Статья опубликована «Мир ПК». 2004, № 9, с. 82-84

Преступники и автоматы

Максим Мазин, Анатолий Шалыто

В настоящей работе предлагается программное решение одной из логических головоломок на основе теории конечных автоматов, что нетрадиционно.

Введение

Существует обширная литература по занимательным задачам и их решениям. Одна из наиболее известных серий книг по этой тематике принадлежит М. Гарднеру [1 – 3].

Среди этих задач весьма интересен класс логических головоломок. Указанные задачи обычно решаются эвристически, а корректность их решения доказывается средствами математической логики [4, 5].

Для доказательства правильности решения задач этого класса весьма редко используется теория конечных автоматов. В настоящей работе демонстрируется программное решение одной из таких головоломок для его визуализация на основе программирования с явным выделением состояний [6].

Эта задача известна как «Задача о преступниках». Авторы узнали ее от Н.Н. Шамгунова, трехкратного чемпиона Урала по программированию [7]. Сформулируем эту задачу.

Задача о преступниках

Суд приговорил десять преступников к наказанию:

- *каждый из них будет сидеть в отдельной камере, не имея возможности общаться с остальными преступниками;*
- *ежедневно с утра, начиная с первого дня заключения, одного из них будут отводить в карцер, а вечером того же дня возвращать назад в камеру;*
- *в карцере есть лампочка, которую заключенный, находящийся в карцере, сможет включать и выключать. Надзиратели должны тщательно следить за тем, чтобы сидящие в карцере не оставляли друг другу никаких посланий. При этом надзиратели сами не будут ни включать, ни выключать лампочку;*
- *заключение закончиться в тот день, когда один из осужденных сообщит надзирателю о том, что каждый из десяти преступников побывал в карцере хоть один раз. Если это окажется правдой, то все десять преступников будут отпущены на свободу, в противном случае – их казнят;*
- *до начала заключения преступникам разрешено собраться вместе и разработать план действий, который позволит им покинуть место лишение свободы, не прибегая к средствам, не описанным в рассматриваемой постановке задачи;*
- *порядок, в котором заключенные будут сидеть в карцере, им не известен, но известно, что согласно **плану** надзирателей:*

  - *каждый заключенный побывает в карцере;*
  - *после выхода из него заключенный вновь попадает туда через конечное число дней;*
- *предполагается, что ни один из преступников не может ни освободиться, ни умереть иначе как в результате того, что кто-то из заключенных сообщит надзирателю о том, что каждый преступник побывал в карцере.*

Необходимо предложить такой алгоритм поведения преступников, который позволит им выбраться из тюрьмы за конечное время без риска быть казненными.

Сложность предложенной задачи состоит в том, что заключенные могут обмениваться информацией только через состояние лампочки (включена, выключена) – используя один бит. Из постановки задачи следует, что для освобождения преступникам обмен информацией необходим, так как, по крайней мере, один из них должен узнать, что остальные побывали в карцере.

Решение

Предлагаемое решение основано на том, чтобы выбрать одного преступника, который будет собирать информацию, назовем его ***главарем***, в то время как все остальные заключенные будут ему эту информацию поставлять. Каждый заключенный должен сообщить ***главарю*** о том, что побывал в карцере. При этом сообщения от разных заключенных не должны накладываться друг на друга.

Каждый преступник, попадая в карцер, должен руководствоваться следующими правилами:
- все, кроме ***главаря***, обязаны включить лампочку ровно один раз;
- выключать лампочку имеет право только ***главарь***. Выключая лампочку, он узнает о том, что еще один человек побывал в карцере, по крайней мере, один раз;
- ***главарем*** считается преступник, попавший в карцер в первый день. Если до начала заключения лампочка была включена, то ***главарь*** в первый день ее выключает. При этом, естественно, он не считает, что кто-то из десяти преступников побывал в карцере до него;
- включать можно только выключенную лампочку, а выключать только включенную. Например, если ***главарь***, в очередной раз, попав в карцер, обнаруживает лампочку выключенной, то он не предпринимает никаких действий;
- когда ***главарь*** узнает о том, что каждый преступник побывал в карцере, он сообщит об этом надзирателю, который должен освободить заключенных.

