Подписано электронной подписью:
Вержицкий Данил Григорьевич
Должность: Директор КГПИ ФГБОУ ВО «КемГУ»
Дата и время: 2024-02-21 00:00:00
471086fad29a3b30e244c728abc3661ab35c9d50210dcf0e75e03a5b6fdf6436

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Кемеровский государственный университет»
Новокузнецкий институт (филиал)

Факультет информатики, математики и экономики
Кафедра математики, физики и математического моделирования

Е.В. Решетникова

СТРУКТУРЫ И АЛГОРИТМЫ КОМПЬЮТЕРНОЙ ОБРАБОТКИ ДАННЫХ

Методические указания к выполнению лабораторных работ
для обучающихся по направлению подготовки
02.03.03 Математическое обеспечение и администрирование информационных систем,
профиль «Программное и математическое обеспечение информационных технологий»

Новокузнецк

2020

УДК [378.147.88: 004.043](072)
ББК 74.484(2Рос-4Кем)я73+32.973-018.6я73
Р47

Решетникова Е.В.

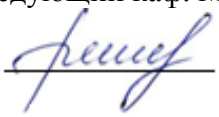
Р47 Структуры и алгоритмы компьютерной обработки данных: методические указания к выполнению лабораторных работ для студентов факультета информатики, математики и экономики, обучающихся по направлению подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем, профиль «Программное и математическое обеспечение информационных технологий» / Е.В. Решетникова; Новокузнецкий ин-т (фил.) Кемеров. гос. ун-та. – Новокузнецк : НФИ КемГУ, 2020 – 65 с.

Методические указания содержат индивидуальные задания для лабораторных работ и указания к их выполнению и оформлению; список основной и дополнительной литературы.

Методические указания предназначены для наиболее рациональной организации аудиторной и внеаудиторной самостоятельной работы студентов.

Рекомендовано на заседании кафедры математики, физики и математического моделирования
Протокол № 3 от 22.10.2020

Заведующий каф. МФММ


/ Е.В.Решетникова

Утверждено методической комиссией факультета информатики, математики и экономики
Протокол № 4 от 12.11.2020

Председатель методической комиссии ФИМЭ


_____/Г.Н.Бойченко

УДК [378.147.88: 004.043](072)
ББК 74.484(2Рос-4Кем)я73+32.973-018.6я73
Р47

© Решетникова Елена Васильевна
© Федеральное государственное бюджетное образовательное учреждение высшего образования «Кемеровский государственный университет», Новокузнецкий институт (филиал), 2020
Текст представлен в авторской редакции

Содержание

ПРЕДИСЛОВИЕ 5
1 АБСТРАКТНЫЕ ТИПЫ ДАННЫХ 7
1.1. Лабораторная работа № 1 «Простые типы данных» 7
1.1.1 Задание к лабораторной работе №1 7
1.1.2 Требования к содержанию практической части по лабораторной работе № 1 9
1.2. Лабораторная работа № 2 «Статические структурные типы данных». 10
1.2.1 Задание к лабораторной работе №2 10
1.2.2 Требования к содержанию практической части по лабораторной работе № 2 24
1.3. Лабораторная работа № 3 «Односвязные и двусвязные списки» 25
1.3.1 Задание к лабораторной работе №3 25
1.3.2 Требования к содержанию практической части по лабораторной работе № 3 29
1.4 Лабораторная работа № 4 «Стеки. Очереди. Деки» 30
1.4.1 Задание к лабораторной работе №4 30
1.4.2 Требования к содержанию практической части по лабораторной работе № 4 35
1.5 Лабораторная работа № 5 «Бинарные деревья» 36
1.5.1 Задание к лабораторной работе №5 36
1.5.2 Требования к содержанию практической части лабораторной работы № 5 40
1.6 Лабораторная работа № 6 «Ориентированные графы» 40
1.6.1 Задание к лабораторной работе №6 40
1.6.2 Требования к содержанию практической части лабораторной работы № 6 42

2 АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ 43
2.1. Лабораторная работа № 7 «Сортировка данных» 43
2.1.1 Задание к лабораторной работе №7 43
2.1.2 Требования к содержанию практической части лабораторной работы № 7 45
2.2. Лабораторная работа № 8 «Поиск и расстановка» 47
2.2.1 Задание к лабораторной работе №8 47
2.2.2 Требования к содержанию практической части лабораторной работы № 8 48
2.3. Лабораторная работа № 9 «Эвристические алгоритмы» 50
2.3.1 Задание к лабораторной работе №9 50
2.3.2. Требования к содержанию практической части лабораторной работы № 9 51
3. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА И ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ 64
4. РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА 64
4.1 Основная литература 64
4.2 Дополнительная литература 64

ПРЕДИСЛОВИЕ

Настоящие методические указания адресованы студентам, получающим квалификацию бакалавр по направлению подготовки: 02.03.03 Математическое обеспечение и администрирование информационных систем, профиль «Программное и математическое обеспечение информационных технологий» и направлены на оказание помощи студентам в выполнении лабораторных работ по дисциплине «Структуры и алгоритмы компьютерной обработки данных».

Предметом изучения в данном курсе являются базовые структуры данных и алгоритмы обработки данных, составляющие «золотой фонд» современного программирования и являющиеся обязательной компонентой знаний каждого квалифицированного программиста.

