## Описание

Да предположим, че имаме въображаемо устройство, което периодично изпраща данни. Данните пристигат в пакети, по протокол, описан по-долу.

Това, което се иска в задачата, е да се напише парсър, който обработва суровите байтове, които идват от устройството. Самата комуникация с устройството в случая не е важна, и се предполага, че се управлява от друга, вече написана, част от програмата.

Тъй като връзката с устройството не е 100% надеждна, е възможно да има грешки в получените байтове. Парсърът трябва да може да открива такива грешки, без това да повлиява на работата му.

## Протокол

Един правилно формиран пакет съдържа в себе си, освен същинските данни, следната допълнителна информация:

* Заглавие, два байта с константна стойност, които указват началото на пакета.
* Дължина на данните: 4 бита. Указва колко байта са данните. Тази дължина се отнася единствено и само за същинските данни, и в нея не се брои допълнителната информация.
* Идентификатор: 4 бита. Служи за информация на останалата част от програмата.
* Чексума, един байт, накрая на съобщението, служещ за валидиране на получения пакет.

Байтовете в пакета са подредени по следния начин:

| Номер на байт | Описание |
| --- | --- |
| 0 | Константна стойност 0x44 (68 десетично) |
| 1 | Константна стойност 0x43 (67 десетично) |
| 2 | Байт, който съдържа в себе си две стойности от по 4 бита:   * Горните (старши) битове съдържат дължината на същинските данни: броят байтове, *n*, които следват след този. * Долните (младши) битове съдържат идентификатор   Например, ако този байт е със стойност 93 (0x5D), дължината на данните, *n*, е 5 байта, а идентификатора е 13 (0xD). |
| 3..3+*n*-1 | Данни (общо *n* байта) |
| 3+*n* | Чексума. Чексумата се формира, като се сумират всички предишни байтове (от 0 до 3+*n*-1), по модул 256. |

## Примери

Стойностите на байтовете са дадени в шестнадесетична бройна система.

Пример за валиден пакет:

44 43 3A 11 22 33 27

Пояснение: същинските данни са 3 байта (0x11, 0x22, 0x33), идентификатора а 0xA, чексумата е 0x27, получена, чрез събиране на всички останали байтове по модул 256.

Пример за валиден пакет:

44 43 3A 11 22 33 27 FF FF FF

Пояснение: пакета е същия като предишния път. След края има три допълнителни байта със стойност 0xFF, но това не е грешка.

Пример за невалиден пакет:

44 43 3A 10 22 33 27

Пояснение: чексумата е грешна.

Пример за невалиден пакет:

44 43 BA 11 22

Пояснение: няма достатъчно байтове в пакета

Пример за невалиден пакет:

44 43 0A 91

Пояснение: съобщението е празно (0 байта същински данни).

Пример за невалиден пакет:

42 43 3A 10 22 33 27

Пояснение: заглавието е грешно.

## Задание

Очаква се да се напише функция, която попълва следната структура:

struct Protocol {

enum ParseResult {

PARSE\_OK,

PARSE\_NOT\_ENOUGH\_DATA,

PARSE\_MESSAGE\_EMPTY,

PARSE\_WRONG\_HEADER,

PARSE\_WRONG\_CHECKSUM  
 };

ParseResult result;

int identifier;

int numDataBytes;

char dataBytes[15];  
};

Функцията трябва да има следната сигнатура:

void protocolParse(const char\* data, int numBytes, Protocol\* pResult);

Параметрите на функцията са както следва:

* const char\* data: суровите байтове, дошли от устройството
* int numBytes: броят байтове, налични в паметта, сочена от data. Иначе казано, броят байтове, пристигнали от устройството.
* Protocol\* pResult: указател към структура Protocol, където трябва да се запише резултата.

Структурата трябва да се попълни както следва:

* result: успешно ли е било парсването. Ако е било неуспешно, вида грешка:
* PARSE\_OK: успешно парсване, без грешка
* PARSE\_NOT\_ENOUGH\_DATA: броя байтове, дошъл от устройството (параметъра numBytes), не е достатъчен, иначе казано, пакета е непълен
* PARSE\_MESSAGE\_EMPTY: съобщението е празно (0 байта същински данни)
* PARSE\_WRONG\_HEADER: заглавието е грешно (първите два байта)
* PARSE\_WRONG\_CHECKSUM: чексумата не съвпада
* identifier: идентификатора
* numDataBytes: брой байтове в същинските данни
* dataBytes: същинските данни

Ако е имало грешка при парсването, не е необходимо да се попълва нищо друго по структурата освен result.

## Примери

Да вземем първия пример от по-горе. При него параметрите на функцията биха били:

* data: 44 43 3A 11 22 33 27
* numBytes: 7

Функцията трябва да попълни pResult по следния начин:

* result: PARSE\_OK
* identifier: 0xA
* numDataBytes: 0x3
* dataBytes: {0x11, 0x22, 0x33}

И един от примерите за невалиден пакет:

* data: 44 43 BA 11 22
* numBytes: 5

Функцията трябва да попълни pResult по следния начин:

* result: PARSE\_NOT\_ENOUGH\_DATA
* identifier, numDataBytes, dataBytes: няма значение