Из изложенного следует, что основная идея решения рассматриваемой задачи состоит в том, что ***рядовой преступник*** (не главарь), включая лампочку, сообщает ***главарю*** о том, что побывал в карцере. При этом выключение лампочки ***главарем*** означает, что сообщение принято.

Приведенное выше словесное решение имеет форму традиционную для логических головоломок. Приведем более четкую и строгую форму описания решения, используя автоматный подход [6].

Построим конечный автомат, описывающий поведение преступника. Так как до начала заключения все преступники неотличимы друг от друга, то достаточно одного автомата для описания поведения каждого из них.

Специфицируем интерфейс этого автомата с помощью схемы связей, приведенной на рис. 1.

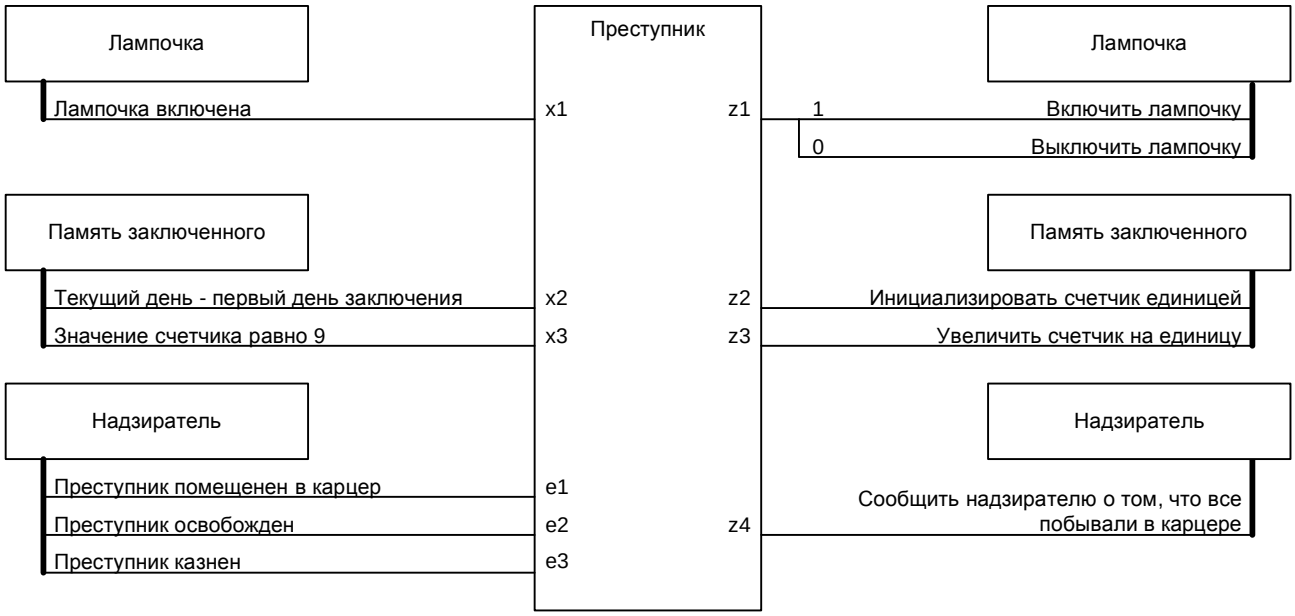

Рис. 1. Схема связей автомата

По постановке и решению задачи построим граф переходов автомата с пятью состояниями:

- «Никогда не включал лампочку»;
- «Главарь»;
- «Однажды включал лампочку»;
- «Освобождение»;
- «Казнь».

Граф переходов, использующий эти состояния, приведен на рис. 2.

