

## **C++. Задание 1. Множественное наследование, управление памятью (9 баллов)**

Разработать консольное приложение позволяющее вводить/выводить/удалять информацию о кухонной утвари [а почему бы и нет :)]. Приложение должно удовлетворять следующим требованиям:

1. Содержать иерархию классов
  - 1.1. Абстрактный класс «Кухонная утварь»

Поля:

    - Инвентарный номер
  - 1.2. Класс «Плита», наследуется от «Кухонная утварь»

Поля:

    - Цвет
  - 1.3. Класс «Кастрюля», наследуется от «Кухонная утварь»

Поля:

    - Объем
  - 1.4. Класс «Плита электрическая», наследуется от «Плита»

Поля:

    - Мощность
  - 1.5. Класс «Плита газовая», наследуется от «Плита»

Поля:

    - Расход газа
  - 1.6. Класс «Мультиварка», наследуется от «Плита электрическая» и «Кастрюля»

Поля:

    - Имеет ли функцию скороварки
2. Объекты хранить в динамической структуре в виде упорядоченного бинарного дерева. В качестве ключа для помещения объекта в дерево использовать инвентарный номер.
3. Реализовать возможность
  - 3.1. Ввода данных
    - Тип кухонной утвари
    - Инвентарный номер
    - Поля, соответствующие конкретной утвари

При попытке добавления объекта с инвентарным номером, уже присутствующим в дереве, выдавать соответствующее сообщение и выводить информацию о найденном объекте.
  - 3.2. Удаления данных (по инвентарному номеру)
  - 3.3. Вывода данных (прямой, обратный, симметричный обходы дерева)
4. Для вывода данных реализовать метод print() в каждом классе. При этом метод класса должен печатать только внутренние поля класса + вызывать метод print() классов-родителей.

Критерии оценки. Максимум за задание 9 баллов.

Выполнение минимальных требований: 6 баллов.

Поддержка сбалансированности дерева (в виде AVL-дерева): +3 балла.

Разрешается вместо предложенной иерархии использовать свою при условии, что в ней будет не меньше классов и множественное наследование (по согласованию с преподавателем).

## Задание 2. Конечные автоматы (6 баллов)

Входной файл содержит последовательность байт и состоит из пакетов. Общая структура пакета приведена ниже. Требуется преобразовать данные, содержащиеся в пакетах первичных данных (ID = 0x87), в текстовый файл.

### Структура пакета

<SYNC><SYNC><LEN><ID> [данные] <CRC>

где

- <SYNC> - 1 байт синхронизации (0xAA)
- <LEN> - 1 байт, длина пакета (включаются байт идентификатор, байты данных и контрольные байты);
- <ID> - 1 байт, идентификатор пакета, может иметь любое значение, не равное <SYNC>;
- [данные] - необязательная часть, содержащая передаваемые данные;
- <CRC> - 2 байта, контрольная информация пакета.

В качестве алгоритма проверки содержимого пакета применяется контроль циклическим избыточным кодом (ЦИК, англ. cyclical redundancy check, CRC). Расчет ЦИК производится для всех байт пакета, начиная с идентификатора пакета <ID>, и заканчивая последним байтом перед контрольной суммой.

Программа на языке C, реализующая табличный алгоритм вычисления байтов CRC, прилагается.

Данные в пакете первичных данных (ID = 0x87) содержат значения 16 параметров:

№	Смещение	Описание параметра	Тип
1	0, 1, 2, 3	Ax	int32s
2	4, 5, 6, 7	Ay	int32s
3	8, 9, 10, 11	Az	int32s
4	12, 13, 14, 15	Wx	int32s
5	16, 17, 18, 19	Wy	int32s
6	20, 21, 22, 23	Wz	int32s
7	24, 25	Tax	int16s
8	26, 27	Tay	int16s
9	28, 29	Taz	int16s
10	30, 31	Twx	int16s
11	32, 33	Twy	int16s
12	34, 35	Twz	int16s
13	36, 37	S	int16s
14	38, 39	Timestamp	int16s
15	40	Status	int8u
16	41	Number	int8u

Таким образом, длина каждого пакета первичных данных должна быть равна 45 байтам (1 байт идентификатор, 42 байта данных и 2 байта CRC).

На вход программе поступает входной бинарный файл. Для каждого содержащегося в нем пакета первичных данных (ID = 0x87) требуется проверить его корректность (посчитать контрольную сумму и сравнить с суммой, записанной в этом пакете). Если пакет корректный, вывести значения всех параметров в 1 строку выходного текстового файла. Иначе признать пакет "битым" и не выводить информацию из него. В конце работы программы вывести статистику целых/битых/всего пакетов. Первая строка выходного текстового файла должна содержать «шапку» (наименования полей), вторая и последующие строки — значения.

Все остальные данные из входного файла (не относящиеся к пакетам первичных данных) игнорировать.

Необходимо воспользоваться технологией автоматного программирования, т.е. в программе должны быть явно выделены состояния.

### Пример программы для проверки контрольной суммы

```
const uint16_t crc16tab[256]= {
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
    0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
    0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
    0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
    0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
    0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
    0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
    0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
    0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
    0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
    0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
    0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
    0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
    0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
    0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
    0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
    0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
    0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
    0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
    0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
```

```

0xd94c,0xc96d,0xf90e,0xe92f,0x99c8,0x89e9,0xb98a,0xa9ab,
0x5844,0x4865,0x7806,0x6827,0x18c0,0x08e1,0x3882,0x28a3,
0xcb7d,0xdb5c,0xeb3f,0xfb1e,0x8bf9,0x9bd8,0xabbb,0xbb9a,
0x4a75,0x5a54,0x6a37,0x7a16,0x0af1,0x1ad0,0x2ab3,0x3a92,
0xfd2e,0xed0f,0xdd6c,0xcd4d,0xbdaa,0xad8b,0x9de8,0x8dc9,
0x7c26,0x6c07,0x5c64,0x4c45,0x3ca2,0x2c83,0x1ce0,0x0cc1,
0xef1f,0xff3e,0xcf5d,0xdf7c,0xaf9b,0xbfba,0x8fd9,0x9ff8,
0x6e17,0x7e36,0x4e55,0x5e74,0x2e93,0x3eb2,0x0ed1,0x1ef0
};

```

```

uint16_t crc16(const uint8_t *buf, int len)
{
    register int counter;
    register uint16_t crc = 0;
    for( counter = 0; counter < len; counter++)
        crc = (crc<<8) ^ crc16tab[((crc>>8) ^ *(char *)buf+
+)&0x00FF];
    return crc;
}

```

Дополнение к Заданию2 (выдается на паре).

Помимо пакетов первичных данных (ID = 0x87) во входном файле также содержатся пакеты сигнальных данных (ID = 0x98). В пакете сигнальных данных передается всего одно значение типа `int8u`. Контрольная сумма CRC для такого пакета не рассчитывается. Таким образом, пакет сигнальных данных имеет следующую структуру:

<SYNC><SYNC><LEN><ID> [данные]

где

- <SYNC> - 1 байт синхронизации (0xAA)
- <LEN> - 1 байт, длина пакета (включаются 1 байт идентификатор и 1 байт данных, то есть для пакета сигнальных данных <LEN> = 2);
- <ID> - 1 байт, идентификатор пакета, =0x98
- [данные] – 1 байт, значение типа `int8u`.

Данные, содержащиеся в пакетах сигнальных данных, необходимо вывести в отдельный текстовый файл *signals.txt* (по одному значению в строке).

Если при <ID>=0x98 длина пакета <LEN>≠2, такой пакет игнорировать.