В качестве основных задач изучения курса можно выделить следующие:

- знание основных, наиболее важных структур данных и алгоритмов, умение применять их в практике программирования;
- знание принципов и критериев оценки эффективности алгоритмов, понимание важности предварительного анализа при решении конкретных задач и выбора наиболее эффективных алгоритмов решения.

Практическим итогом изучения курса должно стать обогащение арсенала программиста рядом высокоэффективных средств, позволяющих значительно повысить качество разрабатываемых программ.

В методические рекомендации включено: варианты индивидуальных лабораторных заданий, методические указания к их выполнению и оформлению, список основной и дополнительной литературы.

Таким образом, данные методические материалы позволяют студенту подготовиться к лабораторным занятиям по соответствующим темам, успешно выполнить индивидуальные лабораторные задания. Методические

указания могут оказаться полезными при написании курсовых и выпускных квалификационных работ.

1 АБСТРАКТНЫЕ ТИПЫ ДАННЫХ

1.1. Лабораторная работа № 1 «Простые типы данных»

1.1.1 Задание к лабораторной работе №1

Для каждого задания составить блок-схему алгоритма с комментариями. Написать программу и пояснения к ней. Указать типы используемых данных и обосновать выбор. В программе использовать только простые типы данных.

Задание 1. Дана функция $Y(x)$. Вычислить значения функции Y на отрезке $[-10+i,\ i]$ с шагом $h=1$ (i – номер Вашего варианта). $c=3i$. Решить задачу тремя способами, используя различные операторы цикла. Оформить вывод аргументов и соответствующих значений функции в виде горизонтальной и вертикальной таблиц.

$$Y(x)=\begin{cases}\dfrac{c}{i\cdot x}, x\le -8+i\\ (c-i)^2\dfrac{x}{i}^2, -8+i<x<i-2,\\ \dfrac{c-i\cdot x}{2icx+x^2}, x\ge i-2\end{cases}$$

Задание 2. Ввести последовательность чисел с клавиатуры. В случае отсутствия решения вывести соответствующее сообщение.

Задачи к заданию 2 по вариантам.

1. Найти количество и сумму всех положительных чисел последовательности. Указать номер второго положительного числа.
2. Найти количество и произведение всех положительных чисел последовательности. Указать номер третьего положительного числа.
3. Найти количество чисел равных 0. Указать номер последнего нуля.
4. Найти количество и сумму всех отрицательных чисел последовательности. Указать номер второго отрицательного числа.

5. Найти среднее арифметическое всех положительных чисел последовательности. Указать номер пятого положительного числа.

6. Найти количество чисел последовательности больших 20 и их произведение. Указать номер второго числа последовательности, большего 20.

7. Найти количество чисел последовательности меньших -23 и их сумму Указать номер пятого числа последовательности, меньшего -23.

8. Найти количество и произведение всех отрицательных чисел последовательности. Указать номер последнего отрицательного числа.

9. Найти количество и сумму квадратов всех четных чисел последовательности. Указать номер второго четного числа.

10. Найти количество и произведение всех нечетных чисел последовательности. Указать номер четвертого нечетного числа.

11. Найти количество и сумму всех чисел последовательности больших 3, но меньших 25. Указать номер второго такого числа.

12. Найти максимальное из всех четных чисел последовательности. Указать номер пятого четного числа.

13. Найти минимальное из всех отрицательных чисел последовательности. Указать номер третьего отрицательного числа.

14. Найти среднее арифметическое всех четных чисел последовательности. Указать номер третьего четного числа.

15. Найти максимальное из всех нечетных чисел последовательности. Указать номер третьего нечетного числа.

16. Найти количество и произведение всех чисел последовательности больших -5, но меньших 20. Указать номер пятого такого числа.

17. Найти количество и сумму квадратов всех чисел последовательности больших -13, но меньших 6. Указать номер второго такого числа.

18. Найти количество и максимальное среди всех чисел последовательности больших 3, но меньших 25. Указать номер максимального числа.

19. Найти среднее арифметическое всех чисел последовательности больших -12, но меньших 36. Указать номер второго такого числа.

20. Найти количество и произведение всех нечетных чисел последовательности больших 3, но меньших 25. Указать номер первого такого числа.

21. Найти количество и сумму всех четных чисел последовательности больших 12. Указать номер пятого такого числа.

Задание 3. С точностью ε найти сумму бесконечного ряда $S = \sum_{k=1}^{\infty} \frac{k + 2i}{i \cdot k^3 - 7}$. (i – номер Вашего варианта).

1.1.2 Требования к содержанию практической части по лабораторной работе № 1

1. При выполнении заданий следует обдумать и обосновать в отчете тип для каждого используемого в задаче данного. Например, тип данного «номер варианта» (i) - константа типа короткое целое, так как номера вариантов постоянны (не варьируются при отладке программы) могут быть только целыми и не превосходят 30.

2. Для решения задания 1 необходимо написать три программы, в которых будут использоваться различные операторы цикла (оператор for, операторы while с предусловием и с постусловием).

3. В задании 3 предварительно необходимо запросить точность ε, затем программа должна рассчитывать слагаемые по формуле и суммировать их в цикле, пока очередное слагаемое не будет меньше ε.