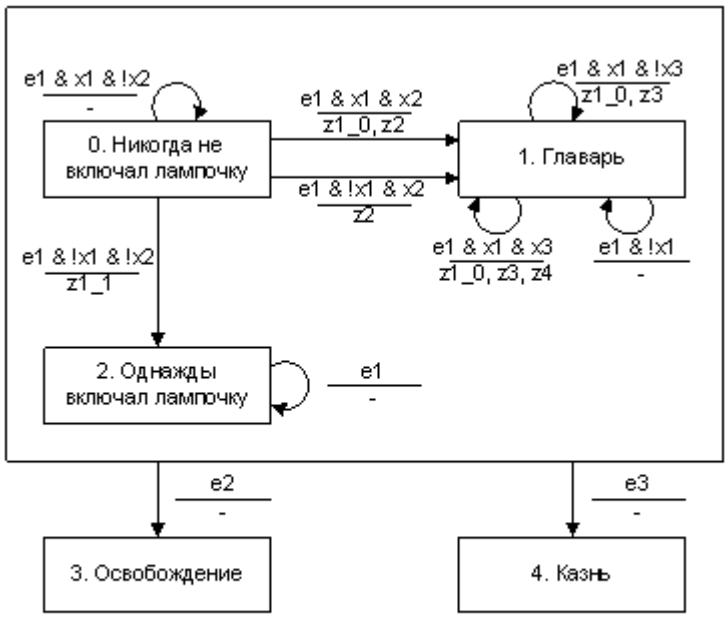

Рис. 2. Граф переходов, описывающий решение задачи

Для соответствия графа постановке задачи введено состояние «Казнь», которое не достижимо, так как решение задачи корректно.

Программная реализация решения

Для **визуализации** решения, представленного в виде автомата, написана программа на языке C++ (Приложение 1). Код программы включает два класса: «Надзиратель» (Jailer) и «Преступник» (Criminal).

В классе «Надзиратель» присутствует поле (_crims) – массив из десяти экземпляров класса «Преступник». «Надзиратель» перебирает преступников в случайном порядке, помещая их на день в карцер, до тех пор, пока один из них не сообщит о том, что все преступники побывали там. Для проверки истинности указанного сообщения в классе «Надзиратель» помечается каждый преступник, побывавший в карцере. Для этого используется массив логических переменных, где значение n-ой из них соответствует тому, что преступник побывал в карцере.

«Надзиратель» генерирует события, вызывая соответствующие методы (обработчики событий), которые реализованы в классе «Преступник». Эти обработчики вызывают автомат, описывающий поведение преступника.

В классе «Преступник» описаны следующие основные поля:
- состояние соответствующего автомата (_y0);
- счетчик, используемый ***главарем*** (_counter), для учета побывавших в карцере преступников;
- номер текущего дня.

Каждый преступник имеет доступ к глобальной переменной, описывающей состояние лампочки (lamp).

Автомат (граф переходов которого изображен на рис. 2), описанные выше входные и выходные воздействия реализованы в виде методов класса «Преступник».

Дополнительно в класс «Преступник» включены методы протоколирования. Для простоты их реализации в этот класс добавлен массив строк с названиями состояний. Каждый экземпляр этого класса осуществляет вывод в отдельный поток.

В приложении 2 приведены два протокола: для рядового преступника и главаря. Из протоколов следует, что преступники освобождены на 96-ой день заключения. Ясно, что для других запусков программы, в силу случайности выбора порядка помещения преступников в карцер, эта величина может измениться (но она не может быть меньше девятнадцати). Отметим, что генератор случайных чисел каждый день выдает номер преступника, который помещается в карцер. В примере, приведенном в приложении 2, генератор выдал номер ***главаря*** в следующие дни: 1, 7, 11, 15. 29, 33, 48, 59, 74, 80, 95. Из протокола следует, что на 96 день выяснилось, что каждый преступник побывал в карцере, по крайней мере, один раз, и поэтому преступники в этот день могут быть освобождены.

Изложенное подтверждает следующее высказывание: "протоколы (истории вычислений), являются конструкциями, вскрывающими механизм работы программы, и поэтому среди теоретиков программирования складывается представление, что множество протоколов лучше характеризует программу, нежели сам исходный программный текст" [8]

Сравнение с аналогичным решением

На следующий день после постановки Н.Н. Шамгуновым задачи описанное выше решение было ему послано, и получено от него подтверждение о правильности решения.

