Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

А. В. Бушкевич, И. В. Лукьянова

КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Методическое пособие
для студентов специальности I-40 02 01
«Вычислительные машины, системы и сети»
заочной формы обучения

Минск 2007

УДК 004.42 (075.8)
ББК 32.973.26-018 я 73
Б 94

Бушкевич, А. В.

Б 94 Конструирование программ и языки программирования : метод. пособие для студ. спец. I-40 02 01 «Вычислительные машины, системы и сети» заоч. формы обуч. / А. В. Бушкевич, И. В. Лукьянова. – Минск : БГУИР, 2007. – 20 с.

ISBN 978-985-488-096-9

В методическом пособии излагаются цель и задачи изучения курса «Конструирование программ и языки программирования». Приводятся варианты заданий к контрольной работе по изучению особенностей языка программирования Си++, требования к содержанию и оформлению работы.

УДК 004.42 (075.8)
ББК 32.973.26-018 я 73

ISBN 978-985-488-096-9

1. Тематика курса

Курс «Конструирование программ и языки программирования» является одним из базовых в рамках обучения студентов методикам и способам создания и проектирования программного обеспечения.

Цель курса – изучение основ организации программного обеспечения с использованием объектно-ориентированного подхода, обучение студентов конструированию программ сложной структуры с использованием машинно-ориентированного языка и языка высокого уровня.

Курс состоит из двух частей: изучение алгоритмического языка программирования Си++ (первый семестр) и изучение языка программирования Ассемблер (второй семестр). Цель первой части курса – изучение основ организации программного обеспечения с использованием объектно-ориентированного подхода.

Курс базируется на знаниях, полученных студентами при изучении дисциплин «Основы алгоритмизации и программирования», «Введение в специальность».

В результате изучения первой части дисциплины «Конструирование программ и языки программирования» студенты должны:

– иметь теоретические сведения об основных положениях объектной модели, природе классов и объектов, принципах классификации, процессе объектно-ориентированного проектирования;

– знать основы идеологии объектно-ориентированного программирования, основные средства языка Си++ для работы с объектами, механизмы наследования, инкапсуляции и полиморфизма, иерархию базовых классов ввода–вывода, особенности реализации перегружаемых функций и перегрузки операций;

– уметь проектировать собственные классы объектов и их иерархию, управлять доступом к элементам класса, пользоваться перегрузкой функций и операций, использовать потоки ввода–вывода.

Основные особенности указанных тем студенты-заочники в краткой форме прорабатывают на установочной сессии.

На зачетной сессии в ходе выполнения двух лабораторных работ проверяются знания и практические навыки, полученные при изучении первой части курса.

2. Варианты контрольных заданий

Контрольная работа состоит из двух заданий:

1) реализация программы работы с динамическими структурами типа очередь, стек, кольцо, бинарное дерево;

2) реализация программы по перегрузке операций и использованию механизмов наследования.

Номер варианта заданий определяется по порядковому номеру фамилии в списке студентов группы. Например, номер фамилии студента в списке 21 – значит, берется первое задание с номером 21 и второе задание также берется с этим же номером. В случае, если номер превышает 30, например 31, 32 и т.д., необходимо вычесть из номера 30 и, таким образом, получить номер варианта задания.

Контрольная работа должна быть выполнена на листах формата А4, сброшюрованных в папку. Страницы следует пронумеровать. Нумерация страниц в работе сквозная. На титульном листе приводятся название факультета, кафедры, наименование дисциплины, номер курса, группы, фамилия, имя, отчество студента, номер варианта. Каждое задание должно содержать текст задачи, если необходимо, пояснения к условию задачи, блок-

схемы алгоритмов функций, разработанных в программе, листинг программы с комментариями и распечатку результатов работы программы.