1.2. Лабораторная работа № 2 «Статические структурные типы данных»

1.2.1 Задание к лабораторной работе №2

Для каждого задания составить блок-схему алгоритма с комментариями. Написать программу и пояснения к ней.

Задание 1. Сформировать статические массивы данных и выполнить операции согласно задаче варианта.

Задачи к заданию 1 по вариантам.

1. Ввести восемь элементов массива X. Вычислить элементы массива Yпо формуле $y[i] = 2*x[i]$. Вывести массив *Y*.

2. Ввести с клавиатуры восемь элементов массива *X*. Вычислить сумму всех элементов массива *X*. Вывести на экран сумму и массив *X*.

3. Ввести шесть элементов массива *A*. Вычислить произведение всех элементов массива A. Вывести массив A.

4. Сформировать массив *K* из 23 элементов. Вычислить сумму элементов, значение которых больше (-2).

5. Сформировать массив *K* из 23 элементов. Вывести его на экран. Вычислить произведение положительных элементов.

6. Ввести массив *P* из 15 элементов. Найти количество отрицательных элементов в массиве.

7. Ввести массив *K* из семи элементов. Вычислить сумму отрицательных элементов.

8. Ввести массив *B* из девяти элементов, определить количество элементов, значение которых меньше 10.

9. Ввести массив B из девяти элементов. Увеличить на 5 все элементы, значение которых меньше 10 и определить их количество.

10. Ввести массив. Найти отдельно произведение положительных элементов и произведение отрицательных элементов.

11. Сформировать массив из 12 элементов. Вывести его на экран, выделив минимальный и максимальный элементы массива (например, восклицательными знаками).

12. С помощью генератора случайных чисел сформировать массив D из 18 элементов. Вывести его на экран. Найти сумму положительных элементов массива, произведение отрицательных и количество нулей.

13. Ввести массив A из 15 целочисленных элементов. Сформировать массив B из квадратных корней и массив C из квадратов элементов массива A. Вывести массивы B и C.

14. Ввести массив. Найти произведение положительных элементов массива, стоящих после максимального элемента.

15. Дан массив из восьми элементов. Поменять местами минимальный и максимальный элементы. Вывести на экран исходный массив и массив после перестановки минимального и максимального элементов, выделив их (например, восклицательными знаками).

16. Сформировать массив из 12 элементов случайным образом. Вывести его. Найти заданный элемент (вводится с клавиатуры) и вывести его индекс. Если искомый элемент не найден, то вывести сообщение об этом на экран.

17. Дан массив A из N элементов типа Char. Сформировать массив B из N элементов типа Boolean следующим образом: если элемент a[i] - буква . то элементу b[i] присвоить значение True, если же a[i] - цифра, то b[i] присвоить значение False.

18. Сформировать массив A из N элементов. Вывести массив на экран. Определить, есть ли в массиве A элементы, равные числу M.

19. Ввести массив *P* из 15 элементов. Найти количество отрицательных элементов в массиве.

20. С помощью генератора случайных чисел сформировать массив D из 18 элементов. Вывести его на экран. Найти сумму положительных элементов массива, произведение отрицательных и количество нулей.

21. Ввести с клавиатуры восемь элементов массива *X*. Вычислить сумму всех элементов массива *X*. Вывести на экран сумму и массив *X*.

22. Ввести массив B из девяти элементов. Увеличить на 5 все элементы, значение которых меньше 10 и определить их количество.

23. Ввести массив. Найти произведение положительных элементов массива, стоящих после максимального элемента.

Задание 2. Сформировать динамические массивы данных и выполнить операции согласно задаче варианта.

Задачи к заданию 2 по вариантам.

1. Сформировать массив A из N элементов случайным образом и целое число M (вводится с клавиатуры или формируется с помощью генератора случайных чисел). Из элементов массива A, не превышающих числа M, сформировать массив B. Вывести оба массива на экран. Для обоих массивов найти сумму ненулевых элементов (отдельно для каждого массива).

2. Сформировать массив C из N элементов случайным образом. Из положительных элементов массива C сформировать массив A. Вывести оба массива на экран. Для обоих массивов найти произволение ненулевых элементов (отдельно для каждого массива).

3. Сформировать массив K из N элементов случайным образом. Вывести его. Из элементов массива, абсолютное значение которых не превышает пяти, сформировать массив C. Вывести его на экран.

4. Сформировать массив K из N элементов случайным образом. Из ненулевых элементов массива K сформировать массив X. Вывести оба массива на экран. Вычислить количество положительных элементов, расположенных после минимального элемента массива X.

5. С помощью генератора случайных чисел сформировать массив E из N элементов. Сформировать массив F из положительных элементов массива E. Вывести оба массива на экран. Найти сумму нечетных элементов массива F, расположенных перед минимумом, и количество четных элементов массива F, расположенных после максимума.

6. Сформировать массив B из N элементов случайным образом. Вывести его. Из массива B сформировать массив A таким образом, чтобы в массиве A были только неповторяющиеся элементы массива B.

7. С помощью генератора случайных чисел сформировать массив A. Вывести его на экран. Сформировать одномерный массив B из положительных элементов массива A. Вывести полученный массив на экран. Определить среднее арифметическое отрицательных элементов массива A и минимальный элемент и его координаты массива B.

