1.17. (\*) Даны различных целых 3 числа. Выбрать из этих чисел среднее по значению (например, для чисел 8, 5, 100 правильным ответом будет 8). Числа вводятся в произвольном порядке. Использовать методы Math.Min(...) и Math.Max(...), условный оператор использовать нельзя, только присваивания.

2.17. Дан номер некоторого года (положительное целое число). Вывести число дней в этом году, учитывая, что обычный год насчитывает 365 дней, а високосный — 366 дней. Високосным считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400 (например, годы 300, 1300 и 1900 не являются високосными, а 1200 и 2000 — являются).

3.34.

Условный оператор + функции + перечисления + декомпозиция задачи (разбиение на подзадачи) + формальный подход.

На изображниях ниже (для каждого варианта свое) приводится часть координатной плоскости (−10 ≤ x ≤ 10, −10 ≤ y ≤ 10) с графиками функций (парабола, линия) и фигурами (круг, прямоугольник). Размер одной клетки сетки – 1 (единица). Таким образом параметры функций и фигур можно однозначно определить из рисунка.

Области, на которые разбивается плоскость графиками и контурами фигур, закрашены разными цветами (белый, серый, желтый, оранжевый, голубой, зеленый). Необходимо реализовать программу, которая для точки (x, y) определяет цвет области, в которую данная точка попадает. Будем считать, что если точка (x, y) попадает на линию графика функции или контура фигуры, то правильным ответом будет цвет любой соседней области. Также стоит заметить, что некоторые комбинации линий предполагают их пересечение за пределами изображенного на рисунке фрагмента плоскости, поэтому, чтобы исключить неопределенность ответа, допустимыми значениями (x, y) являются только такие, которые попадают в приведенное изображение.

В программе должна быть реализована следующая функция (статический метод), которая для точки (x, y) возвращает цвет точки на изображении:

static SimpleColor GetColor(double x, double y) {

// реализовать самостоятельно

...

}

Как видно из заголовка функции, она должна возвращать значение типа SimpleColor – перечисления (enum) вида:

enum SimpleColor {

Black,

White,

Gray,

Red,

Orange,

Yellow,

Green,

Blue

}

На изображниях приводится часть координатной плоскости (−10 ≤ x ≤ 10, −10 ≤ y ≤ 10) с графиками функций (парабола, линия) и фигурами (круг,

прямоугольник). Размер одной клетки сетки – 1 (единица). Таким образом параметры функций и фигур можно однозначно определить из рисунка.

Области, на которые разбивается плоскость графиками и контурами фигур, закрашены разным цветом

Для декомпозиции и упрощений задачи необходимо реализовать вспомогательные функции (для некоторых вариантов только часть из них), которые будут использованы для реализации функции GetColor:

// проверяет, находится ли точка (x, y)

// выше линии y = a \* (x - x0) + y0

static bool IsPointAboveLine(double x, double y,

double x0, double y0, double a) {

return y > a \* (x - x0) + y0;

}

// проверяет, находится ли точка (x, y)

// выше пораболы y = a \* (x - x0) ^ 2 + y0

static bool IsPointAboveParabolaY(double x, double y,

double x0, double y0, double a) {

// реализовать самостоятельно

...

}

// проверяет, находится ли точка (x, y)

// справа от пораболы x = a \* (y - y0) ^ 2 + x0

static bool IsPointRightOfParabolaX(double x, double y,

double x0, double y0, double a) {

// реализовать самостоятельно

...

}

// проверяет, находится ли точка (x, y)

// внутри круга (x - x0) ^ 2 + (y - y0) ^ 2 = r ^ 2

static bool IsPointInCircle(double x, double y,

double x0, double y0, double r) {

// реализовать самостоятельно

...

}

// проверяет, находится ли точка (x, y)

// внутри прямоугольника с верхним левым углом (x0, y0)

// шириной w и высотой h

static bool IsPointInRect(double x, double y,

double x0, double y0, double w, double h) {

// реализовать самостоятельно

...

}

Ниже приведена реализация функции GetColor для варианта 1 данной задачи с использованием описанных выше вспомогательных функций.

static SimpleColor GetColor(double x, double y) {

if (IsPointRightOfParabolaX(x, y, 2, 2, 1))

return SimpleColor.Orange;

if (IsPointAboveLine(x, y, -3, 1, 2.5) &&

!IsPointAboveLine(x, y, 0, -2, 0))

return SimpleColor.Green;

if (!IsPointAboveLine(x, y, -3, 1, 2.5) &&

!IsPointAboveLine(x, y, 0, -2, 0))

return SimpleColor.Yellow;

return SimpleColor.Gray;

}

Можно (рекомендуется) еще сильнее декомпозировать приведенную выше реализацию, описав для каждого графика функции или фигуры отдельную функцию вычисления принадлежности той или иной области, например, для варианта 1 это будут функции (здесь графики и функции пронумерованы следующим образом: 1 – наклонная линия, 2 – параллельная оси X линия, 3 – парабола):

static bool IsPointAboveLine1(double x, double y) {

return IsPointAboveLine(x, y, -3, 1, 2.5);

}

static bool IsPointAboveLine2(double x, double y) {

return IsPointAboveLine(x, y, 0, -2, 0);

}

static bool IsPointRightOfParabolaX3(double x, double y) {

return IsPointRightOfParabolaX(x, y, 2, 2, 1);

}

И тогда реализация GetColor будет максимально простой:

static SimpleColor GetColor(double x, double y) {

if (IsPointRightOfParabolaX3(x, y))

return SimpleColor.Orange;

if (IsPointAboveLine1(x, y) && !IsPointAboveLine2(x, y))

return SimpleColor.Green;

if (!IsPointAboveLine1(x, y) && !IsPointAboveLine2(x, y))

return SimpleColor.Yellow;

return SimpleColor.Gray;

}

В методе Main необходимо для несколько заранее выбранных точек на рисунке (обязательно должны быть охвачены все области, на которые разбивается плоскость) распечатать результат, который возвращает GetColor

(и самостоятельно убедиться, что данная функция работает корректно). Также программа должна запрашивать координаты одной произвольной точки и для нее таким же образом печатать результат.

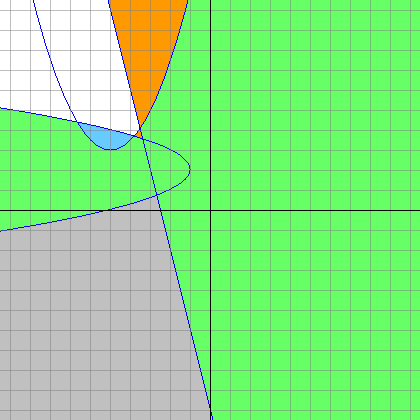
Для печати результата функции для точки (x, y) реализовать функцию:

static void PrintColorForPoint(double x, double y) {

// реализовать самостоятельно

...

}



5.17. Вложенные циклы. Простые алгоритмы.

Напечать в консоле следующие фигуры.

Для печати реализовать отдельныю функциюс параметрами.

Для различных вариантов могут быть заданы следующие параметры:

1. h – высота фигуры

2. w – ширина фигыры

3. s – размер фигуры

Для каждого варианта указаны допустимые значения данных параметров, а также в скобках приведены значения параметров для приведенного примера.

Запрещено использовать строки, массивы, а также управлять курсором, можно только использовать циклы, условный оператор, а также Console.Write(...) и Console.WriteLine(…)

s >= 2, s — четное (в примере ниже: s = 8)

/\

//\\

///\\\

////\\\\

\\\\////

\\\///

\\//

\/