ЛАБОРАТОРНАЯ РАБОТА № 5
«Классы, свойства, индексаторы»

Цель работы: получить навыки создания простейших классов с использованием принципа инкапсуляции, свойств, индексаторов, перегрузки операторов.

Лабораторное задание

Спроектировать класс согласно варианту. Продемонстрировать работу класса в консольном приложении. В программе должна выполняться проверка всех разработанных элементов класса.

Вариант 1

Создать класс Point, содержащий следующие члены класса:

– поля: int x, y;

– конструкторы, позволяющие создать экземпляр класса с нулевыми координатами, с заданными координатами;

– методы, позволяющие вывести координаты точки на экран, рассчитать

расстояние от начала координат до точки, переместить точку на плоскости на вектор (a, b);

– свойства, позволяющие получить/установить координаты точки (доступное для чтения и записи), умножить координаты точки на скаляр (доступное только для записи);

– индексатор, позволяющий по индексу 0 обращаться к полю x, по индексу 1 – к полю y, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей x и y на 1; констант true и false – обращение к экземпляру класса дает значение true, если значение полей x и y совпадает, иначе false;

операции бинарный + – одновременно добавляет к полям x и y значение скаляра.

Вариант 2

Создать класс Triangle, содержащий следующие члены класса:

– поля: int a, b, c;

– конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

– методы, позволяющие вывести длины сторон треугольника на экран, рассчитать периметр треугольника, рассчитать площадь треугольника;

– свойства, позволяющие получить/установить длины сторон треугольника (доступное для чтения и записи), установить, существует ли треугольник с данными длинами сторон (доступное только для чтения);

– индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, по индексу 2 – к полю c, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a, b и c на 1; констант true и false – обращение к экземпляру класса дает значение true, если треугольник с заданными длинами сторон существует, иначе – false; операции * – одновременно умножает поля a, b и c на скаляр.

Вариант 3

Создать класс Rectangle, содержащий следующие члены класса:

– поля: int a, b;

– конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

– методы, позволяющие вывести длины сторон прямоугольника на экран, рассчитать периметр прямоугольника, рассчитать площадь прямоугольника;

– свойства, позволяющие получить/установить длины сторон прямоугольника (доступное для чтения и записи), установить, является ли данный прямоугольник квадратом (доступное только для чтения);

– индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a и b; констант true и false – обращение к экземпляру класса дает значение true, если прямоугольник с заданными длинами сторон является квадратом, иначе – false; операции * – одновременно умножает поля a и b на скаляр.

Вариант 4

Создать класс Money, содержащий следующие члены класса:

– поля: int first (номинал купюры); int second (количество купюр);

– конструктор, позволяющий создать экземпляр класса с заданными значениям полей;

– методы, позволяющие вывести номинал и количество купюр, определить, хватит ли денежных средств на покупку товара на сумму N рублей, определить, сколько штук товара стоимости n рублей можно купить на имеющиеся денежные средства;

– свойства, позволяющие получить/установить значение полей (доступное для чтения и записи), рассчитать сумму денег (доступное только для чтения);

– индексатор, позволяющий по индексу 0 обращаться к полю first, по индексу 1 – к полю second, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей first и second; операции ! – возвращает значение true, если поле second не нулевое, иначе false; операции бинарный + – добавляет к значению поля second значение скаляра.

Вариант 5

Создать класс для работы с одномерным массивом целых чисел. Разработать следующие члены класса:

– поля: int [] IntArray;

– конструктор, позволяющий создать массив размерности n;

– методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы массива в порядке возрастания;

– свойство, возвращающее размерность массива (доступное только для чтения), и свойство, позволяющее умножить все элементы массива на скаляр (доступное только для записи);

– индексатор, позволяющий по индексу обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; операции ! – возвращает значение true, если элементы массива не упорядочены по возрастанию, иначе – false; операции бинарный * – умножить все элементы массива на скаляр; операции преобразования класса массив в одномерный массив (и наоборот).