8. С помощью генератора случайных чисел формировать массив K. Вывести его. Из элементов массива, абсолютное значение которых не превышает пяти, сформировать массив C. Вывести его на экран.

9. Сформировать массив P из N элементов случайным образом. Из элементов массива, меньших числа K, сформировать массив M. Вывести оба массива на экран. Вычислить произведение элементов, расположенных перед максимальным элементом массива M.

10. Сформировать массив K из N элементов случайным образом. Из ненулевых элементов массива K сформировать массив X. Вывести оба массива на экран. Вычислить количество положительных элементов, расположенных после минимального элемента массива X.

11. Сформировать массив B из N элементов случайным образом. Вывести его. Из массива сформировать массив A таким образом, чтобы в массиве A были только неповторяющиеся элементы массива B. Вывести массив A. Найти заданный элемент C в массиве A.

12. Сформировать массив A из N элементов случайным образом и целое число M (вводится с клавиатуры или формируется с помощью генератора случайных чисел). Из элементов массива A, не превышающих числа M, сформировать массив B. Вывести оба массива на экран. Для обоих массивов найти сумму ненулевых элементов (отдельно для каждого массива).

13. Сформировать массив C из N элементов случайным образом. Из положительных элементов массива C сформировать массив A. Вывести оба массива на экран. Для обоих массивов найти произволение ненулевых элементов (отдельно для каждого массива).

14. Сформировать массив K из N элементов случайным образом. Вывести его. Из элементов массива, абсолютное значение которых не превышает пяти, сформировать массив C. Вывести его на экран.

15. Сформировать массив K из N элементов случайным образом. Из ненулевых элементов массива K сформировать массив X. Вывести оба массива на экран. Вычислить количество положительных элементов, расположенных после минимального элемента массива X.

16. С помощью генератора случайных чисел сформировать массив E из N элементов. Сформировать массив F из положительных элементов массива E. Вывести оба массива на экран. Найти сумму нечетных элементов массива F, расположенных перед минимумом, и количество четных элементов массива F, расположенных после максимума.

17. Сформировать массив B из N элементов случайным образом. Вывести его. Из массива B сформировать массив A таким образом, чтобы в массиве A были только неповторяющиеся элементы массива B.

18. С помощью генератора случайных чисел сформировать массив A. Вывести его на экран. Сформировать одномерный массив B из положительных элементов массива A. Вывести полученный массив на

экран. Определить среднее арифметическое отрицательных элементов массива A и минимальный элемент и его координаты массива B.

19. С помощью генератора случайных чисел формировать массив K. Вывести его. Из элементов массива, абсолютное значение которых не превышает пяти, сформировать массив C. Вывести его на экран.

20. Сформировать массив P из N элементов случайным образом. Из элементов массива, меньших числа K, сформировать массив M. Вывести оба массива на экран. Вычислить произведение элементов, расположенных перед максимальным элементом массива M.

21. Сформировать массив K из N элементов случайным образом. Из ненулевых элементов массива K сформировать массив X. Вывести оба массива на экран. Вычислить количество положительных элементов, расположенных после минимального элемента массива X.

22. Сформировать массив B из N элементов случайным образом. Вывести его. Из массива сформировать массив A таким образом, чтобы в массиве A были только неповторяющиеся элементы массива B. Вывести массив A. Найти заданный элемент C в массиве A.

Задание 3. Написать программу, выполняющую операции над строковой переменной, согласно задаче Вашего варианта.

Задачи к заданию 3 по вариантам.

1. С клавиатуры ввести последовательность символов. Вычислить длину этой последовательности.

2. С клавиатуры ввести предложение из нескольких слов, разделенных любым количеством пробелов. Вычислить количество слов в предложении.

3. С клавиатуры ввести последовательность символов, не содержащую пробелов. Вывести на экран эту последовательность в обратном порядке («информатика» — «акитамрофни»).

4. С клавиатуры ввести предложение, состоящее из произвольного числа слов. Подсчитать количество повторяющихся слов в предложении.

5. С клавиатуры ввести строку, состоящую из произвольного числа слов. Написать функцию, которая выводит на экран строку, в которой перевернуто каждое слово («мир труд май» — «рим дурт йам»).

6. С клавиатуры ввести последовательность символов. Вычислить длину этой последовательности.

7. С клавиатуры ввести предложение из нескольких слов, разделенных любым количеством пробелов. Вычислить количество слов в предложении.

8. С клавиатуры ввести последовательность символов, не содержащую пробелов. Вывести на экран эту последовательность в обратном порядке («информатика» — «акитамрофни»).

9. С клавиатуры ввести предложение, состоящее из произвольного числа слов. Подсчитать количество повторяющихся слов в предложении.

10. С клавиатуры ввести строку, состоящую из произвольного числа слов. Написать функцию, которая выводит на экран строку, в которой перевернуто каждое слово («мир труд май» — «рим дурт йам»).

11. С клавиатуры ввести последовательность символов. Вычислить длину этой последовательности.

12. С клавиатуры ввести предложение из нескольких слов, разделенных любым количеством пробелов. Вычислить количество слов в предложении.

13. С клавиатуры ввести последовательность символов, не содержащую пробелов. Вывести на экран эту последовательность в обратном порядке («информатика» — «акитамрофни»).