Через некоторое время на форуме сайта http://www.softcraft.ru развернулась дискуссия относительно этой задачи. В результате решение, близкое к изложенному, было предложено И. Дехтяренко [9], а на основе этого решения В.С. Любченко выполнил имитационное моделирование задачи [10].

Основное отличие предлагаемого авторами решения состоит в том, что в настоящей работе весьма просто выделен ***главарь,*** а в указанных работах проблема выделения ***главаря*** оставлена без решения, и поэтому, по нашему мнению, решение задачи на указанном сайте в целом оказывается неполным.

Заключение

Автоматное представление решения лишено недостатков, таких как нечеткость и двусмысленность, характерных для неформальных описаний. Такое представление позволяет в значительной мере формально перейти к тексту программы, визуализирующей решение,

правильность работы которой подтверждается протоколированием в терминах решаемой задачи.

Существуют и другие задачи о преступниках [1, 2] и пленниках [11], но это «уже не по нашему ведомству».

Программа доступна на сайте http://is.ifmo.ru (раздел «Статьи»).

Работа выполнена при поддержке компании “eVelopers Corp.”

Литература

1. *Гарднер М.* Математические головоломки и развлечения. М.: Мир, 1999.
2. *Гарднер М.* Математические досуги. М.: Мир, 2000.
3. *Гарднер М.* Математические новеллы. М.: Мир, 2000.
4. *Кулик Б.А.* Логические основы здравого смысла. СПб.: Политехника, 1997.
5. *Кулик Б.А.* Логика естественных рассуждений. СПб.: Политехника, 2001.
6. *Шалыто А.А., Туккель Н.И.* Программирование с явным выделением состояний // Мир ПК. 2001. № 8, 9. http://www.osp.ru/pcworld. Раздел «Архив».
7. *http://is.ifmo.ru. Раздел «Мысли»*. Задачи на сообразительность. Задача о преступниках.
8. *Ершов А.П.* Смешанные вычисления //В мире науки. 1984. №6.
9. *Дехтяренко И.* Форум сайта http://www.softcraft.ru. Тема №35 «Задача о преступниках».
10. *Любченко В.С.* Задача о преступниках. http://softcraft.ru/auto/ka/crimes/crimes.shtml.
11. *Грэхем Р., Кнут Д., Поташник О.* Конкретная математика. Основание информатики. М.: Мир, 1998.

Статья написана в ноябре 2002 года.

ОБ АВТОРАХ

Мазин Максим Александрович — программист компании «eVelopers Corp.». Бакалавр прикладной математики. E-mail: mazine@rain.ifmo.ru.

Шалыто Анатолий Абрамович — заведующий кафедрой «Технологии программирования» Санкт-Петербургского государственного университета информационных технологий, механики и оптики (СПбГУ ИТМО). E-mail: shalyto@mail.ifmo.ru.