Задания с динамическими структурами типа очередь, стек, кольцо, бинарное дерево

1. Используя классы, создать кольцо. Его записи содержат указатель на очередь. В записях очереди хранятся строки знаков (не более 80 символов). Среди знаков могут быть открывающие и закрывающие скобки (круглые, квадратные, фигурные). Элементов кольца не менее пяти, элементов очереди не менее трех. Проверить, предшествует ли каждая открывающая скобка соответствующей закрывающей (т.е. правильность расстановки скобок). Вывести в файл номер очереди, если скобки расставлены правильно. Если нет, то указать номер позиции открывающей скобки и номер ошибочной позиции. Затем данные файла необходимо вывести на экран. Имя файла вводится с клавиатуры после создания кольца.

2. Используя классы, создать бинарное дерево. В качестве ключа используется целое число. Записи бинарного дерева содержат указатель на строку, которая является именем файла, в файлах содержится текст. Реализовать функцию, которая для поддерева максимальной длины выводит на экран содержимое файлов, имена которых оканчиваются на «.asm». В строке выводить не более 60 символов, переход на новую строку выполнять на месте пробела. Перед выводом файла в первой строке в позициях с 30-й по 40-ю вывести имя файла, а в позициях с 40-й по 52-ю вывести текущую дату: DD.MM.YY. Рекурсии не использовать. Другие элементы записи бинарного дерева можно самим объявлять и создавать. Записей в бинарном дереве должно быть не менее 10.

3. Используя классы, создать упорядоченное бинарное дерево, которое описывает справочник файлов в файловой системе. Каждому узлу соответствует некоторый файл, в узле содержатся имя файла и дата последнего

обращения к нему. При создании дерево можно заполнить из текстового файла. Узлов в дереве не менее 15. Реализовать функцию, которая удаляет из дерева все файлы (узлы), обращение к которым было осуществлено до даты, введенной с клавиатуры. Исходное и результирующее дерево вывести на экран.

4. Используя классы, создать кольцо. Одним элементом записи кольца является указатель на очередь, вторым элементом – признак упорядочения очереди (В – возрастания, У – убывания). При создании кольца данные можно заполнять из файла. Элементов в кольце должно быть не менее 4, каждая очередь должна содержать не менее 7 записей. Реализовать функцию, которая, не используя сортировок, создает на месте исходных очередей новую упорядоченную очередь путем слияния исходных. Полученную упорядоченную очередь структурами записать в файл (имя файла задается в головном модуле). Записи очереди включают фамилию, имя, отчество, адрес, телефон, дату рождения. Записи очереди упорядочены в соответствии с фамилиями. Новых очередей не создавать. Созданные кольца и очереди, а также результирующую очередь вывести на экран.

5. Используя классы, создать кольцо, записи которого содержат указатель на очередь, в которой посимвольно хранится имя файла. В файлах записаны строки длиной не более 100 знаков. Строка оканчивается точкой. Слова в строке разделены произвольным количеством пробелов. Элементов в кольце должно быть не менее 6. Очереди можно заполнять из файла. Реализовать функцию, в которой читать строки в оперативную память, и в оперативной памяти на месте старой строки получить новую, разделяя слова двумя пробелами. Строки поместить в стек и рассортировать его по алфавиту, строки в стеке не перемещать. Исходные очереди и полученный стек вывести на экран. Системные функции не использовать.

6. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), определяющий частоту обращения к узлу. В записях дерева хранятся фамилия, имя, отчество, адрес, место

работы, должность, дата рождения. Элементы дерева заполнять из файла. Реализовать функцию, которая создаст новое бинарное дерево. В качестве ключа использовать частоту обращения к узлу дерева. Путь к узлам дерева должен быть оптимальным (наикратчайшим). Записи из старого дерева в новое не перемещать. Рекурсии и библиотечные функции не использовать. В дереве не более 30 узлов. Частоты в узлах дерева не совпадают. Исходное и результирующее дерево вывести на экран.