14. С клавиатуры ввести предложение, состоящее из произвольного числа слов. Подсчитать количество повторяющихся слов в предложении.

15. С клавиатуры ввести строку, состоящую из произвольного числа слов. Написать функцию, которая выводит на экран строку, в которой перевернуто каждое слово («мир труд май» — «рим дурт йам»).

16. С клавиатуры ввести последовательность символов. Вычислить длину этой последовательности.

17. С клавиатуры ввести предложение из нескольких слов, разделенных любым количеством пробелов. Вычислить количество слов в предложении.

18. С клавиатуры ввести последовательность символов, не содержащую пробелов. Вывести на экран эту последовательность в обратном порядке («информатика» — «акитамрофни»).

19. С клавиатуры ввести предложение, состоящее из произвольного числа слов. Подсчитать количество повторяющихся слов в предложении.

20. С клавиатуры ввести строку, состоящую из произвольного числа слов. Написать функцию, которая выводит на экран строку, в которой перевернуто каждое слово («мир труд май» — «рим дурт йам»).

Задание 4. Написать программу, выполняющую операции над двумерным массивом, согласно задаче Вашего варианта.

Задачи к заданию 4 по вариантам.

1. Ввести с клавиатуры матрицу *P 7x10*. Вывести се на экран. Найти сумму отрицательных элементов матрицы, произведение положительных элементов, количество ненулевых элементов.

2. Ввести с клавиатуры матрицу P 5x7 построчно. Вывести ее на экран. Все ненулевые элементы заменить обратными по величине (обратным для числа a является число 1/a) и противоположными по знаку. Вывести измененную матрицу на экран.

3. Сформировать матрицу *S* 12x12. Вывести матрицу. Заменить нулями все элементы, расположенные на главной диагонали и выше ее. Вывести измененную матрицу.

4. Ввести матрицу Г 4х4. Вывести ее на экран. Вычислить произведение положительных элементов, лежащих под главной диагональю.

5. Сформировать случайным образом целочисленную матрицу *NxN*. Сформировать два одномерных массива *A* и *B:* в массив *A* записать элементы матрицы, расположенные на главной диагонали и ниже нее, в массив *B* — элементы, лежащие на побочной диагонали и ниже нее. Вывести оба массива на экран теми же цветами, что главная и побочная диагональ матрицы соответственно.

6. Ввести с клавиатуры матрицу С 4х3. Вывести ее на экран. Создать из матрицы С матрицу Р таким образом, что все столбцы матрицы С станут строками матрицы *P,* а все строки столбцами матрицы *P* в том же порядке, т. е. $P=C^T$. Вывести матрицу *P*.

7. Ввести с клавиатуры матрицу Р 3х4 и число *H.* Вывести ее на экран. Определить номер 1-го элемента, большего Н, и посчитать сумму элементов, ему предшествующих, если считать от левого верхнего угла матрицы построчно.

8. Латинским квадратом порядка *N* называется квадратная таблица размером *Nx N,* каждая строка и каждый столбец которой содержат все числа от 1 до N. Для заданного N (вводится с клавиатуры) в матрице *L NxN* построить латинский квадрат порядка *N*.

9. Сформировать матрицу N 12х15. Поменять местами минимальный и максимальный элементы. Вывести матрицу до замены и после.

10. Сформировать матрицу *A 6x7*. Вывести ее на экран. Найти произведение элементов строки, в которой находится элемент с наибольшим значением.

11. Ввести матрицу А 7х8. Сформировать три одномерных массива: *B* - из максимальных элементов строк матрицы *A, C* - из минимальных

элементов столбцов матрицы *A*, *D* - из средних арифметических элементов строк матрицы *A*.

12. Сформировать вещественную матрицу A 6x10. Найти среднее арифметическое наибольшего и наименьшего значений ее элементов, расположенных ниже главной диагонали.

13. Сформировать матрицу A 5x7. Вывести матрицу. Найти и вывести значение минимального элемента. Вывести координаты всех минимальных элементов матрицы и их количество.

14. Дана целочисленная матрица 10x10. Сформировать одномерный массив, состоящий из сумм отрицательных элементов строк, и одномерный массив, состоящий из сумм отрицательных элементов столбцов.

15. Сформировать матрицу *B* 10x10. Все элементы с наибольшим значением в данной матрице заменить нулями, если эти элементы не принадлежат 1-й или последней строкам. Вывести матрицу до и после замены.

16. Используя генератор случайных чисел, сформировать матрицу A 10x8 из вещественных чисел. Вывести матрицу A. Получить новую матрицу *C* из матрицы *A* путем деления всех элементов матрицы *A* на ее наибольший по модулю элемент.

17. Дана целочисленная матрица 8x8. Найти се минимальный элемент и сумму элементов заштрихованной области.



18. Дана целочисленная матрица 8x8. Найти ее минимальный элемент и сумму элементов заштрихованной области.



19. Дана целочисленная матрица размерностью *Mx К.* Найти минимальное значение для верхней половины матрицы и максимальное значение для нижней половины.

20. Дана целочисленная матрица 8x8. Найти ее максимальный элемент и сумму элементов заштрихованной области.



21. Ввести символьную матрицу 5x7. Сформировать одномерный массив из сумм кодов символов каждой строки и одномерный массив из сумм кодов символов каждого столбца.

