В. И. Великодный

Задачи по программированию

Приднестровский государственный университет
им. Т. Г. Шевченко

Физико-математический факультет

Кафедра прикладной математики и информатики

В. И. Великодный

Задачи по программированию

Учебное пособие

Бендеры, 2017

УДК 681.3.06(075.8)
ББК 3973я73
В27

Р е ц е н з е н т ы:
Коровай А. В., зав. каф. ПМиИ ПГУ им. Т. Г. Шевченко, к.ф.-м.н.
Марков Д. А., нач. УИР ПГУ им. Т. Г. Шевченко, к.ф.-м.н.

В27 Великодный В. И. Задачи по программированию: Учебное пособие — Бендеры: РВТ, 2017. — 84 с.

Пособие содержит задачи для вводного курса программирования для направлений подготовки, связанных с разработкой программного обеспечения. Задачи охватывают широкий круг тем и парадигм: структурное, объектно-ориентированное, функциональное программирование. Хотя задачи не привязаны к конкретному языку программирования, предполагается, что они будут решаться на языке C#.

Пособие предназначено для студентов, обучающихся по направлениям «Прикладная математика и информатика», «Прикладная математика» и других.

УДК 681.3.06(075.8)
ББК 3973я73

Рекомендовано Научно-методическим советом ПГУ им. Т. Г. Шевченко

Введение

Единственный способ научиться программировать — это программировать. Только решая задачи, сложные или простые, можно получить необходимые навыки и опыт.

Существует большое количество сборников задач по программированию, но лишь небольшая их часть удовлетворяет требованиям современных университетских курсов. Настоящий сборник представляет собой попытку восполнить этот пробел. Была поставлена цель подобрать или составить такие задачи, чтобы они охватывали весь курс программирования, не только конструкции управления процессом выполнения. В частности, в сборнике есть задачи на применение современной технологии обработки данных LINQ, написание регулярных выражений, проектирование классов.

Сборник содержит 360 задач. Предполагается, что задачи будут решаться на языке программирования C#, но в большинстве случаев выбор языка программирования непринципиален. Практически все задачи предложены автором сборника, хотя в тривиальных случаях они и могут совпадать с известными задачами из других сборников. Некоторые темы задач практически не встречаются в литературе (например, задачи на интерфейсы в ООП или задачи на параллельное программирование). Большое количество идей и замечаний было предложено старшим преподавателем кафедры прикладной математики и информатики Е. В. Калинковой.

У некоторых задач намеренно оставлена нечёткая формулировка условий. Во-первых, это приближение к реальной работе, когда чёткие постановки задач — редкость. Во-вторых, это развивает навыки анализа задачи, умение отделить главное от второстепенного. Кроме того, студентам не навязываются искусственные ограничения, такие, как запрет на использование каких-либо конструкций языка при решении.

Некоторые задачи поучительны: рассказывают об интересных фактах, заставляют вспомнить результаты из смежных областей знаний, таких как математика и физика.

Почти каждую из задач можно решить «в лоб». Но во многих случаях (и это тоже сделано намеренно), первое очевидное решение не будет самым эффективным. Немного подумав, можно сократить его всего до нескольких команд.

Номер каждой задачи состоит из трёх частей: номеров темы и раздела, а также варианта. Всего вариантов 10 — от 0 до 9. Такой подход позволяет легко распределить задачи между студентами. Например, вариант можно назначить в соответствии с последней цифрой номера зачётной книжки или номера в списке группы.

Исходные тексты последней версии сборника находятся на веб-странице пособия — https://github.com/velikodniy/programming-assignments. О найденных опечатках или неточностях можно сообщить по адресу электронной почты vadim@velikodniy.name.

1. Линейные алгоритмы

1.1. Арифметические выражения

1.1-0 Написать программу для приближённого вычисления значения тангенса угла в радианах по формуле

$$\operatorname{tg} x \approx x + \frac{1}{3}x^3 + \frac{2}{15}x^5.$$

1.1-1 Велосипедист движется равномерно и прямолинейно со скоростью v $\frac{\text{м}}{\text{с}}$. Написать программу, вычисляющую время в минутах, за которое он пройдёт s км.

1.1-2 Даны два тела с массами m_1 и m_2, расположенные на расстоянии d друг от друга. Найти силу их взаимного притяжения. (Гравитационная постоянная $\gamma \approx 6{,}67 \times 10^{-11}$ $\frac{\text{м}^3}{\text{кг}\cdot\text{с}^2}$.)

1.1-3 Написать программу, вычисляющую сумму членов бесконечно убывающей геометрической прогрессии, первый член которой равен b, а знаменатель — q.

1.1-4 Тело подброшено вертикально с начальной скоростью v. Написать программу, вычисляющую высоту тела в момент времени t. (Ускорение свободного падения $g \approx 9{,}81$ $\frac{\text{м}}{\text{с}^2}$, сопротивлением воздуха пренебречь.)

1.1-5 Длина комнаты без окон равна a, ширина — b, высота — c. В комнате есть дверь площадью $w \times h$. (Все размеры даны в метрах.) Написать программу, вычисляющую общую площадь стен.

1.1-6 Написать программу, вычисляющую сумму первых n ($n > 3$) членов арифметической прогрессии, если известны первый член a_1 и третий a_3.

1.1-7 Автомобиль двигаясь прямолинейно и равноускоренно за t с проехал s м. Написать программу, вычисляющую его ускорение, если начальная скорость была нулевой.

1.1-8 На проводник длины L с током силой I действует магнитное поле. Вектор магнитной индукции перпендикулярен проводнику, и его модуль равен B. Написать программу, вычисляющую силу, действующую на проводник.

1.1-9 Дан прямоугольный треугольник с катетами a и b, лежащими на осях координат. Катет a лежит на оси абсцисс, а катет b — на оси ординат. Найти координаты его центра тяжести.

1.2. Математические функции

1.2-0 Написать программу, вычисляющую площадь кольца со внутренним и внешним радиусами, равными r и R соответственно.

1.2-1 Дан треугольник со сторонами a, b и c. Написать программу, вычисляющую угол между сторонами a и b.

1.2-2 Написать программу, вычисляющую расстояние между точками (x_1, y_1) и (x_2, y_2).

1.2-3 Дан треугольник со сторонами a, b и c. Написать программу, вычисляющую его площадь.

1.2-4 Написать программу, вычисляющую угол между векторами с координатами (a_x, a_y) и (b_x, b_y).

1.2-5 Дана длина окружности l. Найти площадь круга, ограниченного ей.

1.2-6 Даны катеты a и b прямоугольного треугольника. Найти радиус вписанной окружности.

1.2-7 Написать программу, находящую значение выражения $\left|\frac{x+y}{2}\sqrt{\frac{\ln(x+y)}{xy}}\right|$ для заданных x и y.

1.2-8 Написать программу, вычисляющую расстояние до линии горизонта от точки, расположенной на высоте h над поверхностью Земли. (Считать Землю идеальным шаром с радиусом $R = 6350$ км.)

1.2-9 Известно расстояние D между двумя наблюдательными пунктами и углы α_1 и α_2, под которыми с них видна цель. Углы определяются между направлением на цель и направлением на другой наблюдательный пункт. Написать программу, вычисляющую расстояния от цели до наблюдательных пунктов.

1.3. Целочисленная арифметика

1.3-0 Написать программу, меняющую местами две первые цифры с двумя последними в заданном четырёхзначном числе.

1.3-1 Дано пятизначное число. Составить программу, вычисляющую сумму цифр стоящих на нечётных позициях. Позиции нумеруются справа налево начиная с 1.

1.3-2 Дан объём информации I в байтах ($I < 2^{30}$). Написать программу, выражающую его через байты, килобайты и мегабайты. Например, 1234567 байт = 1 Мбайт 181 Кбайт 647 байт.

1.3-3 Написать программу, определяющую, сколько полных оборотов сделают часовая и минутная стрелки за указанное количество секунд.

1.3-4 Написать программу, вычисляющую целую часть от деления числа, составленного из первых трёх цифр заданного пятизначного числа, на число, составленное из оставшихся двух цифр.

1.3-5 Написать программу, находящую сумму и произведение цифр четырёхзначного числа.

1.3-6 Дано целое число k ($0 \leqslant k \leqslant 26$). Вывести его представление в троичной системе.

1.3-7 Дано четырёхзначное число. Его цифры суммируются, после чего эта операция применяется к результату, пока не останется одна цифра. Написать программу, находящую эту цифру.

1.3-8 Сколько кубиков с длиной ребра h можно поместить в коробку с размерами $a \times b \times c$. (Все размеры — целые числа.)

1.3-9 С начала суток до некоторого момента прошло s с. Написать программу, выводящую время в этот момент с точностью до минут.

1.4. Операции ввода и вывода

1.4-0 Дан угол в радианах. Написать программу, переводящую его в градусы, минуты и секунды (градусы и минуты — целые числа). Ответ вывести в виде: «Угол a рад равен $d^\circ\ m'\ s''$». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку. (Код символа градуса в Unicode — 00B0.)

1.4-1 Даны числа a, b и c. Написать программу, вычисляющее их среднее гармоническое $g = \frac{3}{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}}$. Ответ вывести в виде: «Среднее гармоническое чисел a, b и c равно g.». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-2 Дан угол в градусах. Написать программу, переводящую его в деления угломера (единица измерения углов, принятая в артиллерии). 6000 делений угломера составляют 360°. Последние две цифры обычно отделяются дефисом. Например, 705 делений угломера записываются как «7-05». Ответ вывести в виде: «Угол d° равен a-b.». Вместо буквенных обозначений должны стоять конкретные целые числа. Перед запросом ввода с клавиатуры выводить подсказку. (Код символа градуса в Unicode — 00B0.)

1.4-3 Даны два точечных заряда q_1 и q_2, расположенные на расстоянии d друг от друга. Написать программу, вычисляющую силу их взаимного притяжения. Коэффициент пропорциональности $k = \frac{1}{4\pi\varepsilon_0}$, где $\varepsilon_0 \approx 8{,}85 \times 10^{-12}\ \frac{\Ф}{\text{м}}$. Ответ вывести в виде: «Сила притяжения между зарядами q_1 Кл

и q_2 Кл, находящихся на расстоянии d м, равна F Н.». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-4 Написать программу, вычисляющую площадь правильного n-угольника, вписанного в окружность радиуса R. Ответ вывести в виде: «Площадь правильного n-угольника, вписанного в окружность радиуса R, равна S.». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-5 Написать программу, находящую для некоторого x приближённое значение $\sin x$ по формуле $\sin x \approx x - \frac{x^3}{6}$ и вычисляющую абсолютную погрешность результата. Ответ вывести в виде: «Значение $\sin x$ приближённо равно y (погрешность — e).». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 4-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-6 Написать программу, вычисляющую для некоторого момента времени, заданного часами и минутами (целые числа), угол в градусах между часовой и минутной стрелками. Ответ вывести в виде: «В момент h:m угол между стрелками равен d.». Вместо буквенных обозначений должны стоять конкретные целые числа. Перед запросом ввода с клавиатуры выводить подсказку. (Код символа градуса в Unicode — 00B0.)

1.4-7 Написать программу вычисляющую сумму вклада через n лет при p % годовых с капитализацией процентов, если первоначальный вклад был равен S_0 руб. Капитализация процентов означает, что величина, на которую ежегодно увеличивается вклад, вычисляется на основе текущей суммы вклада. Ответ вывести в виде: «При первоначальном вкладе S_0 руб. и p % годовых с капитализацией процентов сумма вклада составит S_n руб..». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-8 Написать программу, переводящую вес в килограммах в фунты и унции. 1 фунт ≈ 0,454 кг, при этом в 1 фунте 16 унций. Ответ с точностью до унций вывести в виде: «m кг ≈ p фунтов o унций». Вместо буквенных обозначений должны стоять конкретные целые числа. Перед запросом ввода с клавиатуры выводить подсказку.

1.4-9 Написать программу, переводящую сумму в рублях в евро для указанного обменного курса. Ответ вывести в виде: «R руб. = E евро.». Вместо буквенных обозначений должны стоять конкретные числа с точностью до 2-го знака после запятой. Перед запросом ввода с клавиатуры выводить подсказку.

2. Алгоритмы с ветвлением

2.1. Условный оператор

2.1-0 Три точки на плоскости заданы своими координатами. Написать программу, определяющую, лежат ли они на одной прямой.

2.1-1 Дан шестизначный номер билета. Написать программу, проверяющую, является ли билет «счастливым». (Билет будем считать «счастливым», если сумма первых трёх цифр равна сумме последних трёх цифр.)

2.1-2 Стоимость минуты разговора по телефону — p руб. Если продолжительность разговора превышает 5 минут, то на оставшуюся часть времени действует скидка, равная 20 %. Написать программу, определяющую стоимость разговора заданной продолжительности t.

2.1-3 Написать программу, проверяющую, равно ли утроенное произведение цифр заданного двузначного числа ему самому. (Например, число 15 удовлетворяет этому условию).

2.1-4 Написать программу, определяющую, является ли заданное четырёхзначное число палиндромом. (Число-палиндром — это число, запись которого слева направо совпадает с записью справа налево.)

2.1-5 Даны радиус окружности и катеты прямоугольного треугольника. Написать программу определяющую, можно ли вписать треугольник в окружность.

2.1-6 Написать программу, вычисляющую для заданного n значение функции

$$f(n) = \begin{cases} \frac{n}{2}, & \text{если } n \text{ чётное;} \\ 3n+1, & \text{если } n \text{ нечётное.} \end{cases}$$

2.1-7 Дан прямоугольник размерами $w \times h$. Написать программу, определяющую, можно ли полностью накрыть его n плитками размера $a \times a$. (Все размеры — целые числа.)

2.1-8 Снаряд выпущен под углом α к горизонту с начальной скоростью v. Написать программу, проверяющую, попадёт ли он в цель высотой h, находящуюся на расстоянии L от пушки. (Ускорение свободного падения $g \approx 9{,}81\ \frac{\text{м}}{\text{с}^2}$, сопротивлением воздуха пренебречь.)

2.1-9 Дано целое число k ($1 \leqslant k \leqslant 365$). Написать программу, определяющую, придётся ли k-й день года на воскресенье, если 1 января — понедельник.

2.2. Составные условия

2.2-0 Даны две стороны треугольника и угол между ними. Составить программу, определяющую, является ли треугольник равносторонним.

2.2-1 Написать программу, определяющую, является ли указанный год високосным. (Год не является високосным, если его номер не кратен 4, либо кратен 100, но при этом не кратен 400.)

2.2-2 Даны два целых числа. Написать программу, определяющую, является ли наименьшее из них чётным.

2.2-3 Написать программу, определяющую, можно ли из отрезков с длинами a, b и c составить треугольник.

2.2-4 Написать программу, определяющую, расположены ли цифры заданного четырёхзначного числа в нём по возрастанию слева направо.