7. Используя классы, создать кольцо. Записи кольца включают указатель на очередь, содержащую указатели на строки знаков, оканчивающиеся точкой. При создании кольца данные можно заполнять из файла. Элементов в кольце должно быть не менее 4, каждая очередь должна содержать не менее 5 записей. Реализовать функцию, которая выводит на экран по столбцам (для каждой очереди свой) строки, которые с начала и с конца читаются одинаково. Строки не должны повторяться. При отсутствии таких строк для рассматриваемой очереди вывести на экран строку: «Искомые записи отсутствуют».

8. Используя классы, создать очередь, записи которой содержат размеры матрицы и указатель на саму матрицу (максимальный размер матрицы 20x20). Элементами матрицы являются строки знаков. При создании очереди матрицы можно заполнять из файла. Очередь должна содержать не менее 10 элементов. Реализовать функцию, в которой из всех строк, хранящихся в очереди, выбрать 5 самых коротких. Матрицы сохранить и не дублировать. Исходную очередь (по матрицам) и полученные пять строк вывести на экран.

9. Ввести n (n не более 30). Используя классы, создать n колец, записи которых содержат строку (не более 15 символов). При создании кольца данные можно заполнять из файла. Реализовать функцию сортировки записей кольца в алфавитном порядке строк. Используя классы, создать очередь, запись очереди содержит указатель на начало очередного кольца. Реализовать функцию, осуществляющую слияние колец в одно рассортированное кольцо. Сортировки при слиянии колец не использовать, строки знаков в кольцах не перемещать,

системные функции не использовать. Промежуточные результаты и окончательный результат вывести на экран.

10. Используя классы, создать дерево, вершины которого содержат строку с названием товара (максимальная длина строки 20 байт), количеством товара – тип целое (int) и номер помещения – также int (это ключ). С клавиатуры вводятся название товара, количество и операция – добавить или отнять. Если вводится операция добавить и суммарное количество превышает некоторое предельно допустимое (для unsigned int – 65535), то необходимо добавить новое помещение (одно или несколько). Если вводится операция отнять и в поле «Количество» остается 0, помещение убирается из дерева в пул незанятых. Реализовать функции создания дерева, добавления и удаления товара. Количество свободных помещений ограничено некоторым N. Если при добавлении указанного товара на складе нет, то создать новую вершину. Если затребован несуществующий товар, выдать соответствующее сообщение.

11. Используя классы, создать генеалогическое дерево, вершины которого содержат имя, фамилию, отчество (все – строки до 20 символов) и дату рождения – целое число типа long, являющееся ключом. Элементы дерева заполнять из файла. Узлов в дереве должно быть не меньше 30. Реализовать функции поиска, которые принимают имя, фамилию или отчество (на выбор), некоторую дату и выводят на экран в алфавитном порядке остальные инициалы всех однофамильцев, родившихся не позднее указанной даты. Реализовать функцию вывода исходного дерева на экран.

12. Используя классы, создать дерево, вершины которого содержат строку с инициалами (максимальная длина 40 символов) и год рождения – целое число типа int, являющееся ключом. Создать 5 деревьев до 20 узлов в каждом. Элементы дерева заполнять из файла. Реализовать функцию, которая получает от пользователя некоторый год и сливает исходные деревья, удалив из них все элементы с годом, меньшим введенного. В случае встречи в разных деревьях вершин с одинаковым ключом оставить в результирующем дереве только одну.

Новых вершин не создавать, ненужные вершины удалить из памяти. Результат вывести на экран.

13. Используя классы, создать дерево, вершины которого содержат имя, фамилию, девичью фамилию (все – строки до 20 символов), пол – символ М или Ж, и дату рождения – целое число типа long, являющееся ключом. Элементы дерева заполнять из файла. Узлов в дереве должно быть не меньше 50. Реализовать функцию поиска, которая принимает дату и выводит на экран в алфавитном порядке всех замужних женщин, родившихся не позднее указанной даты. Признак, по которому считать, что женщина замужем, – наличие в записи девичьей фамилии, отличной от нуля. Реализовать функцию вывода исходного дерева на экран.