22. Дана целочисленная матрица Mx*K.* Сформировать новую матрицу путем прибавления в каждой строке к каждому элементу исходной матрицы максимального значения элемента данной строки.

23. Дана матрица 7x6. Сформировать одномерный массив, содержащий сумму максимального и минимального элементов каждой строки. Вывести матрицу и массив на экран.

25. Ввести символьную матрицу 5x7. Сформировать одномерный массив из минимальных кодов символов каждой строки и одномерный массив из максимальных кодов символов каждого столбца.

26. Дана целочисленная матрица NxK. Сформировать новую матрицу путем прибавления в каждом столбце к каждому элементу исходной матрицы минимального значения элемента данного столбца.

27. Дана матрица 7x7. Сформировать одномерный массив, содержащий элементы главной диагонали исходной матрицы. Вывести матрицу и массив на экран.

28. Сформировать вещественную матрицу N 6x6. Найти среднее арифметическое наибольшего и наименьшего значений ее элементов, расположенных ниже главной диагонали.

29. Сформировать матрицу *K 5x10* Вывести матрицу. Сформировать одномерный массив из минимальных элементов каждой строки матрицы.

30. Ввести с клавиатуры матрицу O 4x5. Вывести ее на экран. Найти среднее арифметическое элементов каждой строки матрицы и вычесть его из элементов этой строки.

31. Сформировать вещественную матрицу A 14x14. Построить одномерный массив P по правилу: если в i-й строке матрицы элемент, принадлежащий главной диагонали, отрицателен, то элемент массива P_i равен сумме положительных элементов i-й строки матрицы, в противном случае P_i равен произведению отрицательных элементов i-й строки матрицы. Вывести массив *P* на экран.

32. Дана целочисленная матрица *MxK*. Найти минимальное значение для верхней половины матрицы и максимальное значение для нижней половины.

33. Целочисленную матрицу K 8x8 заполнить нулями и единицами, расположив их в шахматном порядке. Дополнительно: сделать то же самое для квадратной матрицы произвольного порядка.

Задание 5. Написать программу, определяющую структурный тип данных, согласно задаче Вашего варианта.

Задачи к заданию 5 по вариантам.

1. Определить структурный тип adress (улица, дом, квартира). Ввести указатель на тип adress. Выделить память для хранения информации. Проинициализировать, вывести на экран.

2. Определить структурный тип анкета (поля: фио, адрес, год рождения, группа). Ввести информацию с клавиатуры для группы из трех студентов. Вывести студентов, у которых фамилия начинается на букву «С».

3. Определить структурный тип клиент (поля: фамилия, год рождения, количество покупок, стоимость покупок). Сгенерировать и вывести на экран информацию о клиенте.

4. Определить структурный тип студент (поля: фио (вложенная структура), адрес, год рождения, группа, 10 оценок). Объявить переменную структурного типа студент. Сгенерировать случайным образом информацию для этой переменной. Вывести на экран.

5. Определить структурный тип adress (улица, дом, квартира). Ввести указатель на тип adress. Выделить память для хранения информации. Проинициализировать, вывести на экран.

6. Определить структурный тип анкета (поля: фио, адрес, год рождения, группа). Ввести информацию с клавиатуры для группы из трех студентов. Вывести студентов, у которых фамилия начинается на букву «С».

7. Определить структурный тип клиент (поля: фамилия, год рождения, количество покупок, стоимость покупок). Сгенерировать и вывести на экран информацию о клиенте.

8. Определить структурный тип студент (поля: фио (вложенная структура), адрес, год рождения, группа, 10 оценок). Объявить переменную структурного типа студент. Сгенерировать случайным образом информацию для этой переменной. Вывести на экран.

9. Определить структурный тип adress (улица, дом, квартира). Ввести указатель на тип adress. Выделить память для хранения информации. Проинициализировать, вывести на экран.

10. Определить структурный тип анкета (поля: фио, адрес, год рождения, группа). Ввести информацию с клавиатуры для группы из трех студентов. Вывести студентов, у которых фамилия начинается на букву «С».

11. Определить структурный тип клиент (поля: фамилия, год рождения, количество покупок, стоимость покупок). Сгенерировать и вывести на экран информацию о клиенте.

12. Определить структурный тип студент (поля: фио (вложенная структура), адрес, год рождения, группа, 10 оценок). Объявить переменную структурного типа студент. Сгенерировать случайным образом информацию для этой переменной. Вывести на экран.

13. Определить структурный тип adress (улица, дом, квартира). Ввести указатель на тип adress. Выделить память для хранения информации. Проинициализировать, вывести на экран.

14. Определить структурный тип анкета (поля: фио, адрес, год рождения, группа). Ввести информацию с клавиатуры для группы из трех студентов. Вывести студентов, у которых фамилия начинается на букву «С».

15. Определить структурный тип клиент (поля: фамилия, год рождения, количество покупок, стоимость покупок). Сгенерировать и вывести на экран информацию о клиенте.

16. Определить структурный тип студент (поля: фио (вложенная структура), адрес, год рождения, группа, 10 оценок). Объявить переменную структурного типа студент. Сгенерировать случайным образом информацию для этой переменной. Вывести на экран.

