# Задача 4. Простой BSON

| | |
|---|---|
| Источник: | основная |
| Имя входного файла: | `input.txt` |
| Имя выходного файла: | `output.txt` |
| Ограничение по времени: | 1 секунда |
| Ограничение по памяти: | разумное |

Формат BSON является бинарным вариантом известного формата JSON. Его спецификация доступна по адресу: `http://bsonspec.org/spec.html`

В файле `input.txt` записан документ (объект) в формате BSON, внутри которого нет поддокументов (подобъектов и подмассивов). В этом объекте могут быть лишь поля следующих типов: `double`, `string`, `bool`, `null`, `int32`, `int64`. Нужно прочитать этот BSON, и вывести его в текстовом виде как JSON.

В описании формата BSON используются следующие базовые типы:

- `byte`: просто байт.
- `int32`: знаковое 32-битное целое число.
- `int64`: знаковое 64-битное целое число.
- `double`: вещественное число двойной точности.

Все числа записываются с little-endian порядком байтов. Целые числа записываются в дополнительном коде, а вещественные — согласно стандарту IEEE 754.

Документ BSON начинается с числа типа `int32`, которое обозначает размер всего документа в байтах. Далее идёт описание всех полей документа одно за другим. В самом конце документа стоит дополнительный нулевой байт.

Описание поля начинается с одного байта $T$, который определяет, какого типа значение в нём записано:

- $T = 1$: значение типа `double`.
- $T = 2$: значение типа `string`.
- $T = 8$: значение типа `bool`.
- $T = 10$: значение типа `null`.
- $T = 16$: значение типа `int32`.
- $T = 18$: значение типа `int64`.

Сразу после типа записано имя поля в виде строки, заканчивающейся на дополнительный нулевой байт. После этого нулевого байта записано значение поля.

Значение поля типа `bool` может быть равно либо `false`, либо `true`. Значение `false` записывается в BSON-файле как байт с нулевым значением, а `true` — как байт с единичным. Для поля типа `null` никаких байтов в значение поля **не** пишется.

Значение поля типа `string` записывается следующий образом. Сначала записано число $L > 0$ типа `int32`, которое равно количеству байтов в строке, включая дополнительный нулевой байт. В следующих $L - 1$ байтах задана сама строка. В конце добавлен дополнительный нулевой байт.

При записи документа в текстовом формате JSON в первой строке пишется открывающая фигурная скобка, а в последней — закрывающая фигурная скобка. Каждая строка между ними описывает одно поле. Поля нужно описывать в том же порядке, в котором они даны в BSON-файле. Описание поля начинается с имени поля, заключённого в двойные кавычки, и двоеточия сразу после него. Далее должен быть поставлен один пробел, после которого записано значение поля. В самом конце строки должна стоят запятая, если только это поле не является последним полем документа.

Значение типа `string` заключается в двойные кавычки, остальные значения — нет. Зна-

чение типа `bool` пишется как одно из слов `false` или `true`. Значение типа `null` пишется как слово `null`. Значение целочисленного типа пишется в десятичной системе исчисления без ведущих нулей (как обычно). Значение типа `double` должно быть записано при помощи `printf` с форматом `"%0.15g"`.

Гарантируется, что все числа типа `double` являются нормальными числами и по модулю не превышают $10^{100}$. Гарантируется, что все строковые значения и имена полей **не** содержат символов, требующих экранирования в формате JSON. То есть они не содержат символов обратного слэша (ASCII 92), двойных кавычек (ASCII 34), а также контрольных символов (ASCII 0-31). Учтите, что строковые значения и имена полей задаются в кодировке UTF-8, что в частности имеет значение для русских символов в примере.

Размер входного файла не превышает один килобайт.

# Пример

input.txt

```
5F 00 00 00 02 48 57 00 0E 00 00 00 48 65 6C 6C
6F 2C 20 57 6F 72 6C 64 21 00 02 D0 9F 20 D0 9C
00 17 00 00 00 D0 9F D1 80 D0 B8 D0 B2 D0 B5 D1
82 20 D0 B2 D1 81 D0 B5 D0 BC 21 00 01 64 62 6C
00 66 66 66 66 66 BA 81 40 10 69 6E 74 00 25 00
00 00 0A 6E 6F 6E 65 00 08 62 6C 6C 00 01 00
```

output.txt

```
{
    "HW": "Hello, World!",
    "П М": "Привет всем!",
    "dbl": 567.3,
    "int": 37,
    "none": null,
    "bll": true
}
```

# Комментарий

Пример входных и выходных данных можно скачать по ссылке.

В целях тестирования вы можете создавать BSON-файлы при помощи online-конвертера: `https://json-bson-converter.appspot.com` Учтите, что конвертация обратно в JSON на этом сайте **не** работает.

Кроме того, вы можете использовать для конвертации следующие скрипты на языке Python (не забудьте запустить "`pip install bson`" после установки Python 3):

```
import bson, json
with open("input.txt", "rt", encoding = "utf-8") as f:
    data = json.load(f)
with open("output.txt", "wb") as f:
    f.write(bson.dumps(data))
```

```
import bson, json
with open("input.txt", "rb") as f:
    data = bson.loads(f.read())
with open("output.txt", "wt", encoding = "utf-8") as f:
    json.dump(data, f, indent = 4, ensure_ascii = False)
    print("", file=f)
```