для заданного положения, но и ещё для двух миллиардов конфигураций, отличающихся от своей «избранной» той последовательностью движений, что переводит заданную конфигурацию в свою «избранную». Все эти наборы не пересекаются и вместе составляют полное множество из 43 квинтиллионов конфигураций.

3) Алгоритм Бога

Алгоритм, который позволяет собрать кубик Рубика за наименьшее число шагов. Этот метод использует практически полный перебор всех возможных комбинаций вращений кубика Рубика, для нахождения числа Бога (число вращений, за которое можно решить кубик, находящийся в любом состоянии, кстати, равно оно 20) потребовалось несколько недель работы на мощностях Гугла. Гугл не раскрывает характеристики

оборудования, но известно, что было затрачено 1.1 млрд секунд компьютерного времени на выполнение расчетов.

Недостатком данных алгоритмов является то, что их использование занимает большое количество времени, так как используется практически полный перебор, поэтому было принято решение реализовывать сборку кубика Рубика методом креста.

Данный метод имеет огромное количество вариаций, но общая суть остается примерно одинаковой.

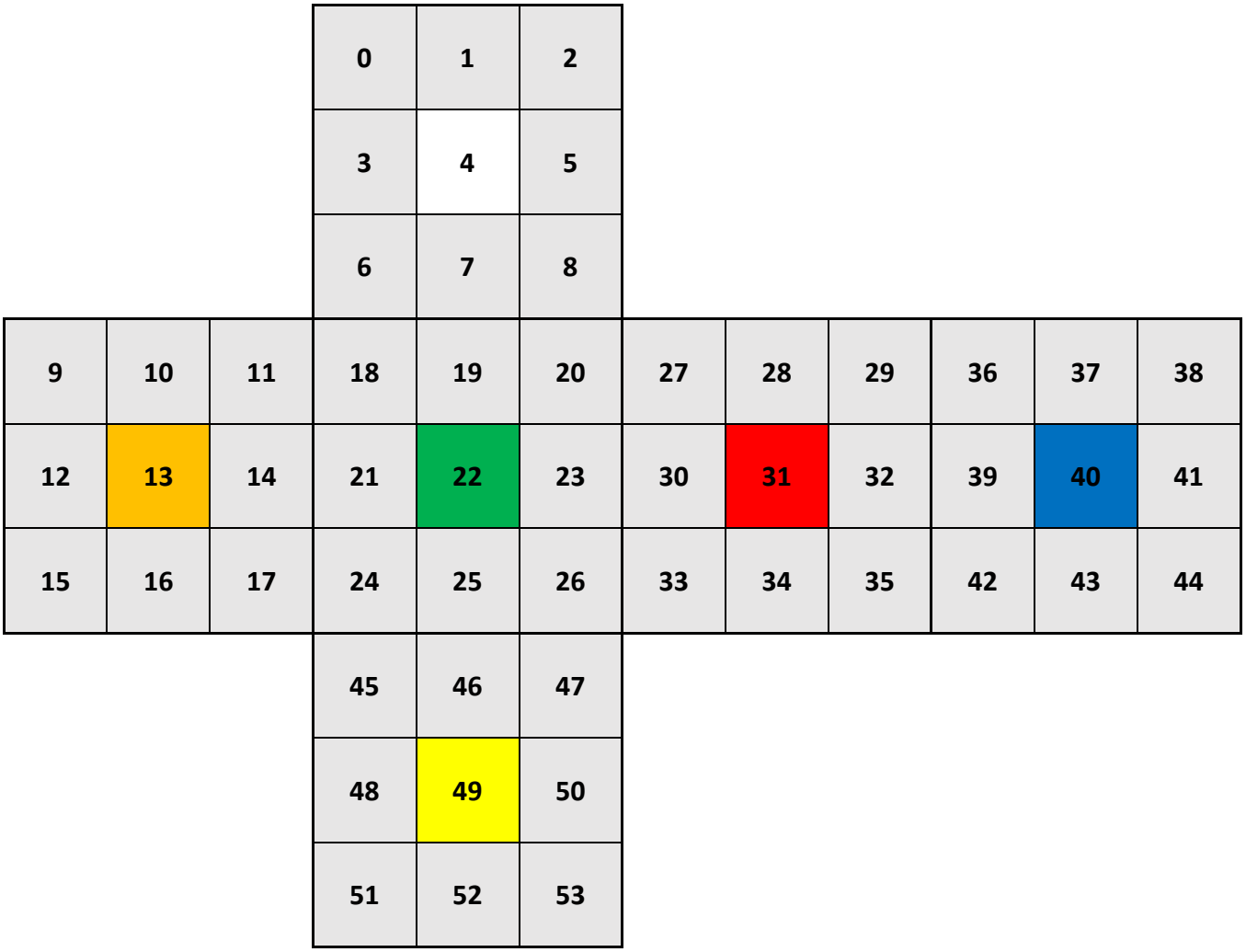
Алгоритм сборки методом креста:

1) Сборка «правильного» креста;
2) Ориентирование углов 1 слоя;
3) Сборка второго слоя;
4) Сборка креста на 3 слое;
5) Позиционирование боковых элементов для «правильного» креста;
6) Позиционирование углов;
7) Ориентирование углов 3 слоя.