Приложение 1. Код программной реализации

```
// main.cpp
#include <ostream>
#include <fstream>
using namespace std;

#include "Jailer.h"
```

```
#include "Criminal.h"

bool lamp;    // Состояние лампочки

int main()
{
    Jailer *jailer = new Jailer(10);
    jailer->start();
    return 0;
}

// Jailer.h
#ifndef __JAILER_H__
#define __JAILER_H__

class Criminal;

class Jailer
{
public:
    Jailer(int);       // Познакомиться с заключенными
    ~Jailer();         // Разогнать заключенных

    void start();      // Начать заключение
    void inform();     // Проинформировать стражника о конце заключения

protected:
    bool check();      // Проверить все ли побывали в карцере
virtual int getNext(); // Выбрать следующего заключенного
    void release();    // Сообщить всем об освобождении
    void kill();       // Всех казнить

private:
    Criminal** _crims;    // Массив преступников
    bool*      _was;      // Массив заключенных, побывавших в карцере
    ofstream*  _logs;     // Потоки вывода для логов
    int        _num;      // Число заключенных
    bool       _informed; // Охранник проинформирован?
};

#endif // __JAILER_H__

// Jailer.cpp
#include <stdlib.h>
#include <time.h>
#include <fstream>
#include <strstream>
using namespace std;
#include "Jailer.h"
#include "Criminal.h"

Jailer::Jailer(int num):
  _num(num), _informed(false)
{
    srand((unsigned) time(0));
    _crims = new Criminal*[_num];
    _was   = new bool[_num];
    _logs  = new ofstream[_num];
    char buf[256];
```

```
    ostrstream str(buf, 255);
    for (int i = 0; i < _num; i++) {
        str.seekp(0);
        str<<"log-"<<i<<".txt"<<'\0';
        _logs[i].open(buf);
        _crims[i] = new Criminal(this, &_logs[i]);
        _was[i]   = false;
    }
}

Jailer::~Jailer()
{
    for (int i = 0; i < _num; i++)
        delete _crims[i];
    delete[] _crims;
    delete[] _was;
    for (int i = 0; i < _num; i++)
        _logs[i].close();
    delete[] _logs;
}

bool Jailer::check()
{
    for (int i = 0; i < _num; i++)
        if (!_was[i]) return false;
    return true;
}

void Jailer::inform()
{
    _informed = true;
}

void Jailer::kill()
{
    for (int i = 0; i < _num; i++)
        _crims[i]->e3();
}

void Jailer::release()
{
    for (int i = 0; i < _num; i++)
        _crims[i]->e2();
}

int Jailer::getNext()
{
    return rand() % _num;
}

void Jailer::start()
{
    int i;
    do {
        i = getNext();
        _crims[i]->e1();
        _was[i] = true;
        Criminal::nextDay();
    } while(! _informed);
```

```
    if (check()) release();
    else kill();
}

// Criminal.h
#ifndef __CRIMINAL_H__
#define __CRIMINAL_H__

extern bool lamp;
class Jailer;

class Criminal
{
public:
    // Конструктор знакомит :) преступника с надзирателем
    Criminal(Jailer *, ostream *);

    // Преступник освобожден
    bool free() {return _y0 == 3;}

    // Преступник ни разу не включал лампочку
    bool wait() {return _y0 == 0;}

    // Наступил следующий день
    static void nextDay() {_curDay++;}

    // Возвращает номер текущего дня
    static int today() {return _curDay;}

    void e1();    // Преступник помещен в карцер
    void e2();    // Преступник освобожден
    void e3();    // Преступник казнен

protected:
    void A0(int); // A0 - автоматная функция (преступник)

    bool x1();    // Лампочка включена?
    bool x2();    // Текущий день - первый день заключения?
    bool x3();    // Значение счетчика равно 9?

    void z1_0();  // Выключить лампочку
    void z1_1();  // Включить лампочку
    void z2();    // Инициализировать счетчик единицей
    void z3();    // Увеличить счетчик на единицу
    void z4();    // Сообщить о том, что все побывали в карцере

    // Протоколирование начала обработки события автоматом
    void logBegin();

    // Протоколирование конца обработки события автоматом
    void logEnd();

    // Протоколирование изменение состояния автомата
    void logTrans(int);

    // Протоколирование опроса входной переменной
    void logVar(int, int, char *);
```

```
    // Протоколирование выходного действия
    void logOut(int, char *);

    // Протоколирование выходного действия
    void logOut(int, int, char *);

    // Протоколирование события
    void logEvent(int, char *);

private:
    int      _y0;      // Состояние преступника