14. Используя классы, создать кольцо. Записи кольца включают в себя указатель на очередь, содержащую указатели на строки знаков, оканчивающиеся точкой. При создании кольца данные можно заполнять из файла. Элементов в кольце должно быть не менее 5, каждая очередь должна содержать не менее 4 записей. Реализовать функцию, которая выводит на экран по столбцам (для каждой очереди свой) строки, которые содержат не менее двух букв «а». Строки не должны повторяться. При отсутствии таких строк для рассматриваемой очереди вывести на экран строку: «Искомые записи отсутствуют».

15. Используя классы, создать упорядоченное бинарное дерево, которое описывает справочник файлов в файловой системе. Каждому узлу соответствует некоторый файл, в узле содержатся имя файла и дата последнего обращения к нему. При создании дерево можно заполнить из текстового файла. Узлов в дереве не менее 35. Реализовать функцию, которая удаляет из дерева все файлы (узлы), в имени которых содержится расширение .txt и обращение к которым было произведено после даты, введенной с клавиатуры. Исходное и результирующее дерево вывести на экран.

16. Используя классы, создать бинарное дерево. В качестве ключа используется целое число. Записи бинарного дерева содержат указатель на строку, которая является именем текстового файла. Реализовать функцию, которая для поддерева длиной не более трех узлов выводит на экран содержимое файлов, имена которых оканчиваются на «.txt». В строке выводить не более 60 символов, переход на новую строку выполнить на месте пробела. Перед выводом файла в первой строке в позициях с 30-й по 40-ю вывести имя файла, а в позициях с 40-й по 52-ю вывести текущую дату: DD.MM.YY. Рекурсии не использовать. Другие элементы записи бинарного дерева можно самим объявлять и создавать. Записей в бинарном дереве должно быть не менее 15.

17. Используя классы, создать очередь, записи которой содержат размеры матрицы и указатель на саму матрицу (максимальный размер матрицы 15x15). Элементами матрицы являются строки знаков. При создании очереди матрицы можно заполнять из файла. Очередь должна содержать не менее 10 элементов. Реализовать функцию, в которой из всех строк, хранящихся в очереди, выбрать 5 самых длинных. Матрицы сохранить и не дублировать. Исходную очередь (по матрицам) и полученные пять строк вывести на экран.

18. Используя классы, создать кольцо. Его записи содержат указатель на очередь. В записях очереди хранятся строки (не более 50 символов). В строках записаны слова, разделенные одним пробелом. Элементов кольца не менее пяти, элементов очереди не менее трех. При создании очередей данные заполнять из файла. Реализовать функцию, которая меняет местами первое слово с последним, второе – с предпоследним и т.д. Исходное кольцо и результирующее вывести на экран.

19. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), являющийся ключом. В записях дерева хранятся строки (не более 20 символов). В дереве не более 50 узлов. Элементы дерева заполнять из файла. Реализовать функцию, которая оставит в дереве

только те строки, которые с начала и с конца читаются одинаково, а другие удалит. Исходное и результирующее дерево вывести на экран.

20. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), являющийся ключом. В записях дерева хранятся строки знаков (не более 80 символов). Среди знаков могут быть открывающие и закрывающие скобки (круглые, квадратные, фигурные). В дереве не более 50 узлов. Элементы дерева заполнять из файла. Реализовать функцию вывода дерева на экран. Реализовать функцию, которая проверяет, предшествует ли каждая открывающая скобка соответствующей закрывающей (т.е. правильность расстановки скобок). Вывести на экран значение ключа узла, если скобки расставлены правильно. Если нет, то кроме значения ключа вывести номер позиции открывающей скобки и номер ошибочной позиции.

21. Используя классы, создать кольцо. Записи кольца включают в себя указатель на очередь, содержащую указатели на строки знаков. При создании кольца данные можно заполнять из файла. Элементов в кольце должно быть не менее 7, каждая очередь должна содержать не менее 4 записей. Исходное кольцо вывести на экран. Реализовать функцию, которая переворачивает строки наоборот: первый символ становится последним, а последний – первым и т.п. Вывести полученные строки на экран. Строки не должны повторяться.