2.2-5 Прямоугольник со сторонами, параллельными осям координат, задан координатами левого верхнего и нижнего правого углов. Написать программу, проверяющую, находится ли точка (x, y) внутри прямоугольника.

2.2-6 Дано трёхзначное число. Написать программу, определяющую, содержит ли оно нечётные цифры.

2.2-7 Даны два отрезка числовой прямой. Написать программу, определяющую, пересекаются ли они.

2.2-8 На плоскости даны точки $A(x_A, y_A)$, $B(x_B, y_B)$ и $C(x_C, y_C)$. Написать, программу, определяющую расстояние от точки (x, y) до ближайшей из указанных.

2.2-9 Написать программу, проверяющую, можно ли вставить стержень с сечением в виде прямоугольника с размерами $w \times h$ в прямоугольное отверстие с размерами $W \times H$.

2.3. Несколько условий или оператор выбора

2.3-0 Написать программу, выводящую три заданных числа в порядке возрастания.

2.3-1 В старояпонском календаре был принят 12-летний цикл, в котором года носили названия животных: крыса, корова, тигр, заяц, дракон, змея, лошадь, овца, обезьяна, курица, собака и свинья. Например, 1988 год был годом дракона. Написать программу, определяющую животное, соответствующее указанному году.

2.3-2 Написать программу, решающую уравнение вида $ax^2+bx+c=0$ по заданным коэффициентам. Учесть случай, когда $a=0$.

2.3-3 Написать программу, находящую медиану трёх заданных чисел. (Медиана — это число, расположенное посередине упорядоченного списка.)

2.3-4 Даны три стороны треугольника. Написать программу, определяющую, является ли он остроугольным, тупоугольным, прямоугольным, равнобедренным или равносторонним. (Треугольник может относиться к нескольким классам одновременно.)

2.3-5 Дано целое число k ($1 \leqslant k \leqslant 365$). Написать программу, определяющую день недели k-го дня года, если 1 января — понедельник.

2.3-6 Дано трёхзначное десятичное число, запись которого не содержит нулей. Написать программу, определяющую, сколько различных чисел можно составить из цифр этого числа, используя каждую цифру один раз. (Например, из цифр числа 112 можно составить три числа: 112, 121 и 211.)

2.3-7 Написать программу, определяющую количество решений системы линейных уравнений с двумя неизвестными.

2.3-8 Две окружности заданы координатами центров и радиусами. Написать программу, определяющую, пересекаются ли они в точности в одной точке.

2.3-9 Дано целое число n — расстояние в метрах. Написать программу, выводящую на экран это расстояние с указанием единицы измерения в нужной форме. Например: «31 метр», «52 метра», «15 метров».

3. Циклы

3.1. Цикл с параметром

3.1-0 На счёте в банке лежит S руб. Через каждый месяц размер вклада увеличивается на p %. Написать программу, выводящую таблицу с суммой вклада в каждом из следующих 12 месяцев.

3.1-1 Последовательность чисел $a_0, a_1, a_2, \ldots$ образуется по закону:

$$a_0 = 1,$$
$$a_k = ka_{k-1} + \frac{1}{k}.$$

Написать программу, вычисляющую a_n для заданного номера n.

3.1-2 Начав тренировки спортсмен в первый день пробежал L км. Каждый следующий день он увеличивал пробег на p % от пробега предыдущего дня. Написать программу, определяющую суммарный пробег за N дней.

3.1-3 Последовательность Фибоначчи определяется следующим образом:

$$a_0 = 1,$$
$$a_1 = 1,$$
$$a_k = a_{k-1} + a_{k-2}.$$

Написать программу, выводящую первые n членов последовательности.

3.1-4 Дано натуральное число n. Написать программу, вычисляющую сумму

$$n^2 + (n+1)^2 + \ldots + (2n)^2.$$

3.1-5 Написать программу, вычисляющую частичную сумму гармонического ряда

$$1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}$$

для заданного n.

3.1-6 Написать программу, вычисляющую сумму

$$\frac{1}{2} + \frac{2}{3} + \ldots + \frac{n}{n+1}$$

для заданного n.

3.1-7 Известно сопротивление каждого из элементов электрической цепи. Все элементы соединены параллельно. Написать программу, вычисляющую общее сопротивление цепи.

3.1-8 Двойной факториал числа определяется следующим образом:

$$n!! = \begin{cases} 1 \cdot 3 \cdot 5 \cdot \ldots \cdot n, & \text{если } n \text{ нечётное,} \\ 2 \cdot 4 \cdot 6 \cdot \ldots \cdot n, & \text{если } n \text{ чётное.} \end{cases}$$

Написать программу, вычисляющую двойной факториал для заданного натурального n.

3.1-9 Написать программу, вычисляющую для заданного n значение суммы

$$1 + \frac{1}{1!} + \frac{1}{2!} + \ldots + \frac{1}{n!}.$$

3.2. Цикл с условием

3.2-0 Написать программу, находящую наибольший общий делитель двух чисел с помощью алгоритма Евклида.

3.2-1 Дано действительное число a. Написать программу, находящую такое наименьшее число n, что

$$1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n} > a.$$

3.2-2 В 50-х годах XX века был предложен (к сожалению, неэффективный) метод Фибоначчи получения последовательности псевдослучайных чисел. Последовательность задаётся следующими формулами:

$$\begin{aligned} X_0 &= 1; \\ X_1 &= 1; \\ X_k &\equiv X_{k-1} + X_{k-2} \pmod{m}, \end{aligned}$$

где m — некоторое натуральное число.

Эта последовательность повторяется с определённым периодом. Например, для $m = 3$ последовательность имеет вид:

$$1, 1, 2, 0, 2, 2, 1, 0, 1, 1, \ldots,$$

то есть период равен 8.

Написать программу, определяющую для заданного m длину периода последовательности.

3.2-3 Приближённое решение уравнения $\cos x = x$ можно найти как предел последовательности, заданной рекуррентным соотношением:

$$\begin{aligned} a_0 &= 1, \\ a_k &= \cos a_{k-1}. \end{aligned}$$

Дано малое действительное число $\varepsilon > 0$. Написать программу, находящую первый член последовательности, для которого $|a_k - \cos a_k| < \varepsilon$.

Этот метод называется методом простых итераций.

3.2-4 Написать программу, проверяющую, является ли введённое значение факториалом некоторого числа.

3.2-5 Дана последовательность чисел $a_1, a_2, \ldots, a_n$. Написать программу, проверяющую, является ли последовательность упорядоченной по возрастанию. Если это не так, то вывести номер первого числа, нарушающего упорядоченность.

3.2-6 Написать программу, находящую наименьший делитель заданного числа, отличный от 1.

3.2-7 Написать программу, находящую сумму заданного натурального числа n и числа, полученного записью цифр числа n в обратном порядке.

3.2-8 Дано натуральное число n и цифра k. Написать программу, находящую номер первого вхождения цифры в число. (Позиции отсчитываются справа налево начиная с 0.)

3.2-9 Дано натуральное число. Написать программу, проверяющую является ли она палидромом. (Число называется палидромом, если оно записывается одинаково слева направо и справа налево.)

3.3. Проверка условия внутри цикла

3.3-0 Написать программу, находящую наибольшую и наименьшую цифры заданного натурального числа.

3.3-1 Число называется автоморфным, если оно равно последним цифрам своего квадрата. Например, $5^2 = 25$. Написать программу, находящую все трёхзначные автоморфные числа.

3.3-2 Написать программу, находящую все двузначные числа, сумма квадратов цифр которых кратна 13.

3.3-3 Написать программу, определяющую сколько раз цифра k входит в десятичную запись заданного числа n.

3.3-4 Так называемая сиракузская последовательность чисел для заданного числа n строится следующим образом. Если n чётное, то оно делится на 2, иначе заменяется на $3n + 1$. Затем действия повторяются над полученным числом.

Например, если $n = 5$, то получаем последовательность

$$5, 16, 8, 4, 2, 1.$$

Гипотеза Коллатца состоит в том, что сиракузская последовательность через конечное число шагов всегда приходит к единице.

Напишите программу, определяющую, за сколько шагов последовательность достигнет единицы для заданного начального числа n.

3.3-5 Написать программу, удаляющую из записи числа все чётные цифры. Например, из числа 12345 должно получиться 135.

3.3-6 Написать программу, проверяющую для заданного числа, упорядочены ли его цифры по возрастанию справа налево.

3.3-7 Написать программу, находящую количество делителей заданного числа n.

3.3-8 Дано натуральное число n и действительные числа $a_1, a_2, \ldots, a_n$. Написать программу, вычисляющую

$$\max_{1 \leqslant i \leqslant n} |a_i| .$$

3.3-9 Написать программу, находящую все двузначные числа, равные утроенному произведению своих цифр.

3.4. Вложенные циклы

3.4-0 Написать программу, находящую первые 100 простых чисел.

3.4-1 Некоторая последовательность начинается с натурального числа, кратного 3. Каждый следующий член равен сумме кубов цифр предыдущего. Известно, что любая такая последовательность начиная с некоторого члена становится постоянной, и её члены равны 153. Написать программу, находящую количество членов, не равных 153 для указанного первого члена.

3.4-2 Написать программу, находящую все натуральные корни уравнения

$$a^2 + b^2 = c^2,$$

где $a, b, c \in \{1, 2, 3, \ldots, 10\}$.

3.4-3 Дано натуральное число n. Написать программу, выводящую его разложение на простые множители.

3.4-4 Даны натуральные числа m и n. Написать программу, находящую все натуральные числа, меньшие n, квадрат суммы цифр которых равен m.

3.4-5 Натуральное n-значное число называется числом Армстронга, если сумма его цифр, возведённых в n-ю степень, равна самому числу. Например, $153 = 1^3 + 5^3 + 3^3$. Написать программу, находящую все трёхзначные числа Армстронга.

3.4-6 Дано число n. Написать программу, находящую число с наибольшей суммой делителей, не превосходящее n.

3.4-7 Как показал индийский математик С. Рамануджан, 1729 — наименьшее число, представимое двумя различными способами в виде суммы кубов двух натуральных чисел x и y ($x \geqslant y \geqslant 0$). То есть, $x^3 + y^3 = 1729$. Написать программу, находящую обе пары x и y.

3.4-8 Два натуральных числа называются дружественными, если каждое из них равно сумме всех делителей другого (само число в качестве делителя не рассматривается). Написать программу, находящую все пары дружественных чисел, меньших 50000.

3.4-9 Число называется совершенным, если оно равно сумме своих делителей за исключением его самого. Например, делителями числа 28 являются следующие числа: $1, 2, 4, 7, 14, 28$. При этом $28 = 1 + 2 + 4 + 7 + 14$, то есть 28 — совершенное число. Древним грекам были известны только первые 4 совершенных числа. Написать программу, находящую их.

4. Функции

4.1. Функции

4.1-0 Даны координаты вершин двух треугольников. Написать программу, проверяющую, лежит ли один треугольник внутри другого. (Определить функцию для проверки, находится ли точка внутри треугольника.)

4.1-1 Даны n натуральных чисел. Написать программу, находящую их общий делитель. (Определить функцию для расчёта наибольшего общего делителя двух чисел.)

4.1-2 Даны три квадратных уравнения:

$$ax^2 + bx + c = 0,$$
$$bx^2 + ax + c = 0,$$
$$cx^2 + ax + b = 0,$$

где $a, b, c \neq 0$. Написать программу, определяющую, сколько из них имеют действительные корни. (Определить функцию, позволяющую распознавать наличие вещественных корней в квадратном уравнении.)

4.1-3 Написать программу для вычисления биномиального коэффициента

$$C_n^k = \frac{n!}{k!(n-k)!}$$

для заданных n и k. (Определить функцию для вычисления факториала числа.)

4.1-4 Два простых числа называются «близнецами», если модуль их разности равен 2 (например, 41 и 43 — «близнецы»). Написать программу, находящую все числа-близнецы, не превышающие 200. (Определить функцию для распознавания простых чисел.)

4.1-5 Написать программу, находящую периметр треугольника по координатам его вершин. (Определить функцию для расчёта длины отрезка по координатам его концов.)

4.1-6 Три прямые заданы в виде уравнений вида $ax + by + c = 0$. Написать программу, проверяющую, образуют ли они треугольник. (Определить функцию, проверяющую, пересекаются ли две прямые ровно в одной точке.)

Случай, когда три прямые пересекаются в одной точке отдельно рассматривать не требуется.

4.1-7 Три вектора на плоскости заданы своими координатами. Написать программу, находящую пару векторов, образующих наименьший угол. (Определить функцию, вычисляющую угол между векторами.)

4.1-8 Написать программу, находящую все трёхзначные числа, у которых ровно k делителей. (Определить функцию, находящую количество делителей числа.)

4.1-9 Написать программу, находящую корни системы из двух линейных уравнений с двумя неизвестными по заданным коэффициентам. (Определить функцию, вычисляющую определитель матрицы 2×2.)

4.2. Функции высшего порядка

4.2-0 Определить функцию, вычисляющую значение выражения

$$\sum_{m=1}^{n}\left(f(m)\prod_{\substack{k=1,\\ k\neq m}}^{n}\frac{k}{k-m}\right)$$

для заданного натурального числа n и функции f. Написать программу, использующую эту функцию.

Рассмотренное выражение — значение в нуле интерполяционного многочлена Лагранжа. Иными словами, программа будет по значениям $f(1), f(2), \ldots, f(n)$ приближённо находить значение $f(0)$.

4.2-1 Уравнение вида $f(x) = x$ при $|f'(x)| < 1$ может быть решено методом простых итераций. В этом методе рассматриваются последовательные приближения x_k к корню уравнения. x_0 выбирается произвольно, а каждое следующее приближение вычисляется по формуле

$$x_{k+1} = f(x_k).$$

Вычисления продолжаются, пока $|f(x_k) - x_k| \geqslant \varepsilon$, где ε — некоторое малое число.

Определить функцию, решающую уравнение по заданным f, x_0 и ε. Написать программу, использующую эту функцию.

4.2-2 Многие уравнения вида $f(x) = 0$ можно решить методом хорд. Для этого выбираются два числа x_0 и x_1 — концы отрезка, содержащего корень. Затем выполняется последовательное приближение к корню:

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_0}{f(x_k) - f(x_0)}.$$

Вычисления продолжаются, пока $|f(x_k)| \geqslant \varepsilon$, где ε — некоторое малое число.

Определить функцию, решающую уравнение по заданным f, x_0, x_1 и ε. Написать программу, использующую эту функцию.

4.2-3 Определить функцию, вычисляющую минимум средних

$$\min_{\substack{1\leqslant m\leqslant n,\\ m\in\mathbb{N}}}\left(\frac{1}{m}\sum_{k=1}^{m}f(k)\right)$$

для заданного натурального числа n и функции f. Написать программу, использующую эту функцию.