    int      _counter; // Личный счетчик в голове преступника
    Jailer*  _jailer;  // Надзиратель
    ostream& _log;     // Поток вывода для логов
static int   _curDay;  // Текущий день
static const char* _stateNames[]; // Название состояний для функций
                                  // протоколирования
};

#endif // __CRIMINAL_H__

// Criminal.cpp
#include <ostream>

using namespace std;
#include "Jailer.h"
#include "Criminal.h"

int Criminal::_curDay = 1;
const char* Criminal::_stateNames[] = {
    "Никогда не включал лампочку",
    "Главарь",
    "Однажды включал лампочку",
    "Освобождение",
    "Казнь"
};


Criminal::Criminal(Jailer *jailer, ostream *log):
  _jailer(jailer), _log(*log), _y0(0) {}


// A0 автомат заключенный
void Criminal::A0(int e)
{
    int y0_old = _y0;
    int _x1, _x2, _x3;

    logBegin();
    switch (_y0) {
        case 0: // 0. Никогда не включал лампочку
            _x1 = x1();
            _x2 = x2();
            if (e == 1 &&  _x1 && !_x2) {                    _y0 = 0;}
            else
            if (e == 1 &&  _x1 &&  _x2) {z1_0(); z2();       _y0 = 1;}
            else
            if (e == 1 && !_x1 &&  _x2) {        z2();       _y0 = 1;}
            else
```

```
            if (e == 1 && !_x1 && !_x2) {z1_1();            _y0 = 2;}
            else
            if (e == 2)                 {                   _y0 = 3;}
            else
            if (e == 3)                 {                   _y0 = 4;}
        break;

        case 1: // 1. Главарь
            _x1 = x1();
            _x3 = x3();
            if (e == 1 &&  _x1 && !_x3) {z1_0(); z3();      _y0 = 1;}
            else
            if (e == 1 &&  _x1 &&  _x3) {z1_0(); z3(); z4(); _y0 = 1;}
            else
            if (e == 1 && !_x1)         {                   _y0 = 1;}
            else
            if (e == 2)                 {                   _y0 = 3;}
            else
            if (e == 3)                 {                   _y0 = 4;}
        break;

        case 2: // 2. Однажды включал лампочку
            if (e == 1)                 {                   _y0 = 2;}
            else
            if (e == 2)                 {                   _y0 = 3;}
            else
            if (e == 3)                 {                   _y0 = 4;}
        break;

        case 3: // 3. Освобождение
        break;

        case 4: // 4. Казнь
        break;

    }

    if (_y0 != y0_old)
        logTrans(y0_old);

    logEnd();
}

void Criminal::e1() {
      logEvent(1, "Преступник помещен в карцер");
      A0(1);
}

void Criminal::e2() {
      logEvent(2, "Преступник освобожден");
      A0(2);
}

void Criminal::e3() {
      logEvent(3, "Преступник казнен");
      A0(3);
}

bool Criminal::x1()
```

```
{
    logVar(1, lamp?1:0, "Лампочка включена?");
    return lamp;
}

bool Criminal::x2()
{
    logVar(2, (_curDay == 1)?1:0,"Текущий день - первый день заключения?");
    return _curDay == 1;
}

bool Criminal::x3()
{
    logVar(3, (_counter == 9)?1:0, "Значение счетчика равно 9?");
    return _counter == 9;
}

void Criminal::z1_0()
{
    logOut(1, 0, "Выключить лампочку");
    lamp = false;
}

void Criminal::z1_1()
{
    logOut(1, 1, "Включить лампочку");
    lamp = true;
}

void Criminal::z2()
{
    logOut(2, "Инициализировать счетчик единицей");
    _counter = 1;
}

void Criminal::z3()
{
    logOut(3, "Увеличить счетчик на единицу");
    _counter ++;
}

void Criminal::z4()
{
    logOut(4, "Сообщить надзирателю о том, что все побывали в карцере");
    _jailer->inform();
}

void Criminal::logBegin()
{
    _log<<"A0."<<_y0<<" ["<<_stateNames[_y0]<<"]"<<endl;
}

void Criminal::logEnd()
{
    _log<<"A0."<<_y0<<" ["<<_stateNames[_y0]<<"]"<<endl<<endl;
}

void Criminal::logTrans(int y0_old)
{
```

```
    _log<<"    "<<"A0."<<_y0<<" ["<<_stateNames[y0_old]<<"] => "
                <<"A0."<<_y0<<" ["<<_stateNames[_y0]<<"]"<<endl;
}

void Criminal::logVar(int x, int v, char *str)
{
    _log<<"    "<<"x"<<x<<"="<<v<<": ["<<str<<"] - "<<(v?"ДА":"НЕТ")<<endl;
}

void Criminal::logOut(int z, char *str)
{
    _log<<"    "<<"z"<<z<<":   ["<<str<<"]"<<endl;
}

void Criminal::logOut(int z, int v, char *str)
{
    _log<<"    "<<"z"<<z<<"_"<<v<<": ["<<str<<"]"<<endl;
}

void Criminal::logEvent(int e, char *str)
{
    _log<<"e"<<e<<": ["<<str<<"] - день: "<<_curDay<<endl;
}
```