Вариант 6

Создать класс для работы с двумерным массивом целых чисел. Разработать следующие члены класса:

– поля: int [,] intArray;

– конструктор, позволяющий создать массив размерности n×m;

– методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, вычислить сумму элементов i-го столбца;

– свойства, позволяющие вычислить количество нулевых элементов в массиве (доступное только для чтения), установить значение всех элементов главной диагонали массива, равное скаляру (доступное только для записи);

– двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если двумерный массив является квадратным; операции бинарный + – сложить два массива соответствующих размерностей; операции преобразования класса массив в двумерный массив (и наоборот).

Вариант 7

Создать класс для работы с двумерным массивом вещественных чисел. Разработать следующие функциональные члены класса:

– поля: double [][] doubelArray;

– конструктор, позволяющий создать ступенчатый массив;

– методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы каждой строки массива в порядке убывания;

– свойство, возвращающее общее количество элементов в массиве (доступное только для чтения), и свойство, позволяющее увеличить значение всех элементов массива на скаляр (доступное только для записи);

– двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если каждая строка массива упорядочена по возрастанию, иначе – false; операции преобразования класса массив в ступенчатый массив (и наоборот).

Вариант 8

Создать класс для работы со строками. Разработать следующие члены класса:

– поле: string line;

– конструктор, позволяющий создать строку на основе заданного строкового литерала;

– методы, позволяющие подсчитать количество цифр в строке, выводить на экран все символы строки, встречающиеся в ней ровно один раз, вывести на

– экран самую длинную последовательность повторяющихся символов в строке;

– свойство, возвращающее общее количество символов в строке (доступное только для чтения);

− индексатор, позволяющий по индексу обращаться к соответствующему символу строки (доступный только для чтения);

− перегрузку: операции унарного ! – возвращает значение true, если строка не пустая, иначе – false; констант true и false – обращение к экземпляру класса дает значение true, если строка является палиндромом, false – в противном случае; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования класса строка в тип string (и наоборот).

Вариант 9

Создать класс для работы со строками. Разработать следующие члены класса:

− поле: StringBuilder line;

− конструктор, позволяющий создать строку на основе заданного строкового литерала, и конструктор, позволяющий создавать пустую строку;

− методы, позволяющие подсчитать количество пробелов в строке, заменить в строке все прописные символы на строчные, удалить из строки все знаки препинания;

− свойство, возвращающее общее количество элементов в строке (доступное только для чтения), и свойство, позволяющее установить значение поля в соответствии с введенным значением строки с клавиатуры, а также получить значение данного поля (доступно для чтения и записи);

− индексатор, позволяющий по индексу обращаться к соответствующему символу строки;

− перегрузку: операции унарного + (-) – преобразующей строку к строчным (прописным) символам; констант true и false – обращение к экземпляру класса дает значение true, если строка не пустая, иначе – false; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования строки в тип StringBuilder (и наоборот).

Вариант 10

Самостоятельно изучите тип данных DateTime, на основе которого необходимо создать класс для работы с датой. Данный класс должен содержать следующие члены класса:

− поле DataTime data;

− конструкторы, позволяющие установить заданную дату, дату 1.01.2000;

− методы, позволяющие вычислить дату предыдущего дня, вычислить дату следующего дня, определить сколько дней осталось до конца месяца;

– свойства, позволяющие установить или получить значение поле класса (доступно для чтения и записи), определить, является ли год високосным (доступно только для чтения);

– индексатор, позволяющий определить дату i-го по счету дня относительно установленной даты (при отрицательных значениях индекса отсчет ведется в обратном порядке);

– перегрузку: операции ! – возвращает значение true, если установленная дата не является последним днем месяца, иначе – false; констант true и false – обращение к экземпляру класса дает значение true, если установленная дата является началом года, иначе – false; операции & – возвращает значение true, если поля двух объектов равны, иначе false.