4.2-4 Определить функцию для приближённого вычисления значения определённого интеграла методом левых прямоугольников по формуле

$$\int_a^b f(x)dx \approx \frac{b-a}{n}\sum_{k=0}^{n-1}f\left(a+k\frac{b-a}{n}\right)$$

для заданного натурального n, функции f и пределов интегрирования a и b. Написать программу, использующую эту функцию.

4.2-5 Определить функцию для приближённого вычисления производной по формуле

$$f'(x) \approx \frac{-3f(x)+4f(x+h)-f(x+2h)}{2h}$$

для заданного малого числа h, точки x и функции f. Написать программу, использующую эту функцию.

4.2-6 Правой свёрткой некоторого списка

$$a_0, a_1, a_2, \ldots, a_n$$

с помощью функции $f(x,y)$ называется выражение

$$f(a_0, f(a_1, f(\ldots f(a_{n-1}, a_n))\ldots)).$$

Определить функцию, вычисляющую правую свёртку списка $1, 2, 3, \ldots, n$ для заданного натурального числа $n > 2$ и функции f. Написать программу, использующую эту функцию.

4.2-7 Определить функцию для приближённого вычисления значения определённого интеграла методом трапеций по формуле

$$\int_a^b f(x)dx \approx \frac{b-a}{n}\left(\frac{f(a)+f(b)}{2}+\sum_{k=1}^{n-1}f\left(a+k\frac{b-a}{n}\right)\right)$$

для заданного натурального n, функции f и пределов интегрирования a и b. Написать программу, использующую эту функцию.

4.2-8 Левой свёрткой некоторого списка

$$a_0, a_1, a_2, \ldots, a_n$$

с помощью функции $f(x, y)$ называется выражение

$$f(\ldots f(f(a_0, a_1), a_2) \ldots, a_n).$$

Определить функцию, вычисляющую левую свёртку списка $1, 2, 3, \ldots, n$ для заданного натурального числа $n > 2$ и функции f. Написать программу, использующую эту функцию.

4.2-9 Определить функцию для приближённого вычисления значения определённого интеграла методом центральных прямоугольников по формуле

$$\int_a^b f(x)dx \approx \frac{b-a}{n} \sum_{k=0}^{n-1} f\left(a + \left(k + \frac{1}{2}\right) \frac{b-a}{n}\right)$$

для заданного натурального n, функции f и пределов интегрирования a и b. Написать программу, использующую эту функцию.

4.3. Рекурсивные функции

4.3-0 Определить рекурсивную функцию для вычисления количества цифр десятичной записи натурального числа. Написать программу, использующую эту функцию.

4.3-1 Квадратный корень произвольного действительного числа a можно вычислить при помощи итерационного метода Герона. Начальное приближение $x_0 = 1$, Каждое следующее вычисляется по формуле

$$x_k = \frac{1}{2}\left(x_{k-1} + \frac{a}{x_{k-1}}\right).$$

Итерации повторяются, пока $\left|x_k^2 - a\right| \geqslant \varepsilon$, где $\varepsilon > 0$ — некоторое малое число.

Определить рекурсивную функцию, находящую приближённое значение $\sqrt{a}$ для заданных a и ε. Написать программу, использующую эту функцию.

4.3-2 Определить рекурсивную функцию для нахождения наибольшего общего делителя двух чисел при помощи алгоритма Евклида. Написать программу, использующую эту функцию.

4.3-3 Линейный конгруэнтный метод генерации псевдослучайных чисел заключается в следующем. Выбирается произвольное число x_0, а каждое следующее вычисляется по формуле

$$x_k = (ax_{k-1} + b) \mod m,$$

где коэффициенты выбраны особым образом. Например,

$$a = 1664525, b = 1013904223, m = 2^{32}.$$

Определить рекурсивную функцию, выводящую на экран n псевдослучайных чисел для заданного x_0.

4.3-4 Определить рекурсивную функцию для вычисления функции Аккермана

$$A(m,n) = \begin{cases} n+1, & \text{если } m = 0; \\ A(m-1,1), & \text{если } m > 0, n = 0; \\ A(m-1,A(m,n-1)), & \text{если } m > 0, n > 0. \end{cases}$$

Написать программу, использующую эту функцию.

4.3-5 Последовательность Нарайаны определяется следующим образом:

$$\begin{aligned} a_0 &= 1, \\ a_1 &= 1, \\ a_2 &= 1, \\ a_k &= a_{k-1} + a_{k-3}. \end{aligned}$$

Определить функцию, находящую k-й член последовательности. Написать программу, использующую эту функцию.

4.3-6 Определить рекурсивную функцию для нахождения максимального из n чисел. Написать программу, использующую эту функцию.

4.3-7 Определить рекурсивную функцию находящую количество единиц в двоичной записи заданного числа. Написать программу, использующую эту функцию.

4.3-8 Определить рекурсивную функцию, которая находит максимальную цифру десятичной записи указанного числа. Написать программу, использующую эту функцию.

4.3-9 Для биномиальных коэффициентов верны следующие соотношения:

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^k,$$
$$C_n^0 = 1,$$
$$C_n^n = 1.$$

Определить рекурсивную функцию для вычисления биномиальных коэффициентов. Написать программу, использующую эту функцию.

5. Структуры и классы

5.1. Структуры

5.1-0 Описать структуру Sequence, соответствующую числовым последовательностям. Последовательность задаётся начальным членом a_0 и функцией f (эта функция — часть структуры). Каждый член кроме первого определяется через предыдущий:

$$a_{k+1} = f(a_k).$$

Написать функцию, находящую n-й член последовательности.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-1 Описать структуру LinearEquation, соответствующую линейному уравнению. Уравнение $ax + b = c$ задаётся коэффициентами a, b и c. Определить функцию, находящую корень линейного уравнения.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-2 Описать структуру Interval, соответствующую отрезкам числовой прямой. Отрезок $[a, b]$ задаётся своими концами. Определить функцию, возвращающую длину общей части двух отрезков или ноль, если они не пересекаются.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных.

5.1-3 Описать структуру Circle, соответствующую окружностям. Окружность задаётся координатами центра и радиусом. Определить функцию, проверяющую, пересекаются ли две окружности.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-4 Описать структуру GeometricProgression, соответствующую геометрическим прогрессиям. Определить функцию, находящую сумму бесконечного числа членов прогрессии.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-5 Описать структуру Date, соответствующую датам григорианского календаря после 1583 г. Дата задаётся тройкой: день d, месяц m и год y. Определить функцию, возвращающую название дня недели, соответствующего дате.

Номер дня недели N можно вычислить по следующим формулам.

$$\begin{aligned} a &= \left\lfloor \frac{14-m}{12} \right\rfloor, \\ Y &= y-a, \\ M &= m+12a-2, \\ N &\equiv \left(7000+d+Y+\left\lfloor \frac{Y}{4} \right\rfloor-\left\lfloor \frac{Y}{100} \right\rfloor+\left\lfloor \frac{Y}{400} \right\rfloor+\left\lfloor \frac{31M}{12} \right\rfloor\right) \mod 7. \end{aligned}$$

Если N равен 0, то результат — воскресенье, 1 — понедельник, 2 — вторник и т. д.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-6 Описать структуру Triangle, соответствующую треугольникам. Треугольник задаётся длинами сторон a, b и c. Определить функцию, вычисляющую его углы.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-7 Описать структуру Point2, соответствующую точкам на плоскости. Определить функцию, вычисляющую расстояние от некоторой точки до прямой, заданной двумя другими точками.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-8 Описать структуру Time, соответствующую моментам времени. Время задаётся как тройка: часы (h), минуты (m) и секунды (s). Определить функцию, вычисляющую количество секунд между двумя моментами.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.1-9 Описать структуру Line, соответствующую прямым на плоскости. Прямая задаётся коэффициентами A, B и уравнения

$$Ax+By+C=0.$$

Определить функцию, находящую координаты точки пересечения двух прямых.

Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

5.2. Классы и перегрузка операций

5.2-0 Описать класс комплексных чисел Complex. Комплексные числа имеют вид $a+bi$, где $i=\sqrt{-1}$, $a, b \in \mathbb{R}$. Определить в нем:

- конструктор, принимающий действительную и мнимую часть;
- копирующий конструктор;
- методы Re и Im, возвращающие мнимую и действительную части;
- методы Abs и Arg, возвращающие модуль и аргумент числа;
- операции сложения, вычитания, умножения и деления (аргументы могут быть как комплексными, так и комплексным и действительным числами);
- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-1 Описать класс отрезков числовой прямой Interval. Определить в нем:

- конструктор, принимающий концы отрезка (должен корректно обрабатывать случаи, когда левый конец больше правого);
- копирующий конструктор;
- метод Length, возвращающий длину отрезка.
- операции интервальной арифметики;
- переопределённый унаследованный ToString.

Операции интервальной арифметики определяются следующим образом:

$$[a_1, b_1] + [a_2, b_2] = [a_1 + a_2, b_1 + b_2],$$
$$[a_1, b_1] - [a_2, b_2] = [a_1 - a_2, b_1 - b_2],$$
$$[a_1, b_1] \times [a_2, b_2] = [\min\{a_1a_2, a_1b_2, b_1a_2, b_1b_2\}, \max\{a_1a_2, a_1b_2, b_1a_2, b_1b_2\}],$$
$$\frac{[a_1, b_1]}{[a_2, b_2]} = \left[\min\left\{\frac{a_1}{a_2}, \frac{a_1}{b_2}, \frac{b_1}{a_2}, \frac{b_1}{b_2}\right\}, \max\left\{\frac{a_1}{a_2}, \frac{a_1}{b_2}, \frac{b_1}{a_2}, \frac{b_1}{b_2}\right\}\right], \quad \text{если } 0 \notin [a_2, b_2].$$

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-2 Описать класс Matrix2 матриц вида $\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$, где $a_{ij} \in \mathbb{R}$. Определить в нем:

- конструктор, принимающий четыре элемента матрицы;

- конструктор, принимающий два элемента главной диагонали (остальные элементы равны нулю);
- копирующий конструктор;
- метод Det, возвращающий определитель матрицы;
- метод Inverse, возвращающий обратную матрицу;
- метод Transpose, возвращающий транспонированную матрицу;
- операции сложения и вычитания матриц;
- операции умножения и деления (аргументы могут быть как матрицами, так и матрицей и действительным числом);
- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-3 Описать класс Polynomial2 квадратных многочленов вида $ax^2 + bx + c$, где $a, b, c \in \mathbb{R}$. Определить в нем:

- конструктор, принимающий коэффициенты многочлена;
- копирующий конструктор;
- метод Value, возвращающий значение многочлена в заданной точке;
- операции сложения и вычитания;
- операции умножения и деления на действительное число;
- операцию вычисления остатка от деления одного многочлена на другой;
- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-4 Описать класс Vector3 векторов в пространстве. Определить в нем:

- конструктор, принимающий координаты вектора;
- копирующий конструктор;
- метод Length, возвращающий длину вектора;
- метод Angle, вычисляющий угол между текущим и другим вектором;
- операции сложения и вычитания;
- операцию скалярного умножения вектора на вектор;
- операции умножения и деления на целое число;

- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-5 Описать класс Money денежных сумм, заданных в виде количества рублей и копеек. Определить в нем:

- конструктор, принимающий количество рублей и копеек (должен корректно обрабатывать случаи, когда копеек больше 100, или количества рублей и копеек имеют разные знаки);
- копирующий конструктор;
- метод TransferCost, принимающий величину комиссии за денежный перевод в процентах и возвращающий его полную стоимость с точностью до копеек (например, для суммы 10 р. 15 к. и величины комиссии 5 % полная стоимость составляет 10 р. 66 к.);
- операции сложения и вычитания;
- операции умножения и деления на действительное число (результат должен округляться до копеек);
- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-6 Описать класс Fraction дробей вида $\frac{m}{n}$, где $m \in \mathbb{Z}, n \in \mathbb{N}$. Определить в нем:

- конструктор, принимающий числитель и знаменатель дроби (должен приводить дробь к несократимому виду);
- копирующий конструктор;
- метод IntegerPart, возвращающий целую часть дроби;
- операции сложения, вычитания, умножения и деления (аргументы могут быть как дробями, так и дробью и целым числом);
- переопределённый унаследованный ToString.

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-7 В европейской музыке весь диапазон музыкальных звуков делится на октавы, которые можно пронумеровать числами от −3 до 5. Октавы

с неположительными номерами имеют собственные названия: субконтроктава (−3), контроктава (−2), большая октава (−1), малая октава (0).

В каждой октаве 12 музыкальных звуков (нот): до, до-диез, ре, ре-диез, ми, фа, фа-диез, соль, соль-диез, ля, ля-диез, си.

Описать класс Note музыкальных звуков. Определить в нем:

- конструктор, принимающий номер октавы и номер звука (считать, что нота до имеет номер 0);
- копирующий конструктор;
- метод Frequency, возвращающий частоту ноты (нота ля первой октавы имеет частоту 440 Гц, частота каждой следующей ноты больше в $\sqrt[12]{2}$ раз);
- операции прибавления и вычитания целого числа, позволяющие получить следующие и предыдущие звуки в общей последовательности;
- переопределённый унаследованный ToString, возвращающий текстовое описание звука (например, «до-диез 2-й октавы» или «ля субконтроктавы»).

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-8 Описать класс дат Date. Определить следующие методы:

- конструктор, принимающий номера дня, месяца и года (рассматривать только положительные номера года);
- копирующий конструктор;
- методы Next и Prev, возвращающие следующий или предыдущий день;
- метод IsLeapYear, проверяющий, является ли текущий год високосным;
- операции прибавления и вычитания целого числа, позволяющие получить следующие за текущим или предыдущие дни;
- переопределённый унаследованный ToString, возвращающий текстовое представление даты (например, «15 апреля 1707 г.»).

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

5.2-9 Дуальным числом называется число вида $a + b\varepsilon$, где $a, b \in \mathbb{R}$, а $\varepsilon \neq 0$ — абстрактная величина, такая что $\varepsilon^2 = 0$.

Описать класс DualNumber дуальных чисел. Определить в нем:

- конструктор, принимающий компоненты a и b дуального числа;
- копирующий конструктор;
- метод Pow, позволяющий возвести дуальное число в произвольную натуральную степень;
- операции сложения, вычитания, умножения и деления (аргументы могут быть как дуальными числами, так и дуальным и действительным числами);
- переопределённый унаследованный ToString.

Операции над дуальными числами определены следующим образом:

$$(a_1 + b_1\varepsilon) + (a_2 + b_2\varepsilon) = (a_1 + a_2) + (b_1 + b_2)\varepsilon,$$
$$(a_1 + b_1\varepsilon) - (a_2 + b_2\varepsilon) = (a_1 - a_2) + (b_1 - b_2)\varepsilon,$$
$$(a_1 + b_1\varepsilon) \times (a_2 + b_2\varepsilon) = (a_1 a_2) + (b_1 a_2 + a_1 b_2)\varepsilon,$$
$$\frac{a_1 + b_1\varepsilon}{a_2 + b_2\varepsilon} = \frac{a_1}{a_2} + \frac{b_1 a_2 - a_1 b_2}{a_2^2}\varepsilon.$$

Предусмотреть возможные исключительные ситуации, если это необходимо.