17. Определить структурный тип adress (улица, дом, квартира). Ввести указатель на тип adress. Выделить память для хранения информации. Проинициализировать, вывести на экран.

18. Определить структурный тип анкета (поля: фио, адрес, год рождения, группа). Ввести информацию с клавиатуры для группы из трех студентов. Вывести студентов, у которых фамилия начинается на букву «С».

19. Определить структурный тип клиент (поля: фамилия, год рождения, количество покупок, стоимость покупок). Сгенерировать и вывести на экран информацию о клиенте.

20. Определить структурный тип студент (поля: фио (вложенная структура), адрес, год рождения, группа, 10 оценок). Объявить переменную структурного типа студент. Сгенерировать случайным образом информацию для этой переменной. Вывести на экран.

1.2.2 Требования к содержанию практической части по лабораторной работе № 2

1. При выполнении лабораторной работы для каждого задания требуется написать программу, в которой выполняется формирование определенного типа данных и работа с ними. Задание 1 – статический одномерный массив, задание 2 динамический одномерный массив, задание 3 – строковые данные, задание 4 – статический двумерный массив (таблица), задание 5 – структурный тип.

2. Ввод данных должен осуществляться с клавиатуры через средства интерфейса с учетом требований к входным данным, изложенным в вашей задаче.

3. Вывод результатов должен выполняться на экран (согласно условию решаемой задачи) с помощью интерфейсных средств в соответствующем формате.

1.3. Лабораторная работа № 3 «Односвязные и двусвязные списки»

1.3.1 Задание к лабораторной работе №3

Задание 1. Создайте функции, реализующие операции со списком, необходимым для решения задачи, указанной в вашем варианте индивидуальных заданий (линейный односвязный, линейный двусвязный, циклический односвязный, циклический двусвязный).

Задание 2. Составьте программу решения индивидуальной задачи на основе разработанной динамической структуры данных.

Варианты индивидуальных заданий

1. Описать односвязный линейный список, каждый элемент которого представляет собой запись, состоящую из двух полей: символ и количество его повторений в тексте. Сформировать список из заданного текста. Для создания списка использовать метод вставки в список в отсортированном порядке.

2. Сложение длинных чисел. Используя двусвязные циклические списки сложить два числа: x = 134557952499317879 и y = 79349864365110 . Числа разбить на группы по девять цифр, которые будут являться элементами списка. Результат сложения занести в циклический список.

3. Задача Джозефуса. По кругу стоит группа людей. С терминала вводятся число п и фамилия человека, с которого начинается счет. Начиная с введенной фамилии, по кругу отсчитывается n человек, n-й выбывает из круга и т.д. Выводить фамилии исключаемых из круга людей. Задачу решить, используя односвязный циклический список.

4. Создать два односвязных циклических списка с целочисленными элементами. Переписать в односвязный линейный список элементы с совпадающими значениями.

5. Описать односвязный циклический список символов. Реализовать метод вставки, который выполняет вставку отсутствующего в списке символа в его конец и перемещение в начато элемента с присутствующим в списке значением. Продемонстрировать работу метода.

6. Проверить правильность расстановки скобок (круглых, квадратных, фигурных и угловых) в арифметическом выражении. Для слежения за скобками воспользоваться односвязным циклическим списком.

7. Описать двусвязный линейный список строк. Реализовать метод вставки, который выполняет вставку отсутствующего в списке символа в его конец и перемещение на один элемент к началу списка элемента с присутствующим в списке значением.

8. Из односвязного списка с фамилиями студентов двух групп сформировать два циклических по группам, причем вставка в циклические списки должна выполняться в отсортированном порядке.

9. С использованием односвязного циклического списка определить правильность вложения циклов в программе на языке Бейсик, состоящей из произвольного количества последовательно расположенных или вложенных инструкций:

```
10 FOR x=a TO b STEP c
…
30 NEXT c
```

где `x` — переменная цикла. Для каждого цикла выводить сообщение вида: «`Цикл по x открыт`». «`Цикл по x закрыт`».

10. Создать односвязные списки студентов двух групп с полученными на экзаменах оценками. Сформировать в алфавитном порядке односвязный циклический список студентов, получивших отличные оценки.

11. Сформировать линейный список строк. Исключить из него последние три элемента. Скопировать в двусвязный циклический список вторую половину его элементов, начиная с конца.

12. С использованием односвязного циклического списка определить правильность вложения циклов в программе на языке Бейсик, состоящей из произвольного количества последовательно расположенных или вложенных инструкций:

```
10 FOR x=a TO b STEP c
…
30 NEXT c
```

где x - переменная цикла, один оператор NEXT может завершать одновременно несколько вложенных циклов (например: NEXT x,y,z). Для каждого цикла выводить сообщение вида: «Цикл по x открыт», «Цикл по x закрыт».

13. Описать односвязный циклический список, тип значения элемента которого - запись из двух полей: символ и количество его повторений в тексте. Сформировать список из заданного текста. Переписать в линейный список элементы с русскими буквами.

14. Сформировать два двусвязных циклических списка с элементами - фамилиями клиентов. Удалить из первого списка клиентов, находящихся в двух списках.

15. Сформировать циклический список целых чисел. Скопировать элементы исходного списка в линейный список таким образом, чтобы они были расположены в нем в порядке, обратном к исходному.

