1. Определить класс Base для работы с целыми беззнаковыми числами, состоящими из N десятичных цифр.

Внутри класса число должно быть реализовано с помощью указателя на тип char. Число N и размер отведенной памяти задаются в конструкторе класса.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), сложения, вычитания, «, [], инкремент ++ и декремент −− (справа и слева, как умножение и деление на 10).

При сложении количество значащих десятичных цифр результата (N) может отличаться от N аргументов.

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию output как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – данные одного объекта (последовательность десятичных цифр без пробелов) без указания количества десятичных цифр. Для каждой строки исходного файла надо создать объект класса CData0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr.

2. Определить класс Base для работы со строками, длина которых хранится в самом классе (строки произвольной длины). Длина строки и размер отведенной памяти задаются в конструкторе класса. Внутри класса строка должна быть реализована с помощью указателя char *.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), присваивания обычной строки переменной типа Base (например, Base str; str="aaaa";), «, [], инкремент ++ и декремент −− (справа и слева: добавление символа ! в конец строки и обрезание последнего символа, если это возможно), сложения (конкатенация), умножения (слева и справа) строки на беззнаковое целое число (оно равносильно сложению строки с собой нужное число раз),

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – данные одного объекта (строка без указания длины). Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными из введенной строки. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr.

3. Определить класс Base для работы с квадратной матрицей целых чисел. Внутри класса матрица должна быть реализована с помощью указателя int **. Размер матрицы и отведенной под нее памяти задаются в конструкторе класса.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, [], инкремент ++ и декремент —— (справа и слева: добавление нулевой строки и нулевого столбца в начало матрицы (вычеркивания, если возможно, первой строки и первого столбца), сложения, вычитания матриц, умножения матрицы (слева и справа) на число. При сложении и вычитании размер результата – это минимум из размеров исходных матриц, лишние строки матрицы большего размера игнорируются.

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку (построчно), а во втором данную функцию определить как функцию вывода данных класса как квадратную матрицу. В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта, разделенные пробелами, т.е. элементы матрицы без указания размера. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта.

Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr.

4. Определить класс Base для работы со стеком целых чисел ограниченного размера. Глубина стека задаются в конструкторе класса. В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент −− (справа и слева, работают как push с 0 и pop), сложения (результатом является стек из содержимого обоих стеков, при этом глубины суммируется, использовать push и pop), функции push (добавление числа в стек, если это возможно, возвращает информацию об успешности) и pop (удаление вершины, если это возможно, с сохранением вытесненной вершины, возвращает информацию об успешности),

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные стека, разделенные пробелами без указания глубины, которая определяется как минимально возможная. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr. .

5. Определить класс Base для работы с приоритетной очередью целых чисел ограниченного размера. Глубина очереди задаются в конструкторе класса. В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, декремент —— (справа и слева, работает как pop), сложения с целым числом (работает как push), функции push (добавление числа в очередь, при этом максимальное число должно встать в голову очереди) и pop (удаление вершины, если это возможно, с сохранением вытесненной вершины, возвращает информацию об успешности),

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта, разделенные пробелами без указания глубины, которая определяется как минимально возможная. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr(обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение с числом объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr.

6. Определить класс Base для работы с битовым множеством целых чисел в диапазоне от 0 до N. N и размер отведенной памяти задаются в конструкторе класса. Внутри класса множество должно быть реализовано с помощью указателя. Принадлежность числа множеству означает, что бит, соответствующей этому числу, равен 1, в случае нулевого бита число не принадлежит множеству.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент —— (справа и слева), увеличивающий и уменьшающий (если это возможно) верхнюю границу диапазона, сложения (объединение множеств), вычитания (пересечения). При сложении верхняя граница диапазона результата равна максимуму из соответствующих границ слагаемых, при вычитании минимуму.

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта (диапазон и числа множества в любом порядке), разделенные пробелами. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию вправо внутри массива arr.

7. Определить класс Base для работы с (кольцевой) очередью целых чисел ограниченного размера. Максимальный размер очереди задается в конструкторе класса. В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент −− (справа и слева, работают как push 0 и pop), сложения (результатом является очередь из содержимого обеих очередей, при этом глубины суммируется, использовать push и pop), функции push (добавление числа в очередь, если это возможно, возвращает информацию об успешности) и pop (удаление вершины, если это возможно, с сохранением вытесненной вершины, возвращает информацию об успешности). При сложении исходные очереди не должен изменяться.