Написать программу, использующую этот класс.

6. Наследование и полиморфизм

6.1. Наследование и интерфейсы

6.1-0 Описать интерфейс ICalculation для преобразований чисел типа double. Он должен содержать метод Perform (выполнить преобразование), принимающий число и возвращающий результат преобразования.

Описать два класса, реализующих этот интерфейс: Add и Multiply. Первый класс увеличивает число на некоторую величину, а второй умножает на заданный коэффициент (величины определяются конструкторами).

Написать функцию Calculate, принимающую число и два преобразования. Функция должна последовательно применять преобразования к числу и возвращать результат.

Написать программу, использующую эту функцию.

Пример вызова функции:

```
var x = Calculate(1, new Add(2), new Multiply(3));
```

Построить UML-диаграмму.

6.1-1 Описать интерфейс IShape для геометрических фигур. Интерфейс должен содержать методы: Perimeter и Area, возвращающие периметр и площадь соответственно.

Описать классы Triangle (треугольник) и Disk (круг), реализующие этот интерфейс. Параметры фигур должны задаваться при создании экземпляра.

Написать функцию, принимающую фигуру и выводящую на экран её название, параметры, периметр и площадь.

Написать программу, использующую эту функцию.

Построить UML-диаграмму.

6.1-2 Описать абстрактный класс Person (человек), соответствующий человеку. Экземпляры этого класса должны хранить информацию о фамилии (Surname), имени (Name) и отчестве (Patronymic). Класс должен содержать абстрактный метод Income без параметров, возвращающий годовой доход.

Описать классы Student (студент) и Employee (сотрудник), являющиеся потомками класса Person. Класс Student содержит величину ежемесячной стипендии, а класс Employee — величину месячного оклада и процентную ставку налога.

Написать программу, использующую эти классы.

Построить UML-диаграмму.

6.1-3 Описать интерфейс ISolid для геометрических тел. Интерфейс должен содержать методы: Volume и SurfaceArea, возвращающие объём и площадь поверхности соответственно.

Описать классы Cube (куб) и Cylinder (цилиндр), реализующие этот интерфейс. Параметры тел должны задаваться при создании экземпляра.

Написать функцию, принимающую тело и выводящую на экран её название, параметры, объём и площадь поверхности.

Написать программу, использующую эту функцию.

Построить UML-диаграмму.

6.1-4 Описать интерфейс IPrinter, соответствующий способам вывода действительных чисел на экран. Он должен содержать метод Print, принимающий число и выводящий его на экран в соответствии с конкретной реализацией.

Описать классы PrecisionPrinter и LinePrinter, реализующие этот интерфейс. Первый класс выводит число с указанной точностью, а второй после каждого числа рисует линию указанной длины из знаков «-». Необходимые параметры (точность и длина линии) передаются через конструктор.

Написать функцию PrintNumbers, выводящую числа начиная с 0 с указанным шагом при помощью класса, передаваемого через аргументы. Также через аргументы должно передаваться количество выводимых чисел и шаг.

Написать программу, использующую эту функцию.

Пример вызова функции:

```
PrintNumbers(5, 0.2, new LinePrinter(10));
```

Построить UML-диаграмму.

6.1-5 Описать интерфейс IPolynomial, соответствующий многочленам. Он должен содержать методы IsAbove и IsBelow, проверяющие, находится ли указанная точка на плоскости строго выше или ниже графика многочлена соответственно.

Описать классы Linear (линейный) и Quadratic (квадратный), реализующие этот интерфейс. Написать функцию IsBetween, принимающую два многочлена и координаты точки и проверяющую, находится ли точка выше первого и ниже второго многочлена.

Написать программу, использующую эту функцию.

Построить UML-диаграмму.

6.1-6 Описать интерфейс IVector, соответствующий векторам. Он должен содержать методы:

- Size (размер), возвращающий количество координат вектора;

- Get (получить), возвращающий координату вектора по номеру.

Описать классы Vector2 (двумерный вектор), Vector3 (трёхмерный вектор), реализующие этот интерфейс. Написать функцию, находящую скалярное произведение двух векторов, если это возможно. (Произведение определено только для векторов одного размера.)

Написать программу, использующую эту функцию.

Построить UML-диаграмму.

6.1-7 Реализовать иерархию классов: Point (точка) — MassivePoint (точка с массой) — MassiveBall (шар с массой). Точка определяется координатами в пространстве, шар имеет дополнительную характеристику — радиус. Все параметры должны задаваться при создании объектов.

Написать функции:

- Distance (расстояние), определяющую расстояние между произвольными объектами;
- Attraction (притяжение), находящую силу взаимного притяжения между объектами, имеющими массу.

При вычислениях пренебречь возможными взаимными пересечениями объектов.

Написать программу, использующую эти функции.

Построить UML-диаграмму.

6.1-8 Описать интерфейс ISequence, соответствующий числовым последовательностям. Он должен содержать метод GetElement, возвращающий элемент последовательности по его номеру.

Описать классы ArithmeticProgression (арифметическая прогрессия) и GeometricProgression (геометрическая прогрессия), реализующие этот интерфейс. Параметры прогрессий (первый элемент и разность или знаменатель) должны задаваться при создании экземпляра.

Написать функцию Sum которая возвращают сумму указанного количества элементов последовательности начиная с первого.

Написать программу, использующую эту функцию.

Пример вызова функции:

```
var s = Sum(new ArithmeticProgression(3, 5), 10);
```

Построить UML-диаграмму.

6.1-9 Описать абстрактный класс Viewer, соответствующий зрителям кинотеатра. Он должен содержать:

- поле visits, хранящее количество посещений;

- метод Visit (посетить), увеличивающий число посещений на 1;
- абстрактный метод Cost (стоимость), получающий цену билета и возвращающий его стоимость с учётом скидок.

Изначальное количество посещений задаётся при помощи конструктора.

Описать классы RegularViewer (постоянный посетитель) и StudentViewer (студент), являющиеся потомками класса Viewer. Постоянный посетитель за каждые 10 посещений получает дополнительную скидку в 1 %, но не более, чем 20 %. Для студентов на каждое третье посещение даётся скидка в 50 %.

Написать функцию TotalCost, принимающую экземпляр класса Viewer, цену билета и количество сеансов, и возвращающую общую стоимость билетов с учётом скидок. Эта функция должна моделировать посещения кинотеатра.

Написать программу, использующую эту функцию.

Пример вызова функции:

```
var viewer = new StudentViewer(5);
var total = new TotalCost(viewer, 35.00M, 20);
```

Этот вызов вычисляет стоимость двадцати посещений сеансов, стоящих 35 рублей, для студента, который уже посетил пять сеансов. После вызова количество посещений у объекта viewer будет равно 25.

Построить UML-диаграмму.

6.2. Сравнение экземпляров

6.2-0 Описать класс Triangle, соответствующий треугольникам. Определить в нем операции и методы сравнения, сравнивающие треугольники по площади.

6.2-1 Описать класс Time, соответствующий времени суток (часы и минуты). Определить в нем операции и методы сравнения. Более ранние моменты времени считать меньшими, чем более поздние.

6.2-2 Описать класс Progression, соответствующий геометрическим прогрессиям. Определить в нём операции и методы сравнения, сравнивающие прогрессии по значению суммы бесконечного числа их элементов.

6.2-3 Описать класс Box, соответствующий параллелепипедам. Определить операции и методы сравнения, позволяющие проверить, можно ли вложить один параллелепипед в другой (в этом случае вложенный считать меньшим, чем объемлющий). Стенки считать бесконечно тонкими.

6.2-4 Описать класс RGBColor, соответствующий цвету в модели RGB. В этой модели каждый цвет задаётся тремя числами R, G и B, соответствующими интенсивностям красной, зелёной и синей компонент соответственно. Интенсивности — действительные числа из отрезка $[0; 1]$.

Определить в классе операции и методы сравнения, сравнивающие цвета по яркости. Яркость вычисляется по формуле:

$$Y = 0{,}299R + 0{,}587G + 0{,}114B.$$

6.2-5 Описать класс Deposit, соответствующий банковскому вкладу. Вклад определятся тремя величинами: начальной суммой, сроком (в годах) и процентной ставкой. Каждый год сумма на счету увеличивается на величину процентной ставки.

Определить в классе операции и методы сравнения, сравнивающие вклады по величине чистой прибыли за срок действия вклада.

6.2-6 Описать класс Interval, соответствующий отрезкам числовой прямой. Определить в классе операции и методы сравнения, сравнивающие отрезки следующим образом. Из двух пересекающихся отрезков больше тот, у которого доля общей части больше. Непересекающиеся отрезки считаются разными.

6.2-7 Описать класс Rate, соответствующий пакетам услуг связи. Каждый пакет услуг определяется стоимостью, количеством минут в пакете и количеством бесплатных минут.

Определить в классе операции и методы сравнения, сравнивающие пакеты по экономичности. Более экономичным является тот пакет услуг, в котором стоимость одной минуты ниже.

6.2-8 Описать класс Point, соответствующий точкам в пространстве. Определить в классе операции и методы сравнения. Считать меньшей ту точку, которая ближе к началу координат.

6.2-9 Описать класс Line отрезков на плоскости. Определить в нем операции и методы сравнения, сравнивающие отрезки по длине.

7. Массивы и коллекции

7.1. Одномерные массивы

7.1-0 Написать программу, выполняющую вейвлет-преобразование Хаара для заданного массива с длиной, равной степени 2. Само преобразование выполняется следующим образом. В первую половину массива записываются полусуммы пар соседних элементов, а во вторую — полуразности. После этого преобразование повторяется для первой половины массива до тех пор, пока не останется только одна полусумма.

Например, для массива $(2, 8, 6, 0)$ после первого шага получим

$$\left(\frac{2+8}{2}, \frac{6+0}{2}, \frac{2-8}{2}, \frac{6-0}{2}\right) = (5, 3, -3, 3).$$

На втором шаге преобразование будет применено только к полусуммам — $(5, 3)$. В итоге получим массив

$$\left(\frac{5+3}{2}, \frac{5-3}{2}, -3, 3\right) = (4, 1, -3, 3).$$

Так как осталась одна полусумма, это и есть ответ.

7.1-1 Написать программу, случайным образом перемешивающую элементы массива при помощи алгоритма Фишера — Йетса.

Идея метода следующая. На первом шаге первый элемент обменивается со случайно выбранным элементом из массива (им может оказаться и сам первый элемент). На втором шаге второй элемент обменивается со случайно выбранным элементом кроме первого. На третьем — третий элемент со случайным кроме первых двух и так далее. Операция повторяется для каждого элемента массива.

7.1-2 Написать программу, заменяющую все составные числа в массиве последовательных чисел от 2 до указанного числа нулями при помощи алгоритма, называемого решетом Эратосфена. Идея алгоритма следующая: перебираются все элементы, если элемент не нулевой, то заменяются нулями все, идущие после него с шагом, равным значению этого элемента. Алгоритм продолжается до элемента со значением $\lceil \sqrt{N} \rceil$, где N — значение последнего элемента.

Например, для массива $(2, 3, 4, 5, 6, 7, 8, 9)$ последовательность преобразований имеет вид (текущий элемент подчёркнут):

$$(\underline{2}, 3, \not{4}, 5, \not{6}, 7, \not{8}, 9),$$
$$(2, \underline{3}, 0, 5, \not{0}, 7, 0, \not{9}),$$
$$(2, 3, \underline{0}, 5, 0, 7, 0, 0).$$

7.1-3 Написать программу для так называемой «пузырьковой сортировки» массива по неубыванию. Идея метода следующая. Просматриваются все пары соседних элементов и, если их порядок неверный, они обмениваются местами. Проходы по массиву повторяются, пока не окажется, что массив отсортирован.

7.1-4 Написать программу для сортировки массива по неубыванию простым выбором. Идея метода следующая. В массиве ищется наименьший элемент и меняется местами с первым, затем наименьший среди оставшихся меняется местами со вторым и так далее.

7.1-5 Написать программу, переставляющую элементы массива так, чтобы сначала были записаны значения, меньшие некоторого числа p, потом равные ему, а затем остальные.

7.1-6 Написать программу, реализующую так называемое «правило 30» для клеточных автоматов. Дан массив заданной длины, элементы которого равны нулю кроме среднего, который равен единицы. Программа должна выполнить указанное число шагов преобразования массива выводя на каждом шаге результат.

В ходе преобразования проверяются все ячейки массива, кроме крайних. На следующем шаге ячейка становится равной 1 тогда и только тогда, когда она со своими соседями образует одну из четырёх комбинаций: 100, 010, 001 или 011. В противном случае ячейка на новом шаге будет равна 0.

Например для массива размера 7 первые три шага имеют вид:

$$0001000,$$
$$0011100,$$
$$0110010.$$

7.1-7 Написать программу, разбивающую массив на три подмассива с сохранением порядка. При этом суммы элементов в них должны быть как можно ближе друг к другу: разница между наибольшей и наименьшей суммами должна быть минимальной. Программа должны вывести номера первых элементов второго и третьего подмассивов.

Например, массив $(1, 2, 3, 2, 2)$ разбивается на подмассивы $(1, 2)$, (3) и $(2, 2)$. Если элементы нумеруются с нуля, то искомые индексы — 2 и 3.

7.1-8 Написать программу, находящую максимальное значение, которое можно получить, складывая последовательные элементы массива. Например, для массива $(1, -2, 3, 1, -1, 2)$ максимальная сумма равна $3 + 1 + (-1) + 2 = 5$.

7.1-9 Написать программу для тернарного поиска элемента в отсортированном массиве. Алгоритм похож на бинарный поиск. Массив делится

двумя точками на три части приблизительно равного размера. Затем сравнивая искомое значение со значениями в точках, определяем, в какой из частей оно находится. Затем поиск повторяется в этой части, и так далее, пока не будет найден элемент или не окажется, что его нет в массиве.

7.2. Многомерные массивы

7.2-0 Дан двумерный массив целых чисел. Написать программу, заменяющую нулями элементы, стоящие в строках и столбцах, в которых есть нули.

7.2-1 Дан двумерный массив действительных чисел. Написать программу, заменяющую каждый элемент арифметическим средним его смежных соседей по горизонтали, вертикали и диагонали.

7.2-2 Дан двумерный массив размера $M \times N$ действительных чисел. Написать программу для его нормализации по формуле

$$a_{ij}^{\text{норм}} = \frac{a_{ij} - \bar{a}}{\sigma},$$

где $\bar{a}$ — математическое ожидание, а σ — стандартное отклонение, вычисляемые по формулам

$$\bar{a} = \frac{1}{MN} \sum_{i,j} a_{ij},$$

$$\sigma = \sqrt{\frac{1}{MN} \sum_{i,j} (a_{ij} - \bar{a})^2}.$$

7.2-3 Дан двумерный массив действительных чисел. Написать программу, проверяющую, есть ли в нём линейно зависимые строки и столбцы.