Вариант 11

Создать класс Triangle, содержащий следующие члены класса:

– поля: int a, b, c;

– конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

– методы, позволяющие вывести длины сторон треугольника на экран, рассчитать периметр треугольника, рассчитать площадь треугольника;

– свойства, позволяющие получить/установить длины сторон треугольника (доступное для чтения и записи), установить, существует ли треугольник с данными длинами сторон (доступное только для чтения);

– индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, по индексу 2 – к полю c, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a, b и c на 1; констант true и false – обращение к экземпляру класса дает значение true, если треугольник с заданными длинами сторон существует, иначе – false; операции * – одновременно умножает поля a, b и c на скаляр.

Вариант 12

Создать класс Rectangle, содержащий следующие члены класса:

– поля: int a, b;

– конструктор, позволяющий создать экземпляр класса с заданными длинами сторон;

– методы, позволяющие вывести длины сторон прямоугольника на экран, рассчитать периметр прямоугольника, рассчитать площадь прямоугольника;

– свойства, позволяющие получить/установить длины сторон прямоугольника (доступное для чтения и записи), установить, является ли данный прямоугольник квадратом (доступное только для чтения);

– индексатор, позволяющий по индексу 0 обращаться к полю a, по индексу 1 – к полю b, при других значениях индекса выдается сообщение об ошибке;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение полей a и b; констант true и false – обращение к экземпляру класса дает

значение true, если прямоугольник с заданными длинами сторон является квадратом, иначе – false; операции * – одновременно умножает поля a и b на скаляр.

Вариант 13

Создать класс для работы с двумерным массивом целых чисел. Разработать следующие члены класса:

– поля: int [,] intArray;

– конструктор, позволяющий создать массив размерности n×m;

– методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, вычислить сумму элементов i-го столбца;

– свойства, позволяющие вычислить количество нулевых элементов в массиве (доступное только для чтения), установить значение всех элементов главной диагонали массива, равное скаляру (доступное только для записи);

– двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если двумерный массив является квадратным; операции бинарный + – сложить два массива соответствующих размерностей; операции преобразования класса массив в двумерный массив (и наоборот).

Вариант 14

Создать класс для работы с двумерным массивом вещественных чисел. Разработать следующие функциональные члены класса:

– поля: double [][] doubelArray;

– конструктор, позволяющий создать ступенчатый массив;

– методы, позволяющие ввести элементы массива с клавиатуры, вывести элементы массива на экран, отсортировать элементы каждой строки массива в порядке убывания;

– свойство, возвращающее общее количество элементов в массиве (доступное только для чтения), и свойство, позволяющее увеличить значение всех элементов массива на скаляр (доступное только для записи);

– двумерный индексатор, позволяющий обращаться к соответствующему элементу массива;

– перегрузку: операции ++ (--) – одновременно увеличивает (уменьшает) значение всех элементов массива на 1; констант true и false – обращение к экземпляру класса дает значение true, если каждая строка массива упорядочена по возрастанию, иначе – false; операции преобразования класса массив в ступенчатый массив (и наоборот).

Вариант 15

Создать класс для работы со строками. Разработать следующие члены класса:

– поле: StringBuilder line;

– конструктор, позволяющий создать строку на основе заданного строкового литерала, и конструктор, позволяющий создавать пустую строку;

– методы, позволяющие подсчитать количество пробелов в строке, заменить в строке все прописные символы на строчные, удалить из строки все знаки препинания;

– свойство, возвращающее общее количество элементов в строке (доступное только для чтения), и свойство, позволяющее установить значение поля в соответствии с введенным значением строки с клавиатуры, а также получить значение данного поля (доступно для чтения и записи);

– индексатор, позволяющий по индексу обращаться к соответствующему символу строки;

– перегрузку: операции унарного + (-) – преобразующей строку к строчным (прописным) символам; констант true и false – обращение к экземпляру класса дает значение true, если строка не пустая, иначе – false; операции & – возвращает значение true, если строковые поля двух объектов посимвольно равны (без учета регистра), иначе – false; операции преобразования строки в тип StringBuilder (и наоборот).