1. Разработать контейнер «динамический массив» – *TArray*, который:

* содержит элементы типа Pointer / Object / заданного типа данных,
* увеличивает свой размер при добавлении в него элементов,
* предоставляет типовые свойства (текущее количество элементов, количество зарезервированных элементов),
* предоставляет типовые процедуры, применяемые при работе с массивом (вставка по индексу, удаление по индексу, очистка, копирование в другой массив, поиск индекса элемента по ссылке, сортировка),
* при обращении к элементу массива проверяет, что индекс находится в допустимых пределах.

Продемонстрировать использование класса.

1. Разработать контейнер «коллекция» — *TCollection*, который:

* содержит элементы типа *TCollectionItem*, содержащие ссылку на своего владельца,
* увеличивает свой размер при добавлении в него элементов,
* предоставляет типовые свойства (текущее количество элементов, количество зарезервированных элементов),
* предоставляет типовые процедуры, применяемые при работе с коллекцией (вставка, удаление, очистка, копирование в другую коллекцию, поиск индекса элемента по ссылке, сортировка),
* предоставляет типовые события, происходящие при работе с коллекцией (вставка, удаление, очистка),
* при обращении к элементу коллекции проверяет, что индекс находится в допустимых пределах.

Продемонстрировать использование класса.

1. Разработать контейнер «список» — *TList*, который:

* содержит элементы типа Pointer / Object / заданного типа данных,
* предоставляет типовые свойства (первый элемент, следующий элемент),
* предоставляет типовые процедуры, применяемые при работе со списком (вставка элемента, удаление элемента, очистка списка, замена элемента, перемещение списка в другой список),
* предоставляет типовые события, происходящие при работе со списком (вставка, удаление, очистка),

Продемонстрировать использование класса.

1. Разработать контейнер «стек» — *TStack*, который [эмулирует стек процессора, и]:

* содержит элементы заданного типа данных / [произвольные данные, представляемые в виде последовательности байтов],
* предоставляет типовые свойства (указатель на вершину стека, [указатель на начало «кадра»,] указатель на дно стека),
* предоставляет типовые процедуры, применяемые при работе со стеком (загрузка элемента на стек, выталкивание элемента из стека, очистка стека / [создание нового «кадра», выталкивание «кадра», загрузка данных заданного размера, выталкивание данных заданного размера]),
* предоставляет типовые события, происходящие при работе со стеком (загрузка в стек, выталкивание из стека),

Продемонстрировать использование класса.

1. Разработать класс «словарь» — *TDictionary*, который представляет собой набор ключей и их значений, и который:

* содержит ключи типа string и значения типа Pointer / заданного в шаблоне типа данных,
* увеличивает свой размер при добавлении в него элементов,
* предоставляет типовые свойства (количество ключей и значений, признак сортированного «словаря»),
* предоставляет типовые процедуры (вставка ключа и значения, удаление значения по ключу, поиск значения по ключу, очистка, сортировка),
* предоставляет типовые события, происходящие при работе со «словарем» (вставка, удаление, очистка).

Продемонстрировать использование класса.

1. Разработать класс «XML-дерево» — *TXmlTree* — для представления XML-файла в памяти. Класс:

* содержит список тегов и значений типа string, а также список вложенных деревьев типа TXmlTree,
* обеспечивает загрузку и выгрузку XML-файла (без детализации),
* предоставляет типовые свойства (количество ключей и значений, количество вложенных деревьев),
* предоставляет типовые процедуры (вставка ключа и значения в заданную позицию, удаление значения по ключу, поиск значения по ключу или на основе «пути» по аналогии с файловой системой, очистка),
* предоставляет типовые события, происходящие при работе с контейнером (вставка, удаление, очистка),

Продемонстрировать использование класса.

1. Разработать абстракцию файла *TAbstractFile* и два производных класса *TOSFile* (файл операционной системы) и *TMemoryFile* (файл в памяти). Классы:

* предоставляют типовые свойства (размер файла, текущая позиция указателя чтения-записи в файле),
* предоставляют типовые процедуры работы с файлом (запись, чтение, перемещение позиции указателя чтения-записи в файле, копирование содержимого другого файла),
* предоставляет типовые события, происходящие при работе с файлом (чтение, запись).

Продемонстрировать использование классов так: программно создать файл на диске с любым содержимым, скопировать содержимое файла на диске в файл в памяти.

1. Разработать контейнер «битовый массив» — *TBitArray*, который

* содержит элементы типа Boolean так, что на каждый отводится лишь 1 бит,
* увеличивает свой размер при добавлении в него элементов,
* предоставляет типовые свойства (текущее количество элементов, количество зарезервированных элементов),
* предоставляет типовые процедуры, применяемые при работе с массивом (чтение и запись бита по индексу, расширение массива),
* при обращении к биту в массиве проверяет, что индекс находится в допустимых пределах.

Продемонстрировать использование класса.

1. Разработать класс для представления матрицы произвольной размерности – *TMatrix*. Класс:

* содержит элементы типа double,
* принимает размеры матрицы на вход конструктора,
* предоставляет типовые свойства (размеры матрицы, элемент на пересечении строки и столбца),
* предоставляет типовые процедуры, применяемые при работе с матрицами (умножение, транспонирование),

Продемонстрировать использование класса.

1. Разработать класс «кольцевой буфер» - TCircularBuffer. Класс:

* содержит байты данных,
* предоставляет типовые свойства (указатель чтения, указатель записи, количество данных в буфере),
* предоставляет типовые процедуры, применяемые при работе с кольцевым буфером (запись, чтение, наращивание размера буфера),
* генерирует исключительную ситуацию при переполнении буфера.

Продемонстрировать использование класса.