7.2-4 Написать программу, заполняющую двумерный массив значениями из треугольника Паскаля с вершиной в верхнем левом углу.

В полученном массиве первая строка и первый столбец должны содержать только единицы, а остальные элементы должны быть равны сумме элемента выше и элемента левее.

Пример заполнения для массива 3×3:

$$\begin{matrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{matrix}$$

7.2-5 Дан двумерный массив целых чисел. Написать программу, проверяющую, сколько в нём квадратов размера 2×2, состоящих только из чётных элементов.

7.2-6 Дан квадратный массив действительных чисел. Написать программу, формирующую массив с суммами элементов на его диагоналях.

7.2-7 Даны две двумерные матрицы. Написать программу, находящую их произведение по правилам матричного умножения, если это возможно.

7.2-8 Дан двумерный массив вещественных чисел. Написать программу, заменяющую каждый его элемент арифметическим средним элементов ниже и правее него.

7.2-9 Дан двумерный массив. Написать программу, записывающую его элементы в одномерный массив по строкам. При этом нечётные строки обходятся слева направо, а чётные справа налево.

7.3. Коллекции и их применение

7.3-0 Дана последовательность, содержащая шесть видов скобок: (,), [,], {, }. Написать программу, проверяющую, является ли скобочная запись, задаваемая последовательностью, корректной. Иными словами, программа должна проверить правильность вложенности пар скобок, отсутствие «незакрытых» и лишних «закрывающих» скобок.

Например, последовательность «([]{})» корректна, а «)[{]}» нет.

7.3-1 Дана закодированная последовательность чисел. Написать декодирующую программу, работающую по следующему алгоритму. Если очередное число x положительное, то оно копируется в декодированную последовательность, если отрицательное, то копируется $|x|$ последних значений из декодированного потока.

Например, последовательность $(1, 2, 3, -2, -3)$ будет декодирована как $(1, 2, 3, 2, 3, 3, 2, 3)$.

7.3-2 Написать программу, заменяющую каждый элемент последовательности чисел на индекс его первого вхождения.

Например, последовательность $(3, 5, 5, 3, 7)$ будет заменена на $(0, 1, 1, 0, 4)$.

7.3-3 Написать программу, проверяющую, верно ли, что количество повторов какого-либо элемента последовательности равно его значению.

Например, последовательность $(4, 2, 4, 2, 4, 1, 4)$ удовлетворяет этому условию.

7.3-4 Написать программу, которая каждый элемент последовательности дополнительно повторяет столько раз, сколько он встречался до этого.

Например, последовательность

$$(1, 2, 1, 2, 1)$$

будет преобразована в

$$(1, 2, 1, 1, 2, 2, 1, 1, 1).$$

7.3-5 Даны три числа a, b, N. Написать программу, выводящую все возможные числа $x \leqslant N$, которые можно получить из числа 1 при помощи операций «прибавить a» и «умножить на b». Операции можно применять произвольное число раз в произвольном порядке.

Для решения этой задачи можно использовать очередь, хранящую числа, требующие проверки, удовлетворяют ли они условию задачи. Числа, производные от проверяемого, также добавляются в очередь.

Например, если $a = 3, b = 2, N = 8$, то программа должна вывести числа $1, 2, 4, 5, 7, 8$.

7.3-6 Написать программу, рекурсивно вычисляющую n-й элемент последовательности f_n, заданный рекуррентным соотношением

$$f_n = \begin{cases} 2f_{n-1} + f_{\lfloor n/2 \rfloor}, & \text{если } n > 2; \\ 1, & \text{если } n \leqslant 1. \end{cases}$$

Ускорить вычисления с помощью мемоизации. Для этого все уже вычисленные промежуточные значения помещаются в словарь. Если требуется найти f_k, которое уже было посчитано ранее, используется значение из словаря.

7.3-7 Рассмотрим изменённый вариант задачи Иосифа Флавия. Числа от одного до некоторого N записаны по кругу. Затем каждое k-е (начиная с единицы) число вычёркивается, как в считалке с выбыванием. Процесс продолжается, пока не останется одно число. Написать программу, выводящую на экран последовательность вычеркивания чисел.

Например, если $N = 5$, а $k = 3$, то последовательность будет иметь вид

$$(3, 1, 5, 2, 4).$$

7.3-8 Даны два стека целых чисел. В первом содержатся некоторые целочисленные значения (данные), а во втором коды команд. Команды управляют процессом вычислений и могут быть одного из следующих видов:

- 1 — взять два значения из первого стека и положить обратно сумму;
- 2 — взять два значения из первого стека и положить обратно разность (верхушка — уменьшаемое);

- 3 — поменять местами два верхних значения первого стека;
- 4 — скопировать в первый стек значение из второго стека, идущее после этой команды.

Перед выполнением команды извлекаются из второго стека. Если код команды отличается от перечисленных, она игнорируется.

Написать программу, моделирующую стековую машину. То есть, выполняющую команды второго стека и затем выводящую на экран верхушку первого. Если какую-то команду выполнить невозможно — вывести на экран сообщение об ошибке.

Например, если стек данных имел вид $(3, 2, 1)$, а стек команд — $(1, 2)$ (верхушка слева), то результатом работы программы будет число 4.

7.3-9 Написать программу, реализующую следующий алгоритм слияния двух отсортированных последовательностей. Даны две последовательности чисел, отсортированных по возрастанию. Новая последовательность формируется при помощи многократного выбора из начала исходных последовательностей наименьшего элемента.

7.4. LINQ

7.4-0 В некоторой коллекции хранятся радиусы окружностей. Составить LINQ-выражение, выбирающее только те радиусы, для которых площади окружностей больше либо равны 4.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-1 В некоторой коллекции хранятся значения углов в градусах. Составить LINQ-выражение, выбирающее только острые углы.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-2 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, формирующее коллекцию их квадратов.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-3 В некоторой коллекции хранятся углы в радианах. Составить LINQ-выражение, выбирающее только те углы, косинусы которых неотрицательны.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-4 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, выбирающее только положительные двузначные числа.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-5 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, выбирающее только чётные числа.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-6 В некоторой коллекции хранятся вещественные числа. Составить LINQ-выражение, формирующее коллекцию с их целыми частями.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-7 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, выбирающее только числа, заканчивающиеся на 7.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-8 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, выбирающее только числа, делящиеся на 11.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

7.4-9 В некоторой коллекции хранятся целые числа. Составить LINQ-выражение, формирующее коллекцию, содержащую их последние цифры.

Написать программу, применяющую выражение к коллекции из случайных чисел, а также решающую эту же задачу явно.

8. Обработка текстовых данных

8.1. Стандартные методы

8.1-0 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, находящую наибольшее количество цифр, идущих в нём подряд.

8.1-1 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, определяющую, на каких позициях в нём встречаются пробелы.

8.1-2 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, заменяющую в нём все прописные буквы на строчные после первого вхождения символа «*».

8.1-3 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, выводящую частоту каждого из его символов.

8.1-4 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, меняющую регистр всех букв на противоположный.

8.1-5 Дан фрагмент текста, запрашиваемый у пользователя. Фрагмент содержит слова, разделённые пробелами (одним или несколькими). Написать программу, выводящую те же слова через один пробел в обратном порядке.

8.1-6 Дан фрагмент текста, запрашиваемый у пользователя. Фрагмент содержит запись арифметического выражения. Написать программу, проверяющую, правильно ли в нём расставлены круглые скобки.

8.1-7 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, заменяющую в нём гласные русские буквы на символ «*».

8.1-8 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, проверяющую, правильно ли в нём записаны буквосочетания «жи», «ши», «ча», «ща».

8.1-9 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, проверяющую, является ли он палиндромом, то есть читается ли одинаково слева направо и справа налево. При проверке регистр букв, знаки препинания и пробелы не учитываются. Например, строка «Аргентина манит негра.» — палиндром.

8.2. Регулярные выражения

8.2-0 Назовём идентификатором последовательность латинских букв, цифр и знаков подчёркивания, начинающуюся не с цифры. Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, определяющую, сколько в нём различных идентификаторов (регистр символов не учи-

тывать). Разделителем идентификаторов считать любой символ, не являющийся его частью.

8.2-1 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, заменяющую встречающиеся в тексте суммы пары натуральных чисел на результат суммирования. Например, строку вида «Сумма равна 12 + 4.» требуется заменить на «Сумма равна 16.».

8.2-2 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, находящую сумму целых чисел (возможно, со знаком), встречающихся в нём.

8.2-3 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, выводящую на экран список адресов электронной почты без повторов, встречающихся в нём. Считать, что адрес электронной почты имеет вид «пользователь@сервер», где имя пользователя и название сервера могут состоять из латинских букв, цифр, дефисов и точек. Кроме того, адрес не может заканчиваться на точку.

8.2-4 Дан фрагмент текста, запрашиваемый у пользователя. Фрагмент содержит список имён и фамилий. В качестве разделителя используется точка с запятой — «Имя Фамилия; Имя Фамилия; …». Написать программу, преобразующую список к формату «Фамилия, Имя; Фамилия, Имя; …».

8.2-5 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, проверяющую, является ли он записью вещественного числа.

8.2-6 Дан фрагмент текста на русском языке, запрашиваемый у пользователя. Написать программу, переводящую первую букву каждого предложения в верхний регистр.

8.2-7 Дан фрагмент текста на русском языке, запрашиваемый у пользователя. Написать программу, определяющую, сколько в нём слов и русских букв. Словом считается последовательность букв, содержащая не более, чем один дефис. Слово не может начинаться с дефиса или заканчиваться на него.

8.2-8 Дан фрагмент текста на русском языке, запрашиваемый у пользователя. Написать программу, проверяющую, встречаются ли в нём идущие подряд одинаковые слова. Словом считается последовательность букв, содержащая не более, чем один дефис. Слово не может начинаться с дефиса или заканчиваться на него.

8.2-9 Дан фрагмент текста, запрашиваемый у пользователя. Написать программу, выводящую на экран список номеров телефонов без повторов, встречающихся в нём. Номера телефонов в тексте записаны в формате «(код)номер». Номер может содержать пробелы или знаки «-» (выводить нужно без них). Также перед первой скобкой может стоять 0.

9. Файлы и сериализация

9.1. Файлы

9.1-0 В текстовом файле хранится таблица действительных чисел. В каждой строке записаны три числа, разделённые символом «;». Написать программу, находящую для каждого столбца сумму и арифметическое среднее.

9.1-1 Написать программу, оценивающую информационную энтропию в указанном файле.

Для этого она должна просмотреть файл побайтово и составить таблицу частот p_k каждого значения байта (k изменяется от 0 до 255). После этого можно вычислить энтропию по формуле

$$H = -\sum_{k=0}^{255} p_k \log_2 p_k.$$

Информационная энтропия в данном случае будет показывать оценку количества информации, приходящейся на один байт файла. Чем она меньше, тем сильнее можно сжать файл.

В приведённой формуле в качестве p_k должны использоваться вероятности. Так как они, вообще говоря, неизвестны, вместо них используются частоты. Частота байта — это отношение количества его вхождений к общему количеству байтов в файле.

9.1-2 В текстовом файле записана информация о продолжительности телефонных звонков. Информация о каждом звонке записана в отдельной строке в формате «Фамилия:Продолжительность». Продолжительность указана в секундах.

Написать программу, находящую суммарную продолжительность разговоров для каждого из абонентов.

9.1-3 Написать функции для сохранения в двоичный файл и чтения из него двумерного массива целых чисел. В начало файла должны записываться размеры массива, а затем в двоичном виде элементы.

Элементы записываются построчно, то есть, вначале в файл помещаются все элементы первой строки массива, затем второй и так далее.

Функция для записи должна принимать имя файла и массив, функция для чтения — только имя файла и возвращать полученный массив.

Написать программу, использующую эти функции.

9.1-4 Написать программу, шифрующую файл и записывающую результат в новый файл. Шифрование выполнить прибавлением по модулю

2 («исключающее или», XOR) к каждому байту исходного файла заданного однобайтового числа.

Этот метод называется гаммированием. В случае наложения всего одного байта он крайне ненадёжен и не должен использоваться для обеспечения безопасности данных.

9.1-5 Написать программу, удваивающую каждый байт указанного двоичного файла с насыщением. Насыщение означает, что если удвоенное значение превышает 255 (максимальное значение для байта), то результат равен 255.

9.1-6 Написать программу, выводящую те строки текстового файла, в которых встречается указанное слово. Перед выводимой строкой указать её номер. В конце вывести общее количество вхождений слова.

9.1-7 Написать программу, просматривающую указанный файл побайтово и выводящую на экран 10 самых частых значений байтов с указанием частоты.

9.1-8 Написать программу, выводящую отсортированный список слов без повторений, встречающихся в указанном текстовом файле.

Считать словом последовательность букв, регистр символов не учитывать (для этого перевести все слова в нижний регистр). Остальные символы в файле игнорировать.

9.1-9 Написать программу, выводящую таблицу частот русских букв (без учёта регистра) в заданном текстовом файле. Таблицу упорядочить по убыванию частоты.

Частотой буквы считать отношение количества её вхождений к общему количеству букв.

9.2. Файловая система

9.2-0 Написать программу, дописывающую к имени всех файлов с расширением «*.txt» слева через пробел дату создания.

Дата должна быть записана в соответствии со стандартом ISO 8601: год, месяц и день через дефисы. Например: «2016-01-18» для 18 января 2016 года. Этот способ удобен тем, что позволяет сортировать даты естественным образом.

9.2-1 Написать программу, выводящую на экран список файлов с расширением «.txt» в заданной папке, в которых содержится указанное слово.

9.2-2 Написать программу, выводящую на экран все файлы в указанной папке с указанием размера. Также для каждого файла указать его долю в общем объёме.

9.2-3 Написать программу, которая объединяет содержимое всех файлов с именем, заканчивающимся на «.part.txt» и записывающая результат в файл с указанным именем. Файлы объединяются в алфавитном порядке.

9.2-4 Написать программу, выводящую список файлов, одновременно присутствующих в двух указанных папках. Сравнивать файлы по содержимому не требуется, достаточно искать файлы с одинаковыми именами.

9.2-5 Дан файл со списком имён файлов (по одному на строку). Написать программу, проверяющую, присутствуют ли файлы из списка в указанной папке. Имена отсутствующих файлов вывести на экран.

9.2-6 Написать программу, перемещающему каждый файл в папке в подпапку с именем, равным первому символу имени файла (без учёта регистра). Например, все файлы с именем, начинающимся с буквы «А» будут перемещены в папку «А» и т. д. (Папки должны создаваться программой при необходимости.)