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта, разделенные пробелами без указания глубины, которая определяется как минимально возможная. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f);, где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию влево внутри массива arr.

8. Определить класс Base для работы с мультимножеством целых чисел в диапазоне от N_1 до N_2, диапазон и размер отведенной памяти задаются в конструкторе класса. В качестве данных класса использовать массив целых чисел, значение элемента которого равно количеству числа в мультимножестве. В классе должны быть определены необходимые конструкторы, (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент −− (справа и слева, результат – увеличение или уменьшение одного из диапазонов), операторы сложения (объединение множеств), умножения (пересечения), при этом диапазоны исходных множеств могут быть различны.

В этом классе должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I=1, FileName – имя выходного файла, Data – все данные одного объекта, разделенные пробелами: начало и конец диапазона и (в любом порядке) числа, входящее в мультимножество. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию влево внутри массива arr.

9. Определить класс Base для работы с массивом строк (текстом). Количество строк текста задаются в конструкторе класса.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент - - (справа и слева, дописывающие в конец всех строк массива символ ! и удаляющие последний символ в каждой строке массива, если это возможно), сложения (конкатенация, или склейка строк соответствующих строк слагаемых).

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса как текст. В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта (без указания размера массива): строки текста, разделенные пробелами. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию влево внутри массива arr.

10. Определить классы Base для работы с массивом векторов на плоскости и CAngl для работы с массивом углов между векторами. Длина массива и размер отведенной памяти задаются в конструкторах классов.

В классах должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, [], инкремент ++ и декремент −− (справа и слева), увеличивающие и уменьшающие (если это возможно) длину массива путем дублирования (вычеркивания) последнего элемента, операторы сложения (Base и CAngl, возвращающий Base, а также CAngl и Base, возвращающий Base), вычитания (Base из Base, возвращающий CAngl).

В классе Base должна быть создана [чисто] виртуальная функция вывода данных класса в файл вида

virtual int output()=0;

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса в столбец (т.е. по одному элементу данных в одну строку). В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта, разделенные пробелами (без указания размера массива).

Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать нахождение знакопеременной суммы объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию влево внутри массива arr.

11. Определить класс Base для работы с массивом строк (текстом). Количество строк текста задаются в конструкторе класса.

В классе должны быть определены необходимые конструкторы (в том числе копирования, перемещения и загрузки из строки), деструктор, операторы присваивания (копированием и перемещением), «, инкремент ++ и декремент - - (справа и слева, дописывающие в конец всех строк массива символ ! и удаляющие последний символ в каждой строке массива, если это возможно), сложения (конкатенация, или склейка строк соответствующих строк слагаемых).

От класса Base надо породить два класса Child0 и Child1, в первом из которых переопределить функцию int output() как функцию вывода данных класса в файл в одну строку, а во втором данную функцию определить как функцию вывода данных класса как текст. В файле с исходными данными в каждой строке задаются данные для одного экземпляра класса, порожденного от Base. Данные задачи задаются в виде:

I FileName Data

где I = 0 или I= 1, FileName – имя выходного файла, Data – все данные одного объекта (без указания размера массива): строки текста, разделенные пробелами. Для каждой строки исходного файла надо создать объект класса Child0, если I == 0, и экземпляр класса Child1, если I == 1 и заполнить его данными Data. Имя выходного файла следует занести в соответствующее поле созданного объекта. Указатели на созданные объекты надо поместить в массив arr (обычных) указателей на базовый класс. Каждый новый объект должен создаваться функцией вида

Base *CreateData(const char *str, Factory **f), где f – массив фабрик для создания I-го дочернего класса от Base. Далее надо в цикле для каждого объекта из массива arr вызвать функцию output().

Также надо написать разумный тест на все реализованные функции класса. Тест, в частности, должен содержать сложение объектов, на которые указывают сохраненные в массиве arr указатели и циклический сдвиг объектов на одну позицию влево внутри массива arr.