22. Используя классы, создать стек. Одним элементом записи стека является указатель на очередь. В очереди хранятся целые числа в произвольном порядке. При создании стека данные можно заполнять из файла или генерировать случайным образом. Элементов в стеке должно быть не менее 5, каждая очередь должна содержать не менее 6 записей. Реализовать функцию, которая, не используя сортировок, создает на месте исходных очередей новую упорядоченную очередь путем слияния исходных. Новых очередей не создавать. Созданные стек и очереди, а также результирующую очередь вывести на экран.

23. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), являющийся ключом. В записях дерева хранятся строки (не более 50 символов). В дереве не более 30 узлов. Элементы дерева заполнять из файла. Реализовать функцию, которая реорганизует дерево так, чтобы узлы в нем были расположены в зависимости от размера строк, хранящихся в узлах дерева. Исходное и результирующее дерево вывести на экран.

24. Используя классы, создать стек. Записи стека содержат имя, фамилию, отчество (все – строки до 15 символов) и дату рождения – целое число типа long. Элементы стека заполнить из файла. Записей в стеке должно быть не меньше 40. Реализовать функцию поиска, которая принимает имя, фамилию, отчество и некоторую дату и выводит на экран в алфавитном порядке остальные инициалы всех однофамильцев, родившихся не позднее указанной даты. Реализовать функцию вывода исходного стека на экран.

25. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), являющийся ключом. В записях дерева хранятся также целые числа. В дереве не более 50 узлов. Элементы дерева заполнять с помощью функции генерации случайных чисел. Реализовать функцию вывода дерева на экран. Реализовать функцию, которая удаляет из дерева все узлы-дубликаты, т.е. узлы с повторяющимися значениями. Реализовать функцию, которая сортирует узлы дерева по возрастанию его информационной части.

26. Используя классы, создать кольцо. Его записи содержат указатель на стек. В записях стека хранятся строки (не более 80 символов). В строках записаны слова, разделенные одним пробелом. Элементов кольца не менее пяти, элементов стека не менее шести. При создании стеков данные заполнять из файла. Реализовать функцию, меняющую местами все четные и нечетные слова в строке. Исходное кольцо и результирующее вывести на экран.

27. Используя классы, создать стек. Его записи содержат указатель на очередь. В записях очереди хранятся строки (не более 60 символов). В строках записаны слова, разделенные одним пробелом. Элементов стека не менее пяти, элементов очереди не менее четырех. При создании очередей данные заполнять из файла. Реализовать функцию, меняющую местами все четные и нечетные слова в строке. Исходный стек и результирующий вывести на экран.

28. Используя классы, создать кольцо. Его записи содержат указатель на очередь. В записях очереди хранятся строки (не более 70 символов). В строках записаны слова, разделенные одним пробелом. Элементов кольца не менее пяти, элементов очереди не менее четырех. При создании очередей данные заполнять из файла. Реализовать функцию, которая для каждой строки подсчитывает сумму кодов символов каждого слова и, если сумма оказалась четной, разворачивает зеркально это слово в строке. Исходное кольцо и результирующее вывести на экран.

29. Используя классы, создать бинарное дерево. В узлах бинарного дерева имеется элемент (целое число), являющийся ключом. В записях дерева хранятся строки (не более 20 символов). В дереве не менее 30 узлов. Элементы дерева заполнять из файла. Реализовать функцию, которая определяет строку, начинающуюся и заканчивающуюся на одну и ту же букву, и удаляет узел с данной строкой из дерева. Узлов со строками, удовлетворяющими заданному условию, в дереве не более 40 %. Исходное и результирующее дерево вывести на экран.