9.2-7 Написать программу, выводящую на экран список всех файлов с расширением «.txt» в указанной папке. После каждого имени также должна выводиться первая строка файла.

9.2-8 Написать программу, выводящую список файлов в указанной папке, созданных раньше указанной даты.

9.2-9 Написать программу, перемещающую каждый файл в указанной папке в папку с именем, равным дате создания файла. Папки должны создаваться программой при необходимости.

Дата должна быть записана в соответствии со стандартом ISO 8601: год, месяц и день через дефисы. Например: «2016-01-18» для 18 января 2016 года. Этот способ удобен тем, что позволяет сортировать даты естественным образом.

9.3. XML и сериализация

9.3-0 Написать программу, запрашивающую список граждан (определяется фамилий, именем, отчеством и датой рождения) и записывающую его в файл в формате XML.

Написать программу, выводящую список из XML-файла.

9.3-1 Написать программу, получающую список файлов в указанной папке и сохраняющие их имена и даты создания в файл в формате XML.

Написать программу, выводящую список из XML-файла.

9.3-2 Написать программу, запрашивающую с клавиатуры фамилии и номера телефонов и сохраняющую их в файл в формате XML.

Написать программу для поиска номера телефона по фамилии в XML-файле.

9.3-3 Дан файл, хранящий переводы английских слов. Каждая пара перевод — слово занимает одну строку и записана в формате «слово - перевод».

Написать программу сохраняющую информацию о переводах в файл в формате XML.

9.3-4 Написать программу, запрашивающую с клавиатуры информацию о фильмах (название, режиссёр, год выхода) и сохраняющую её в файл в формате XML.

Написать программу, выводящую информацию о фильмах, сохранённую в XML-файл.

9.3-5 Дан файл в формате CSV. В первых трех столбцах записаны фамилия, имя и отчество, а в четвёртом — адрес электронной почты. Написать программу для преобразования файла в формат XML.

В формате CSV данные записываются построчно и разделяются запятыми. В случае, если запятая уже используется в качестве разделителя десятичной части числа (как в русском языке), используют точку с запятой.

9.3-6 Написать программу, запрашивающую информацию о книгах (автор, название и год издания) и сохраняющую её в файл в формате XML.

Написать программу, выводящую информацию о книгах, сохранённую в XML-файл.

9.3-7 Написать программу, запрашивающую информацию о группе (номер, год поступления и фамилии студентов) и сохраняющую её в файл в формате XML.

Написать программу, выводящую информацию о группе, сохранённую в XML-файл.

9.3-8 Дан файл, хранящий информацию о докладах на конференции. Информация о каждом докладе занимает одну строку и записана в формате «Фамилия: Тема (Дата)».

Написать программу для преобразования файла в формат XML.

9.3-9 Написать программу, запрашивающую информацию о студентах (фамилия и средний балл) и сохраняющую её в файл в формате XML.

Написать программу, принимающую на вход XML-файл и выводящую фамилии трёх лучших студентов.

10. Параллельное программирование

10.1. Потоки и параллельные задачи

10.1-0 Дана функция f и массив действительных чисел. Написать программу, находящую среднее значение функции от элементов массива.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-1 Написать программу, находящую сумму делителей числа, введённого с клавиатуры.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-2 Дан массив целых чисел. Написать программу, подсчитывающую количество элементов, кратных числу, введённому с клавиатуры.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-3 Написать программу, находящую приближённое значение определённого интеграла функции f по формуле Симпсона

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left(f(x_0) + 2\sum_{k=1}^{n-1} f(x_{2k}) + 4\sum_{k=1}^{n} f(x_{2k-1}) + f(x_{2n})\right),$$

где $h = \frac{b-a}{2n}$, $x_k = a + kh$.

Чем больше значение n, тем выше точность результата.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-4 Написать программу для транспонирования двумерного массива.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-5 Написать программу для приближённого вычисления числа π методом Монте-Карло (это группа методов для приближённого решения математических задач с использованием случайных чисел).

В частности, для вычисления числа π генерируются N_0 пар равномерно распределённых случайных действительных чисел (x, y), где $x, y \in [-1, 1]$. Подсчитываются те пары, которые попадают в единичный круг. То есть, те, для которых $x^2 + y^2 \leqslant 1$. Если таких пар N, то число π можно оценить как

$$\pi \approx \frac{4N}{N_0}.$$

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-6 В компьютерной графике изображения часто представляются в виде матрицы со значениями яркости отдельных пикселей. Один из методов повышения чёткости изображения заключается в применении ко всем элементам матрицы (кроме крайних) следующего преобразования:

$$a'_{i,j} = 5a_{i,j} - \left(a_{i-1,j} + a_{i+1,j} + a_{i,j-1} + a_{i,j+1}\right).$$

Написать программу, применяющую это преобразование к заданной двумерной матрице.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-7 Для проверки, является ли число простым, можно воспользоваться тестом Ферма. Если для нескольких произвольных чисел a_i, не делящихся на n, верно, что

$$a_i^{n-1} \equiv 1 \pmod{n},$$

то с определённой вероятностью число n является простым. Чем больше количество a_i, тем больше шансов, что n — простое число. Если же хотя бы для одного a окажется, что $a^{n-1} \not\equiv 1 \pmod{n}$, то n — составное число.

Написать функцию возведения целого числа в произвольную степень по заданному модулю. Написать программу, выполняющую тест Ферма.

Проверки распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

10.1-8 Написать программу для параллельной сортировки коллекции. Для этого коллекция делится на две подколлекции, каждая из которых сортируется независимо и параллельно. Затем отсортированные подколлекции объединяются в одну отсортированную.

Объединение (слияние) подколлекций можно выполнить следующим образом. Элементы в начале каждой из них сравниваются, и выбирается меньший элемент. Он извлекается и добавляется в конец результата. Затем действия повторяются.

Сравнить время сортировки массива при параллельных и последовательных вычислениях.

10.1-9 Написать программу, находящую скалярное произведение двух векторов размерности k, представленных массивами.

Вычисления распараллелить на несколько потоков. Сравнить время работы программы при параллельных и последовательных вычислениях.

11. Графические интерфейсы

11.1. Однооконные приложения

11.1-0 Написать программу с графическим пользовательским интерфейсом для решения линейных и квадратных уравнений. Пользователь должен иметь возможность выбрать тип уравнения и ввести коэффициенты.

В интерфейсе должна быть предусмотрена проверка корректности входных данных. Компоненты, которые не требуются для решения выбранной задачи, должны быть неактивны или невидимы.

11.1-1 Написать программу с графическим пользовательским интерфейсом, которая определяет, насколько один цвет в формате RGB близок к другому в евклидовой метрике. (Значения компонентов — целые числа из диапазона от 0 до 255.)

Для проверки близости цветов (R_1, G_1, B_1) и (R_2, G_2, B_2) использовать формулу

$$\rho = \frac{\sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}}{255 \cdot \sqrt{3}}.$$

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных.

11.1-2 Написать программу с графическим пользовательским интерфейсом, которая для введённого текста находит частоту использования каждой буквы. Регистр букв при расчёте не учитывать.

Список букв с частотами должен быть упорядочен по убыванию частоты.

11.1-3 Написать программу с графическим пользовательским интерфейсом для решения треугольников. То есть, программа должна находить стороны и углы треугольника

- по трём сторонам,
- по двум сторонам и углу между ними,
- по стороне и двум прилежащим углам.

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных. Пользователь должен иметь возможность выбрать тип решаемой задачи. Компоненты, которые не требуются для решения выбранной задачи, должны быть неактивны или невидимы.

Материал	ρ, $\frac{\text{Ом}\cdot\text{мм}^2}{\text{м}}$
Серебро	0,015
Медь	0,018
Золото	0,023
Алюминий	0,028
Сталь	0,120
Нихром	1,200

Таблица 1. Удельные сопротивления для различных материалов

11.1-4 Написать программу с графическим пользовательским интерфейсом, которая вычисляет сопротивление провода R по длине l и площади поперечного сечения S. Расчёты выполняются по формуле

$$R = \frac{\rho l}{S},$$

где ρ — удельное сопротивление материала (значения для часто используемых материалов приведены в таблице 1).

Пользователь должен иметь возможность выбрать материал провода из списка.

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных.

11.1-5 Написать программу с графическим пользовательским интерфейсом для расчёта суммы денежного вклада на 12 месяцев.

Входные данные — первоначальный взнос и процентная ставка. Пользователь должен иметь возможность выбрать, начисляются ли проценты на первоначальную или текущую сумму на счету («сложные проценты»).

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных.

11.1-6 Написать программу с графическим пользовательским интерфейсом, которая применяла бы указанное регулярное выражение к введённому тексту.

Пользователь должен иметь возможность выбрать, проверяется ли соответствие текста регулярному выражению или должна выполняться замена. Во втором случае должен указываться текст, на который заменяются найденные фрагменты.

В графическом пользовательском интерфейсе компоненты, которые не требуются для решения выбранной задачи, должны быть неактивны или невидимы.

11.1-7 Написать программу с графическим пользовательским интерфейсом для перевода температуры между кельвинами, градусами Цельсия и Фаренгейта.

Температура в кельвинах t_K и градусах Фаренгейта t_F выражается через температуру в градусах Цельсия t_C следующим образом:

$$t_F = \frac{9}{5} t_C + 32,$$
$$t_K = t_C + 273{,}15.$$

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных. Пользователь должен иметь возможность выбрать направление перевода.

11.1-8 Написать программу с графическим пользовательским интерфейсом для расчёта площади треугольника, круга, трапеции или прямоугольника по длинам сторон. Пользователь должен иметь возможность выбрать геометрическую фигуру.

В графическом пользовательском интерфейсе должна быть предусмотрена проверка корректности входных данных. Компоненты, которые не требуются для решения выбранной задачи, должны быть неактивны или невидимы.

11.1-9 Написать программу с графическим пользовательским интерфейсом, исправляющую текст, набранный в неправильной раскладке. (Например, русский текст, набранный при включённой английской раскладке.)

11.2. Диалоги и взаимодействие форм

11.2-0 Написать программу с графическим пользовательским интерфейсом, которая находит сумму площадей прямоугольников, введённых пользователем. Для ввода размеров прямоугольника предусмотреть отдельную форму.

11.2-1 Написать программу с графическим пользовательским интерфейсом, выводящую сведения о введённом пользователем тексте: количество символов включая и исключая пробелы. Сведения должны отображаться в отдельном окне.

11.2-2 Написать программу с графическим пользовательским интерфейсом, которая для вводимых пользователем значений роста и веса группы людей находит минимальное и максимальное значения индекса массы тела.

Индекс массы тела вычисляется по формуле

$$I = \frac{m}{h^2},$$

где m — масса (кг), h — рост (м).

Для ввода роста и веса предусмотреть отдельную форму.

11.2-3 Написать программу с графическим пользовательским интерфейсом для поиска общего сопротивления цепи из параллельных или последовательно соединённых резисторов.

Для ввода сопротивления предусмотреть отдельную форму. Пользователь должен иметь возможность выбрать тип соединения резисторов.

11.2-4 Написать программу с графическим пользовательским интерфейсом, выводящую на экран списка задач на день с указанием приоритета («важная», «обычная», «не срочно»).

Для ввода текста задачи и приоритета использовать отдельную форму. Пользователь должен выбирать приоритет из списка.

11.2-5 Написать программу с графическим пользовательским интерфейсом, которая для указанного цвета выводит значения его компонент в формате RGB и в формате $Y'C_BC_R$.

Пусть компоненты RGB — однобайтовые целые, тогда компоненты $Y'C_BC_R$ вычисляются по формулам

$$Y' = 16 + 65{,}481 \cdot R + 128{,}553 \cdot G + 24{,}966 \cdot B$$
$$C_B = 128 - 37{,}797 \cdot R - 74{,}203 \cdot G + 112{,}0 \cdot B$$
$$C_R = 128 + 112{,}0 \cdot R - 93{,}786 \cdot G - 18{,}214 \cdot B.$$

Цвет запрашивать с помощью стандартного диалога. Выбранный цвет должен отображаться на главной форме в виде закрашенного квадрата.

11.2-6 Написать программу с графическим пользовательским интерфейсом для вычисления значения многочлена в указанной точке. Каждый коэффициент при степени и сама степень переменной вводятся с помощью отдельной формы.

11.2-7 Написать программу с графическим пользовательским интерфейсом для определения длины ломаной на плоскости. Точки, составляющие ломаную, последовательно вводятся пользователем.

Для ввода координат точек предусмотреть отдельную форму.

11.2-8 Написать программу с графическим пользовательским интерфейсом для вычисления координат центра масс точек на плоскости.

Для ввода координат и массы точек предусмотреть отдельную форму.

11.2-9 Написать программу с графическим пользовательским интерфейсом, которая для указанного цвета подбирает цвет, дополняющий его

до белого. Цвет запрашивать с помощью стандартного диалога. Оба цвета должны отображаться на главной форме в виде закрашенных квадратов.

12. Компонентное программирование

12.1. Библиотеки классов

12.1-0 Разработать библиотеку классов для вычисления значения многочлена в заданной точке по списку коэффициентов. Так, список $(1, 2, 3)$ соответствует многочлену $P(x) = 1x^2 + 2x + 3$. Также реализовать операции сложения и вычитания многочленов (степень может различаться).

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-1 Разработать библиотеку классов для приближённого вычисления значения производных элементарных функций. Для поиска производной функции $f(x)$ в точке x_0 с шагом h можно воспользоваться формулой:

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные (точку и шаг) и дифференцируемую функцию. Должно поддерживаться дифференцирование не менее 10 функций. (Удобно хранить элементарные функции в виде лямбда-выражений в словаре.)

12.1-2 Разработать библиотеку классов для решения в целых числах одного из уравнений

$$ax + by = c,$$
$$ax^2 + by = c,$$

где $a, b, c = \text{const}$. (Такие уравнения называются диофантовыми.)

Решения искать в диапазонах $x_{min} \leqslant x \leqslant x_{max}$ и $y_{min} \leqslant y \leqslant y_{max}$.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные (коэффициенты и диапазон) и решаемое уравнение.

12.1-3 Разработать библиотеку классов для арифметических операций (сложение, вычитание, скалярное произведение) над векторами произвольной размерности.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-4 Разработать библиотеку классов для арифметических операций над обыкновенными дробями. Дробь задаётся числителем и знаменателем.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-5 Разработать библиотеку классов для преобразования обыкновенных дробей в непрерывные и обратно.