Для каждого состояния кубика Рубика предназначена своя формула. В итоге код состоит из ~150 различных формул.

Формат входных данных:

На вход подается строка, состоящая из 54 символов, которая содержит следующие символы, обозначающие цвет элемента: b – голубой, w – белый, r – красный, y – желтый, o – оранжевый, g – зеленый.



Формат выходных данных

Выходная строка содержит инструкции по вращению граней кубика Рубика в виде: BDrUbbL…

Расположение элементов на кубике

ячейка	элемент	N	положение
0	угол	3	верх
1	бок	2	верх
2	угол	2	верх
3	бок	3	верх
4	центр	0	центр
5	бок	1	верх
6	угол	0	верх
7	бок	0	верх
8	угол	1	верх
9	угол	3	право
10	бок	3	низ
11	угол	0	лево
12	бок	10	низ
13	центр	4	центр
14	бок	11	низ
15	угол	7	право
16	бок	7	низ
17	угол	4	лево
18	угол	0	право
19	бок	0	низ
20	угол	1	лево
21	бок	11	верх
22	центр	1	центр
23	бок	8	верх
24	угол	4	право
25	бок	4	низ
26	угол	5	лево

ячейка	элемент	N	положение
27	угол	1	право
28	бок	1	низ
29	угол	2	лево
30	бок	8	низ
31	центр	2	центр
32	бок	9	низ
33	угол	5	право
34	бок	5	низ
35	угол	6	лево
36	угол	2	право
37	бок	2	низ
38	угол	3	лево
39	бок	9	верх
40	центр	3	центр
41	бок	10	верх
42	угол	6	право
43	бок	6	низ
44	угол	7	лево
45	угол	4	верх
46	бок	4	верх
47	угол	5	верх
48	бок	7	верх
49	центр	5	центр
50	бок	5	верх
51	угол	7	верх
52	бок	6	верх
53	угол	6	верх

Описание программы

Struct Center – структуры сохраняющие положение центров граней;

Struct Edge – структуры сохраняющие положение углов;

Struct Corner – структуры сохраняющие положение боковых элементов;

searchEdge – находит необходимые боковые элементы;

searchCorner – находит необходимые углы;

sequence – инициирует повороты граней и записывающая их;

firstStep, secondStep, thirdStep – ф-ии, собирающие 1,2,3 слой кубика соответсвтенно;

whiteCross – собирает белый крест;

cornersFirstStep – ориентирует углы первого слоя;

yellowCross – собирает желтый крест;

finalCorners – ставит углы 3 слоя на правильные места:

permutationFinalCorners – ставит на свои места углы 3 слоя:

positionFinalCorners – позиционирует углы на 3 слое;

isSolved – проверяет собран ли кубик;

R, R2, r, L, L2, l, D, D2, d, U, U2, u, F, F2, f, B, B2, b – осуществляют вращение кубика (двойные, по часовой и против часовой стрелке).

Solve – общая функция сборки;

Test – получает входные данные и частично проверяет их на корректность;

Покрытие тестами

Было разработано 5 тестов, которые проверяют работу алгоритма на корректность:

1) 2 теста с корректной последовательностью (test01.dat, test02.dat), с целью убедиться, что результат получается верным. Полученный результат был практически проверен с помощью сторонних программ и кубика Рубика.
2) 1 тест, в котором содержится неправильное количество символов (test03.dat).
3) 1 тест, в котором некоторых цветов больше, чем других (test04.dat).
4) 1 тест, в котором были поменяни местами «наклейки» на одном углу кубика рубика (test05.dat).
5) 1 тест, в котором были введены некорректные символы (test06.dat)

Со всеми тестами программа успешно справилась.

Во время проверки реакции на тесты было обнаружено и устранено 2 недочета, которые сказывались на корректности работы программы.

Оценка сложности

Сложность по времени

Для данной программы на вход каждый раз подается строка одной и той же длины (54 символа), значит от длины последовательности временная сложность зависеть не может.

По отдельности все методы имеют константную сложность O(1).

Но сложность зависит от того какая именно последовательность была подана на вход, зависит от того, какое количество вращение совершается.

Сложность в среднем имеет вид: O(n*(t+k)), где n – количество поворотов граней; t - число проверок в данном шаге; k – количество действий, которое необходимо совершить, если проверка дала положительный результат.

Минимальная сложность: когда на вход подается уже собранный кубик, будут выполняться только проверки положения кубика, без вращений сторон.

Сложность по памяти

В программе используются: стуктуры (массив), vector, и вспомогательные переменные, для свапа данных. Количество используемой памяти практически не зависит от входной последовательности, только vector, где хранится последовательность действий может увеличиваться или уменьшаться в объемах.