30. Используя классы, создать стек. Его записи содержат указатель на очередь. В записях очереди хранятся строки (не более 30 символов). Элементов стека не менее пяти, элементов очереди не менее четырех. При создании очередей данные заполнять из файла. Найти среднее значение длины строки. Реализовать функцию, которая преобразовывает строки: те из них, длина которых больше среднего, обрезает, а те, длина которых меньше, дополняет пробелами. Исходный стек и результирующий вывести на экран.

Задания по перегрузке операций и использованию механизмов наследования

1. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Перегрузить операторы =, += и операции ввода–вывода в поток. В функции main привести примеры реализации указанных операций.

2. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Перегрузить операторы ==, <, >, != и операции ввода–вывода в поток. В функции main привести примеры реализации указанных операций.

3. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Перегрузить операторы <=, >= и операции ввода–вывода в поток. В функции main привести примеры реализации указанных операций.

4. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Перегрузить операторы +, – и операции ввода–вывода в поток. В функции main привести примеры реализации указанных операций.

5. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Реализуйте с помощью перегрузки ()(int a,int n) операцию выделения подстроки (здесь int a – с какой позиции выделять, int n – сколько символов выделять). В функции main привести примеры реализации указанной перегрузки.

6. Реализовать класс String для работы со строками символов. Память под строку выделять динамически. Постройте класс string так, чтобы для присваивания, передачи параметров и т.п. он имел формат по значению, который обеспечивает копирование полного представления данных строки, а

не просто управляющей структуры данных класса String. В функции main привести примеры реализации класса String.

7. Реализовать класс My_int для работы с целыми числами, который ведет себя в точности как int, за исключением того, что единственные возможные операции – это плюс (унарный и бинарный), минус (унарный и бинарный), умножить, разделить и % – целочисленное деление. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса My_int.

8. Реализовать класс My_int для работы с целыми числами. Перегрузить операторы =, +=, ==, <, >, !=, <=, >= и операции ввода–вывода в поток. В функции main привести примеры реализации класса My_int.

9. Реализовать класс My_int для работы с целыми числами. Перегрузить операторы инкремента (++) и декремента (--), которые работают в обеих формах – префиксной и постфиксной. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса My_int.

10. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузите оператор [] так, чтобы его можно было использовать как справа, так и слева от знака присваивания. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

11. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы +, –, = так, чтобы в них можно было сочетать векторы и числа с плавающей точкой. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

12. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы *, /, = так, чтобы в них можно было сочетать векторы и числа с плавающей точкой.

Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

13. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы *=, /= так, чтобы в них можно было сочетать векторы и числа с плавающей точкой. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

14. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы +=, –= так, чтобы в них можно было сочетать векторы и числа с плавающей точкой. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

15. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы инкремента (++) и декремента (--), которые работают в обеих формах: префиксной и постфиксной. Перегруженные операторы должны увеличивать или уменьшать на единицу целую часть каждого элемента вектора. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

16. Реализовать класс Vector как вектор, состоящий из N чисел типа float. Память под элементы выделять динамически. Перегрузить операторы ==, <, >, !=, <=, >= так, чтобы в них можно было сравнивать векторы. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Vector.

17. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить оператор [] так, чтобы его можно было использовать как справа, так и слева от знака присваивания. Перегруженный оператор [] в качестве аргумента получает число, определяющее позицию элемента в матрице от 0 до N*N–1. Перегрузить операции

ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

18. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы +, –, =. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

19. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы *, /, =. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

20. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы +=, –=. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

21. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы инкремента (++) и декремента (--), которые работают в обеих формах: префиксной и постфиксной. Перегруженные операторы должны увеличивать или уменьшать на единицу целую часть каждого элемента матрицы. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

22. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы ==, <, >, != так, чтобы в них можно было сравнивать матрицы. Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

23. Реализовать класс Matrix размерностью N на N. В Matrix хранятся числа типа float. Память под элементы выделять динамически. Перегрузить операторы <=, >=, != так, чтобы в них можно было сравнивать матрицы.

Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Matrix.

24. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы +, –, =. В случае переполнения параметра «часы» оставлять только значение 23, а остальное отбрасывать. В функции main привести примеры реализации класса Time.

25. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы *, /, =. В случае переполнения параметра «часы» оставлять только значение 23, а остальное отбрасывать. В функции main привести примеры реализации класса Time.

26. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы инкремента (++) и декремента (--), которые работают в обеих формах: префиксной и постфиксной. Увеличение или уменьшение времени выполнять для поля секунд, корректировка полей минут и часов выполняется только в случае переполнения поля секунд и/или минут соответственно. В функции main привести примеры реализации класса Time.

27. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы +=, –=. В случае переполнения параметра «часы» оставлять только значение 23, а остальное отбрасывать. В функции main привести примеры реализации класса Time.

28. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы ==, <, >, !=. Для сравнения переводить значение времени в секунды. В функции main привести примеры реализации класса Time.

29. Реализовать класс Time, который содержит три поля типа int, предназначенные для хранения часов, минут и секунд. Перегрузить операции ввода–вывода в поток, которые работают с объектами данного класса в следующем формате: ЧЧ:ММ:СС. Перегрузить операторы <=, >=, !=. Для сравнения переводить значение времени в секунды. В функции main привести примеры реализации класса Time.

30. Реализовать класс Counter (счетчик). Перегрузить в нем операции инкремента (++) и декремента (--) счетчика, которые работают в префиксной форме. Используя наследование, реализовать класс Counter2, в который добавить возможность использования постфиксной формы операций инкремента (++) и декремента (--). Перегрузить операции ввода–вывода в поток. В функции main привести примеры реализации класса Counter и Counter2.

Литература

Основная

1. Дейтел, Х. М. Как программировать на C++ / Х. М. Дейтел, П. Дж. Дейтел; пер. с англ. – 3-е изд. – М. : Бином, 2001.

2. Страуструп, Б. Язык программирования C++ / Б. Страуструп; пер. с англ. – 3-е, спец. изд. – М. : Бином, 2002.

3. Луцик, Ю. А. Объектно-ориентированное программирование на языке C++ / Ю. А. Луцик, А. М. Ковальчук, И. В. Лукьянова. – Минск : БГУИР, 2003.

4. Скляров, В. А. Язык С++ и объектно-ориентированное программирование : справ. пособие / В. А. Скляров. – Минск : Вышэйш. школа, 1997.

5. Элджер, Дж. С++. Библиотека программиста / Дж. Элджер. – СПб. : Питер, 2001.

Дополнительная

1. Буч, Г. Объектно-ориентированное проектирование с примерами применения / Г. Буч; пер. с англ. – М. : Конкорд, 1992.

2. Соловьев, В. В. Объектно-ориентированное программирование на языке С++ : метод. указания по курсу «Программирование» для студ. спец. 22.01 /

В. В. Соловьев. – Минск : БГУИР, 1994.

3. Климова, Л. С++. Практическое программирование. Решение типовых задач / Л. Климова. – М. : Кудиц-образ, 2001.

4. Касаткин, А. И. Профессиональное программирование на языке Си. Системное программирование / А. И. Касаткин. – Минск : Выш. шк., 1993.

Св. план 2007, поз. 70

Учебное издание

Бушкевич Алексей Владимирович
Лукьянова Ирина Викторовна

КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Методическое пособие
для студентов специальности I-40 02 01
«Вычислительные машины, системы и сети»
заочной формы обучения

Редактор С. Б. Саченко
Корректор М. В. Тезина
Компьютерная верстка Е. Г. Бабичева

Подписано в печать 13.11.2007.	Формат 60x84 1/16.	Бумага офсетная.
Гарнитура «Таймс».	Печать ризографическая.	Усл. печ. л. 1,28.
Уч.-изд. л. 1,0.	Тираж 75 экз.	Заказ 236.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0056964 от 01.04.2004. ЛП №02330/0131666 от 30.04.2004.
220013, Минск, П. Бровки, 6