Непрерывная (цепная) дробь — это дробь вида

$$[a_0; a_1, a_2, a_3, \dots] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \dots}}}.$$

Для преобразования обыкновенной дроби x в непрерывную можно воспользоваться следующей процедурой:

$$\begin{aligned} a_0 &= \lfloor x \rfloor, \quad x_0 = \{x\}, \\ a_1 &= \left\lfloor \frac{1}{x_0} \right\rfloor, \quad x_1 = \left\{ \frac{1}{x_0} \right\}, \\ a_2 &= \left\lfloor \frac{1}{x_1} \right\rfloor, \quad x_2 = \left\{ \frac{1}{x_1} \right\}, \dots \end{aligned}$$

Здесь $\{x\}$ — дробная часть x. Вычисления продолжаются до тех пор, пока x_i не станет равен 0 или не будет достаточно мал (для иррациональных чисел).

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и направление преобразования.

12.1-6 Разработать библиотеку классов для арифметических операций (сложения, вычитания, умножения и деления) в поле Галуа $GF(4)$. Поле $GF(4)$ — это множество из элементов $0, 1, a, b$, в котором операции сложения и умножения определены так, как показано в таблице 2.

Вычитание и деление выполняется по таблицам. Например, $1 : b = a$, так как $a \cdot b = 1$.

Поля Галуа $GF(k)$ широко применяются в алгоритмах шифрования и помехоустойчивых кодах.

+	0	1	a	b
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

×	0	1	a	b
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

Таблица 2. Операции в поле $GF(4)$

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-7 Разработать библиотеку классов для вычисления площади общей части, суммарной площади, а также сравнения площадей двух прямоугольников со сторонами, параллельными осям координат. Прямоугольники задаются координатами противоположных вершин.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-8 Разработать библиотеку классов для сложения и вычитания угловых мер, заданных градусами, минутами и секундами.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

12.1-9 Разработать библиотеку классов для преобразования строкового представления n-значного числа ($2 \leqslant n \leqslant 10$) в десятичное число и обратно.

Разработать приложения с графическим интерфейсом и интерфейсом командной строки, использующие эту библиотеку. Приложения должны запрашивать входные данные и выполняемую операцию.

13. Рекурсивные структуры данных

13.1. Списки

13.1-0 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Sum, вычисляющий сумму элементов списка, которые удовлетворяют заданному критерию. Критерий задаётся функцией, передаваемой как аргумент.

Написать программу, использующую класс.

13.1-1 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Last, извлекающий последний элемент списка.

Написать программу, использующую класс.

13.1-2 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Reverse, меняющий порядок элементов списка на противоположный.

Написать программу, использующую класс.

13.1-3 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Swap, меняющий местами первый и последний элементы списка.

Написать программу, использующую класс.

13.1-4 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Remove, удаляющий элементы списка, которые удовлетворяют заданному критерию. Критерий задаётся функцией, передаваемой как аргумент.

Написать программу, использующую класс.

13.1-5 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Append, добавляющий указанное значение в конец списка.

Написать программу, использующую класс.

13.1-6 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Extend, добавляющий к списку копию другого списка.

Написать программу, использующую класс.

13.1-7 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод Count, возвращающим количество элементов в списке, которые удовлетворяют заданному критерию. Критерий задаётся функцией, передаваемой как аргумент.

Написать программу, использующую класс.

13.1-8 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление перво-

го элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод InsertAt, добавляющий элемент в указанною позицию списка.

Написать программу, использующую класс.

13.1-9 Написать класс SinglyLinkedList, соответствующий односвязному списку. В классе должны быть реализованы следующие методы: Push — вставка указанного значения в начало списка, Pop — удаление первого элемента списка с возвратом удалённого значения. Также необходимо переопределить унаследованный метод ToString для получения представления списка в виде строки.

Написать метод RemoveAt, удаляющий элемент списка с указанным номером.

Написать программу, использующую класс.

13.2. Деревья

13.2-0 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод LeafCount, находящий количество листьев дерева.

Написать программу, использующую этот класс.

13.2-1 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод Search, проверяющий, есть ли в дереве указанный элемент, и возвращающий путь к нему от корня. Путь представляет собой строку, состоящую из букв «L» (поворот налево) и «R» (поворот направо).

Написать программу, использующую этот класс.

13.2-2 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод IsIdealBalanced, проверяющий, является ли дерево идеально сбалансированным.

Написать программу, использующую этот класс.

13.2-3 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

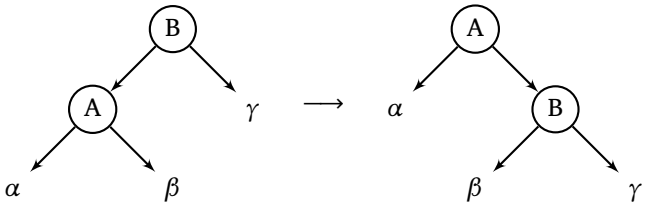

Рис. 1. Правый поворот дерева (греческими буквами обозначены поддеревья)

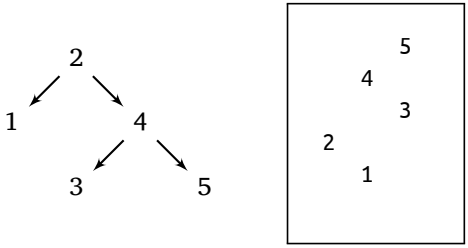

Рис. 2. Пример строкового представления дерева сортировки

Написать метод RightRotate, выполняющий правый поворот дерева относительно корня. Процедура правого поворота показана на рисунке 1.

Написать программу, использующую этот класс.

13.2-4 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод Print, выводящий строковое представление дерева в текстовый поток (например, для последующей записи в файл). Поток должен передаваться через аргументы метода.

Пример строкового представления дерева приведён на рисунке 2.

Написать программу, использующую этот класс.

13.2-5 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод Trim, удаляющий все элементы, уровень которых ниже указанного. Уровень (целое число) передаётся через аргументы метода.

Написать программу, использующую этот класс.

13.2-6 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево),

InOrderWalk (симметричный обход дерева).

Написать метод Flip, изменяющий значение каждого узла на противоположное и зеркально «отражающий» дерево. То есть, все левые подузлы должны стать правыми и наоборот.

Написать программу, использующую этот класс.

13.2-7 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод GetByPath, возвращающий элемент, находящийся в дереве по указанному пути. Путь представляет собой строку, состоящую из букв «L» (поворот налево) и «R» (поворот направо).

Написать программу, использующую этот класс.

13.2-8 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод Height, возвращающий высоту дерева, то есть максимальную длину от корня до листа.

Написать программу, использующую этот класс.

13.2-9 Написать класс SearchTree, реализующий бинарное дерево поиска. Класс должен содержать методы Add (добавление элемента в дерево), InOrderWalk (симметричный обход дерева).

Написать метод Sum, вычисляющим сумму элементов дерева, которые удовлетворяют заданному критерию. Критерий задаётся функцией, передаваемой как аргумент.

Написать программу, использующую этот класс.

14. Обобщённые классы

14.1. Обобщённые структуры данных

14.1-0 Написать обобщённый класс QuadTree, реализующий квадродерево — дерево, у каждого внутреннего узла которого ровно 4 потомка. Значения в узлах могут быть произвольного типа.

Каждый узел дерева должен хранить пару значений (x, y). Ссылки на потомков называются LT, LB, RT, RB. При этом заполнение осуществляется следующим образом. Если добавляется пара (x, y), а текущий узел — (x^*, y^*), то элемент будет добавлен в поддерево в соответствии с таблицей 3.

Класс должен содержать методы:

- Add — добавить узел с указанным значением в дерево в качестве нового листа,
- Exists — проверить, содержит ли дерево узел с указанным значением.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-1 Написать обобщённый класс CircularLinkedList, реализующий кольцевой связный список, хранящий элементы произвольного типа.

Кольцевой связный список — это разновидность односвязного списка, у которого последний элемент содержит ссылку на первый. Экземпляр класса должен содержать ссылку на текущий элемент списка.

Класс должен содержать методы:

- Add — добавить элемент в список после текущего,
- Shift — переместить ссылку на следующий элемент,
- Remove — вернуть текущий элемент и удалить его из списка.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

	L	R
T	$x \leqslant x^*, y > y^*$	$x > x^*, y > y^*$
B	$x \leqslant x^*, y \leqslant y^*$	$x > x^*, y \leqslant y^*$

Таблица 3. Свойства ветвей квадродерева

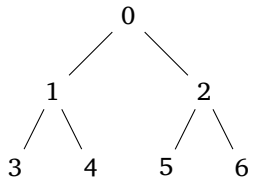

Рис. 3. Соответствие номеров элементов пирамиды и массива

14.1-2 Написать обобщённый класс Heap, соответствующий пирамиде, хранящей элементы произвольного типа. (Эта структура данных также называется кучей.)

Пирамида представляет собой бинарное дерево, элементы которого располагаются в массиве размера $2^d - 1$, где d — высота дерева. Связи между элементами явно не хранятся. Все элементы пронумерованы послойно, как показано на рисунке 3.

Как видно из рисунка, левый потомок узла k будет иметь номер $2k + 1$, а правый — $2k + 2$. Поэтому явное указание связей между узлами не требуется.

Изначально элементы массива равны нулевой ссылке и считаются пустыми. Элементы добавляются так же, как и в бинарном дереве поиска.

Высота дерева должна задаваться в конструкторе.

Класс должен содержать методы:

- Add — добавить элемент с указанным значением в пирамиду,
- Exists — проверить, содержит ли пирамида элемент с указанным значением.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-3 Написать обобщённый класс PriorityStack, реализующий стек с приоритетом, хранящий значения произвольного типа. Вместе со значением элементы стека должны хранить приоритет — некоторое целое число.

Стек с приоритетом реализуется на основе односвязного списка.

При добавлении элемент помещается в такую позицию списка, чтобы приоритеты были упорядочены по возрастанию. На верхушке стека после добавления всегда должен быть элемент с наибольшим приоритетом.

Класс должен содержать методы:

- Push — добавить значение с указанным приоритетом в стек,

- Pop — возвращает элемент с наибольшим приоритетом.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-4 Написать обобщённый класс TernaryTree, реализующий тернарное дерево — дерево, у каждого внутреннего узла которого три потомка. Значения в узлах могут быть произвольного типа. Для каждого узла дерева со значением x должны выполняться свойства: значения узлов левого поддерева меньше, среднего — равны, а правого — больше x.

Класс должен содержать методы:

- Add — добавить узел с указанным значением в дерево в качестве нового листа,
- Exists — проверить, содержит ли дерево узел с указанным значением.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-5 Написать обобщённый класс UnrolledList, реализующий развёрнутый список — односвязный список, содержащий не отдельный значения, а массивы. Размер каждого массива — N (задаётся в конструкторе). При добавлении элемента сперва заполняется первый массив, затем создаётся новый и заполняется он. При этом элемент, хранящий первый массив, ссылается на элемент, хранящий второй.

Класс должен хранить количество заполненный элементов последнего массива, чтобы знать, где заканчивается список.

Класс должен содержать методы:

- Add — добавить новый элемент в конец списка,
- Get — получить элемент списка по номеру.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-6 Написать обобщённый класс PriorityQueue, реализующий очередь с приоритетом, хранящую значения произвольного типа. Вместе со значением элементы очереди должны хранить приоритет — некоторое целое число.

Очередь с приоритетом реализуется на основе двусвязного списка. В нём каждый элемент содержит ссылку как на следующий, так и на предыдущий элементы. Экземпляр класса должен содержать ссылки на первый и последний элементы списка.

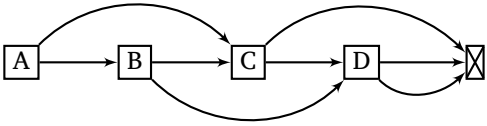

Рис. 4. Односвязный список с дополнительной ссылкой

При добавлении элемент помещается в такую позицию очереди, чтобы приоритеты были упорядочены по возрастанию. В начале очереди будет наименьший приоритет, а в конце — наибольший.

Класс должен содержать методы:

- Enqueue — добавить значение с указанным приоритетом в очередь,
- Dequeue — возвращает элемент с наибольшим приоритетом.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-7 Написать обобщённый класс LinkedList2, реализующий односвязный список, в котором элементы дополнительно содержат ссылку на следующий через один элемент (как показано на рисунке 4). Значения элементов могут быть произвольного типа.

Класс должен содержать методы:

- Add — добавить элемент в начало списка,
- Get — вернуть значение элемента по его номеру (при поиске перемещение выполнять через один элемент при возможности).

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-8 Написать обобщённый класс SearchTree, реализующий бинарное дерево поиска с узлами, хранящими значения произвольного типа.

Каждый узел должен хранить ссылку на родительский узел (для корня — нулевую ссылку).

Класс должен содержать методы:

- Add — добавить элемент в дерево,
- ListFrom — найти узел с указанным значением и вернуть строку со списком значений узлов ветви от корня до найденного узла.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

14.1-9 Написать обобщённый класс OrderedList, реализующий односвязный список, в котором элементы дополнительно содержат ссылку на следующий по величине элемент (то есть, минимальный среди элементов, больших либо равных добавляемому). Значения элементов могут быть произвольного типа.

Класс должен содержать методы:

- Add — добавить элемент в начало списка,
- ListFrom — вернуть строку со списком элементов списка, больших либо равных, чем элемент с указанным номером.

Класс разместить в отдельной библиотеке. Написать программу, использующую его.

15. Графика и визуализация

15.1. Обработка изображений

15.1-0 Написать программу, выполняющую гамма-коррекцию изображения. Для этого значения компонент R, G и B каждого пикселя изменяются по формуле

$$x' = x^{\gamma},$$

где x' — новое значение компоненты, x — старое, а γ задаётся пользователем.

Вычисления выполнять с насыщением. То есть, если значение яркости больше максимально допустимого, то яркость становится равной максимально допустимому значению.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-1 Написать программу, строящую множество Мандельброта. Для этого на изображении размером 300 × 200 пикселей ставятся чёрные и белые точки.

Для каждого пикселя с координатами (x, y) $(0 \leqslant x < 300, 0 \leqslant y < 200)$ выполняется 50 итераций преобразования

$$\begin{aligned} a_{n+1} &= a_n^2 - b_n^2 + \frac{x - 200}{100}, \\ b_{n+1} &= 2a_n b_n + \frac{y - 100}{100}, \end{aligned}$$

где $a_1 = b_1 = 0$.

Если выполняется неравенство $\sqrt{a_{50}^2 + b_{50}^2} < 2$, то пиксель окрашивается в чёрный цвет, иначе в белый.

Программа должна позволять сохранить результат работы на диск.

15.1-2 Написать программу, считывающую из текстового документа четвёрки чисел — координаты концов отрезков, и строящую эти отрезки на изображении размером 200 × 200 пикселей.

Программа должна позволять открыть произвольный текстовый документ на диске и сохранить результат работы.

15.1-3 Написать программу, выполняющую пороговую бинаризацию изображения.

Для каждого пикселя вычисляется яркость по формуле

$$Y = 0{,}299R + 0{,}587G + 0{,}114B,$$

где (R, G, B) — компоненты пикселя (находятся в диапазоне от 0 до 255).