Приложение 2. Пример протокола

Рядовой преступник:

```
e1: [Преступник помещен в карцер] - день: 22
A0.0 [Никогда не включал лампочку]
    x1=1: [Лампочка включена?] - ДА
    x2=0: [Текущий день - первый день заключения?] - НЕТ
A0.0 [Никогда не включал лампочку]

e1: [Преступник помещен в карцер] - день: 27
A0.0 [Никогда не включал лампочку]
    x1=1: [Лампочка включена?] - ДА
    x2=0: [Текущий день - первый день заключения?] - НЕТ
A0.0 [Никогда не включал лампочку]

e1: [Преступник помещен в карцер] - день: 31
A0.0 [Никогда не включал лампочку]
    x1=0: [Лампочка включена?] - НЕТ
    x2=0: [Текущий день - первый день заключения?] - НЕТ
    z1_1: [Включить лампочку]
    A0.2 [Никогда не включал лампочку] => A0.2 [Однажды включал лампочку]
A0.2 [Однажды включал лампочку]

e1: [Преступник помещен в карцер] - день: 38
A0.2 [Однажды включал лампочку]
A0.2 [Однажды включал лампочку]

e1: [Преступник помещен в карцер] - день: 41
A0.2 [Однажды включал лампочку]
A0.2 [Однажды включал лампочку]

e1: [Преступник помещен в карцер] - день: 60
A0.2 [Однажды включал лампочку]
A0.2 [Однажды включал лампочку]

e1: [Преступник помещен в карцер] - день: 85
A0.2 [Однажды включал лампочку]
```

```
A0.2 [Однажды включал лампочку]

e1: [Преступник помещен в карцер] - день: 88
A0.2 [Однажды включал лампочку]
A0.2 [Однажды включал лампочку]

e2: [Преступник освобожден] - день: 96
A0.2 [Однажды включал лампочку]
    A0.3 [Однажды включал лампочку] => A0.3 [Освобождение]
A0.3 [Освобождение]
```

Главарь

```
e1: [Преступник помещен в карцер] - день: 1
A0.0 [Никогда не включал лампочку]
    x1=0: [Лампочка включена?] - НЕТ
    x2=1: [Текущий день - первый день заключения?] - ДА
    z2:   [Инициализировать счетчик единицей]
    A0.1 [Никогда не включал лампочку] => A0.1 [Главарь]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 7
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 11
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 15
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 29
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 33
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 48
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
```

```
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 59
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 74
A0.1 [Главарь]
    x1=0: [Лампочка включена?] - НЕТ
    x3=0: [Значение счетчика равно 9?] - НЕТ
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 80
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=0: [Значение счетчика равно 9?] - НЕТ
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
A0.1 [Главарь]

e1: [Преступник помещен в карцер] - день: 95
A0.1 [Главарь]
    x1=1: [Лампочка включена?] - ДА
    x3=1: [Значение счетчика равно 9?] - ДА
    z1_0: [Выключить лампочку]
    z3:   [Увеличить счетчик на единицу]
    z4:   [Сообщить надзирателю о том, что все побывали в карцере]
A0.1 [Главарь]

e2: [Преступник освобожден] - день: 96
A0.1 [Главарь]
    x1=0: [Лампочка включена?] - НЕТ
    x3=0: [Значение счетчика равно 9?] - НЕТ
    A0.3 [Главарь] => A0.3 [Освобождение]
A0.3 [Освобождение]
```