Если $Y > 128$, то пиксель заменяется на белый, иначе на чёрный.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-4 Написать программу, отображающую на экране три цветовых канала изображения — красный, зелёный и синий.

Каждый канал — это отдельное изображение того же размера, у которого все компоненты кроме одной равны нулю, а ненулевая равна соответствующей компоненте исходного изображения.

Программа должна позволять открыть произвольное изображение на диске.

15.1-5 Написать программу, размывающую изображение. Для этого каждая компонента каждого пикселя заменяется на среднее арифметическое значений компонент соседних пикселей. Соседними считаются пиксели, у которых координата отличается не более, чем на единицу. Например, у внутренних пикселей 8 соседей, у пикселей на границе — 5, на углах — 3 соседа.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-6 Написать программу, выделяющую границы на изображении с помощью оператора Собеля.

Для каждого пикселя с координатами (i, j) (кроме крайних) вычисляются значения

$$\begin{aligned}
Y_{i,j} &= 0{,}299R_{i,j} + 0{,}587G_{i,j} + 0{,}114B_{i,j},\\
V_{i,j} &= (Y_{i+1,j-1} + 2Y_{i+1,j} + Y_{i+1,j-1}) - (Y_{i-1,j-1} + 2Y_{i-1,j} + Y_{i-1,j-1}),\\
H_{i,j} &= (Y_{i-1,j+1} + 2Y_{i,j+1} + Y_{i+1,j+1}) - (Y_{i-1,j-1} + 2Y_{i,j-1} + Y_{i+1,j-1}),\\
G_{i,j} &= \frac{\sqrt{V_{i,j}^2 + H_{i,j}^2}}{8}.
\end{aligned}$$

Затем значения компонент пикселя заменяются на $G_{i,j}$.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-7 Написать программу, добавляющую рамку произвольному изображению. Толщина рамки в пикселях и цвет задаются пользователем.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-8 Написать программу, преобразующую цветное изображение в изображение в оттенках серого. Значения компонент пикселей нового изоб-

ражения вычисляются по формуле

$$R' = G' = B' = 0{,}299R + 0{,}587G + 0{,}114B,$$

где (R', G', B') — компоненты пикселей нового изображения, а (R, G, B) — старого.

Программа должна позволять открыть произвольное изображение на диске и сохранить результат работы.

15.1-9 Написать программу, строящую ломаную на изображении размером 400×400 пикселей на основе строки. Начальная точка ломаной — $(300, 200)$, начальное направление — вверх.

Строка считывается посимвольно. Символ «F» — рисование отрезка в текущем направлении длиной $d = 2$ пикселя, «+» — поворот направления на 90° по часовой стрелке, «—» — поворот направления на 90° против часовой стрелке, остальные символы игнорируются.

Строка, по которой строится ломаная, образуется по следующим правилам (такие правила называют системами Линдемайера или L-системами).

Изначально строка равна «FX». Затем в строке выполняются замены:

$$X \to X + YF$$
$$Y \to FX - Y$$

Например, после первой замены строка «FX» преобразуется в строку

$$FX + YF,$$

а после второй в строку

$$FX + YF + FX - YF.$$

Замены повторяются $k = 15$ раз. Получившаяся кривая называется драконом Хартера — Хейтуэя.

Программа должна позволять сохранить результат работы на диск.

15.2. Графики функций

15.2-0 Написать программу, строящую гистограмму яркостей пикселей изображения.

Для этого для каждого пикселя вычисляется яркость по формуле

$$Y = \lfloor 0{,}299R + 0{,}587G + 0{,}114B \rfloor,$$

где (R, G, B) — компоненты пикселей изображения.

Затем строится график, на котором по оси абсцисс отложены значения яркостей, а по оси ординат — доля пикселей с этой яркостью.

Программа должна позволять открыть произвольное изображение на диске.

15.2-1 Написать программу, строящую гипотрохоиду — кривую, задаваемую параметрическими уравнениями:

$$\begin{cases} x = (R-r)\cos t + h\cos\left(\dfrac{R-r}{r}t\right), \\ y = (R-r)\sin t - h\sin\left(\dfrac{R-r}{r}t\right), \end{cases}$$

где R, r, h — константы, задаваемые пользователем, t — параметр.

15.2-2 Написать программу, строящую график многочлена

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0,$$

где коэффициенты a_i и диапазон значений аргумента x задаются пользователем. Количество коэффициентов может быть различным.

15.2-3 Написать программу, строящую траекторию движения тела, брошенную под углом к горизонту с учётом сопротивления воздуха. Траектория задаётся параметрическими уравнениями

$$\begin{cases} x = v_0\cos\varphi \cdot \dfrac{m}{k}\left(1-\exp\left(-\dfrac{k}{m}t\right)\right), \\ y = \dfrac{m}{k}\left(\left(v_0\sin\varphi + \dfrac{mg}{k}\right)\left(1-\exp\left(-\dfrac{k}{m}t\right)\right) - g\,t\right), \end{cases}$$

где v_0 — начальная скорость, φ — начальный угол, m — масса тела, k — коэффициент сопротивления воздуха, $g \approx 9{,}81\ \frac{\text{м}}{\text{с}^2}$ — ускорение свободного падения, t — время.

Все параметры должны задаваться пользователем.

15.2-4 Дан текстовый файл, содержащий целые числа (по одному на строку). Написать программу, строящую столбчатую диаграмму с частотами первых значащих цифр этих чисел.

Например, для чисел $12, 7, 395, 117$ первые значащие цифры — $1, 7, 3, 1$. А частоты, соответственно, — $P[1] = 0{,}5; P[3] = 0{,}25; P[7] = 0{,}25$.

Программа может использоваться для проверки так называемого закона Бенфорда. Согласно ему в числах, взятых из реальной жизни, меньшие цифры встречаются чаще в начале числа, чем большие.

15.2-5 Метод середин квадратов, предложенный фон Нейманом для генерации последовательности псевдослучайных чисел заключается в следующем. Берётся четырёхзначное число, возводится в квадрат и в качестве нового числа используются средние четыре цифры — с третьей по седьмую справа. Затем действия повторяются. (В настоящее время этот метод не используется из-за плохих статистических характеристик получаемой последовательности.)

Например, пусть дано число 1234. Его квадрат равен 1522756, следовательно новое число в последовательности — 5227.

Написать программу, строящую круговую диаграмму, показывающую соотношение чётных и нечётных чисел в первых N элементах последовательности. Параметр N и начальное значение должны задаваться пользователем.

15.2-6 Дан текстовый файл, содержащий пары вещественных чисел, разделённых пробелом (по одной паре на строку). Написать программу, строящую график в декартовых координатах на основе файла. Первое число — абсцисса, второе — ордината.

Программа должна позволять пользователю указать открываемый файл.

15.2-7 Дан текстовый файл, содержащий пары вещественных чисел, разделённых пробелом (по одной паре на строку). Написать программу, строящую график в полярных координатах на основе файла. Первое число — угловая координата, второе — радиальная.

Программа должна позволять пользователю указать открываемый файл.

15.2-8 Так называемое логистическое преобразование имеет вид

$$x_{n+1} = r x_n (1 - x_n).$$

Определим $F(x_0, r)$ как значение x_{1000} для заданных x_0 и r.

Пусть значение x_0 меняется в полуотрезке $(0; 1]$, а r — от 2,5 до 4 с шагом 0,05.

Написать программу, строящую график, на котором для каждого значения x_0 и r из указанных диапазонов ставится точка с координатами $(r, F(x_0, r))$. Полученный график называется бифуркационной диаграммой логистического преобразования.

15.2-9 Одна из моделей, используемых для анализа численности популяции в экологии, — модель Ферхюльста. Если x_k — численность в k-м году, то численность на следующий год можно приближённо найти по формуле:

$$x_{n+1} = x_n + r x_n \left(1 - \frac{x_n}{K}\right),$$

где r — удельная скорость роста, а K — ёмкость экологической ниши популяции.

Написать программу, строящую график численности популяции. Необходимые параметры должны задаваться пользователем.

Обзор литературы

Для успешного решения задач нужно, безусловно, владеть теоретическими знаниями в области программирования. Этот задачник ориентирован на язык программирования C#, поэтому ниже приводится обзор наиболее удачных, по мнению автора книг, позволяющих выучить этот язык. Впрочем, задачи необязательно решать именно на C#. По большей части (за исключением лишь нескольких тем) они составлены так, чтобы их можно было решить практически на любом языке.

Кроме книг по C# ниже также перечислены полезные книги общего характера о программировании, алгоритмах и технологиях разработки программного обеспечения.

Разумеется, хоть все эти книги и заслуживают внимания, не стоит читать их все одновременно. Лучше просмотреть их, а потом выбрать для себя пару книг, которые будут основными. И уже их внимательно прочитать и выполнить упражнения.

Скажем, если вы плохо знакомы с программированием, то в качестве первой книги о языке C#, наверное, стоит взять самоучитель Шилдта [2]. Если же у вас уже есть опыт программирования и для вас нет необходимости разбираться в том, что такое переменная или цикл, то изучить язык можно и по справочнику Албахари [1].

В списке есть книги, которые стоит прочитать всем, вне зависимости от уровня и опыта. По мнению автора, это руководство по написанию качественных программ Макконнелла [8] и учебник по разработке и анализу алгоритмов Седжвика [14] (используемый, кстати, в Принстонском университете).

Тех, кто только начал изучать программирование, возможно заинтересует вводный курс программирования Гарвардского университета [12]. Он в живой и доступной форме знакомит с основными понятиями, универсальными для разработки приложений для компьютеров.

Язык C#

[1] Албахари Д., Албахари Б. C# 5.0. Справочник. Полное описание языка. — М.: Вильямс, 2014.

Несмотря на то, что это справочник, эту книгу можно использовать для изучения различных возможностей языка. Книга подробная, написана простым языком и отлично подойдёт тем, кто уже знаком с основами программирования.

[2] Шилдт Г. C# 4.0: Полное руководство. — М.: Вильямс, 2011.

Книга ориентирована на начинающих программистов, объяснение ведётся «с нуля».

[3] Троелсен Э. Язык программирования C# 5.0 и платформа .NET 4.5. — 6-е изд. — М.: Вильямс, 2013.

Читается немногим сложнее Шилдта, но зато рассматривается более широкий круг тем.

[4] Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C#. 4-е изд. — СПб.: Питер. 2012.

Эта книга для тех, кто уже знаком с программированием под платформу .NET и хочет узнать детали. Книга очень хорошая, но достаточно сложна для начинающих. К тому же, она больше посвящена платформе .NET, чем C#.

[5] Скит Дж. C#. Программирование для профессионалов. — М.: Вильямс, 2011.

Отличная книга, но, как и предыдущая, ориентированная больше на тех, кто уже знаком с C# на начальном уровне.

[6] Полное руководство по языку программирования C# 6.0 и платформе .NET 4.6 / http://metanit.com/sharp/tutorial/

Написанное простым языком краткое руководство по основам языка C#, рассчитанное на начинающих.

[7] C# 5.0 и платформа .NET 4.5 / http://professorweb.ru/

Ещё один, уже более подробный, онлайн-учебник. В нём рассказывается не только о языке C#, но и о применении его для разработки веб-приложений.

Программирование

[8] Макконнелл С. Совершенный код. — СПб.: Питер, 2007.

Великолепная книга о том, как надо программировать. Она не посвящена какому-то языку программирования или технологии, а скорее представляет собой большой сборник советов и рекомендация, как организовать свою работу, как правильно писать программный код и так далее.

[9] Фридл Дж. Регулярные выражения. — СПб.: Символ-Плюс, 2000.

Всеобъемлющее руководство по регулярным выражениям и их применению.

[10] Буч Г., Максимчук Р. А., Энгл М. У., Янг Б. Дж., Коналлен Дж., Хьюстон К. А. Объектно-ориентированный анализ и проектирование с примерами приложений (UML 2). — 3-е издание. — М.: Вильямс, 2010.

Эта книга поможет лучше понять концепцию объектно-ориентированного программирования и познакомиться с языком UML.

[11] Абельсон Х., Сассман Дж. Дж. Структура и интерпретация компьютерных программ. — М.: Добросвет, 2006.

Один из лучших учебников программирования, использующий, однако, язык Scheme для иллюстрации объяснения.

[12] CS50 Основы программирования / http://javarush.ru/cs50.html

Перевод на русский язык видеозаписи курса основ программирования Гарвардского университета. Курс считается одним из лучших, и в то же время он достаточно простой для самостоятельного изучения.

Алгоритмы

[13] Кормен Т. Х., Лейзерсон Ч. И., Ривест Р. Л., Штайн К. Алгоритмы: построение и анализ, 3-е изд. — М.: Вильямс, 2013.

Один из лучших учебников по алгоритмам. Может использоваться и как справочник. В третьем издании была добавлена глава о параллельных алгоритмах.

[14] Седжвик Р., Уэйн К. Алгоритмы на Java. — М.: Вильямс, 2013.

В отличие от предыдущей книги, эта ориентирована на изучение практических аспектов разработки алгоритмов. Отличительной особенностью является большое количество упражнений, позволяющих закрепить материал.

[15] Вирт Н. Алгоритмы и структуры данных. — М.: Мир, 1989. — 360 с.

Книга от создателя языка Pascal, посвящённая разработке и анализу алгоритмов. Менее подробна, чем предыдущая, но написана более простым языком и ориентирована на более широкий круг читателей.

Технологии

[16] Басня о Git / http://hades.github.io/2009/05/the-git-parable-ru/

Рассказ о том, зачем нужен Git и распределённые системы контроля версий.

[17] Git How To / https://githowto.com/ru

Интерактивный обучающий курс по системе контроля версий Git, которую настоятельно рекомендуется (а в дальнейшем обязательно требуется) использовать для хранения исходных текстов программ.

Заключение

Чтобы научиться программировать — нужно программировать. Наверняка читатель уже понял эту истину во время решения задач из сборника. Но важно помнить, что обучение (и самообучение) — это непрерывный процесс. Поэтому не стоит останавливаться на достигнутом. Есть и другие задачники и учебники.

Для человека, который уже имеет определённые навыки в области программирования, наверное, имеет смысл решать не отдельные задачи, а попробовать реализовать небольшой проект. Это позволит приобрести знания и навыки в областях, которые этот сборник, к сожалению, обошёл стороной: в разработке сложных систем, в проектировании архитектуры, в одновременном сочетании различных технологий.

Но в любом случае важность и полезность практики переоценить невозможно.

Предметный указатель

Оглавление

Учебно-практическое издание

Великодный Вадим Игоревич

Задачи по программированию

Учебное пособие

Издаётся в авторской редакции

Подписано в печать 03.03.2017 г.
Формат 60×84/16. Усл. печ. л. 5,0.
Заказ №015. Тираж 10 экз.

Отпечатано с готового оргинал-макета
типографией ООО «РВТ»
3200, г. Бендеры, ул. Московская, д. 30