16. Сформировать двусвязный список, каждым элементом которого являются сведения о студенте (фамилия, номер группы, год рождения).

Переписать в двусвязный циклический список студентов заданной группы в порядке, обратном их расположению в исходном списке.

17. Сформировать два циклических списка целых чисел. Слить их в один список, так чтобы в нем были только различные элементы.

18. Описать односвязный циклический список, тип значения элемента которого - запись из двух полей: символ русского алфавита и количество его повторений в тексте. Сформировать список из заданного текста. С использованием односвязного линейного списка расположить элементы исходного списка в обратном порядке.

19. С использованием односвязного циклического списка смоделировать работу очереди, создав необходимые подпрограммы - надстройки. Продемонстрировать их работу.

20. Сформировать два списка с фамилиями клиентов. Слить оба списка в циклический, расположив клиентов через одного.

21. Сформировать два односвязных списка целых чисел. Слить их в односвязный циклический список так, чтобы элементы были расположены по возрастанию значений.

22. В односвязном циклическом списке, каждый элемент которого является совокупностью двух целых значений, заменить нулевыми все элементы с заданной суммой.

23. Создать линейные списки студентов двух групп с фамилиями, расположенными в алфавитном порядке (с использованием метода вставки в список в отсортированном порядке). Слить их в один циклический список.

24. По кругу стоит группа людей. С терминала вводятся число n и фамилия человека, с которого начинается счет. Начиная с введенной фамилии, по кругу отсчитывается n человек, n-й выбывает из круга и т.д. Выводить фамилии исключаемых из круга людей. Решить задачу с использованием двусвязного циклического списка,

25. Описать односвязный линейный список, тип значения элемента которого — запись из двух полей: символ и количество его повторений в тексте. Сформировать список из заданного текста. Реализовать метод вставки, выполняющий вставку отсутствующего в списке символа в его конец и перемещение на один элемент к началу списка элемента с присутствующим в списке значением.

26. Сформировать два двусвязных циклических списка с элементами - фамилиями клиентов. Слить эти списки в один, таким образом, чтобы он включал в себя фамилии клиентов поочередно из каждого из исходных списков в обратном порядке.

27. Распределить элементы циклического списка по двум линейным спискам следующим образом: в первый список помещаются элементы с начала циклического списка, а во второй — с его конца.

28. Описать двусвязный список строк и реализовать метод возвращения адреса первого с конца и первого с начала элемента с заданным значением. Исключить второй с начала и второй с конца элемент со значением «дом».

29. Реализовать методы возвращения адреса элемента двусвязного циклического списка по его номеру, начиная от начала и от конца списка. С использованием этих методов поменять местами третий с начала элемент списка с третьим с конца.

1.3.2 Требования к содержанию практической части по лабораторной работе № 3

1. При выполнении лабораторной работы для каждого задания требуется написать программу, в которой выполняется формирование однонаправленного, двунаправленного или циклического списка ***в соответствии с постановкой задачи***, ввод данных элементов списка с учетом разработанного интерфейса, их

обработка и вывод на экран в указанном формате. Для хранения данных списков следует использовать ресурсы динамической памяти.

2. Проверить выполнение основных операций (для заданного типа информационного поля всех элементов):

- просмотр элементов списка (печать списка);

- вставка элемента в список (после элемента с указанным адресом, перед первым элементом, перед элементом с указанным адресом, в конец списка, элемента с указанным номером);

- удаление элемента из списка (после элемента с указанным адресом, первого элемента, перед элементом с указанным адресом, последнего элемента, элемента с указанным номером);

- поиск элемента в списке по значению информационного поля;

- проверка пустоты списка;

- очистка и удаление списка.

3. Ввод данных должен осуществляться с клавиатуры через средства интерфейса или из файла - с учетом требований к входным данным, изложенным в вашей задаче. Ограничениями на входные данные являются максимальный размер строковых данных, диапазоны числовых типов полей структуры и допустимый размер области динамической памяти.

4. Вывод результатов должен выполняться на экран или в файл (согласно условию решаемой задачи) с помощью интерфейсных средств в соответствующем формате.

1.4 Лабораторная работа № 4 «Стеки. Очереди. Деки»

1.4.1 Задание к лабораторной работе №4

Задание 1. Опишите структуру, необходимую для решения задачи, указанной в вашем варианте индивидуальных заданий, и реализуйте функции для выполнения операций над этой структурой.

Задание 2. Составьте программу решения задачи с использованием динамической структуры данных.

Варианты индивидуальных заданий

1. Проверить правильность расстановки скобок в арифметическом выражении (допустимы четыре типа скобок: круглые, квадратные, фигурные и угловые). Для слежения за скобками воспользоваться стеком.

2. Сформировать две очереди с фамилиями клиентов. Слить обе очереди в третью, расположив клиентов через одного.

3. Создать стек, каждый элемент которого является совокупностью двух целых значений. Заменить нулевыми все элементы с заданной суммой значений.

4. Сформировать стек с элементами - строками. Прочитать три нижних элемента стека и поменять местами верхний и нижний элементы.

5. Вычислить значение выражения, записанного в постфиксной форме. Операнды представляют собой переменные с однобуквенным идентификатором или (и) цифры от 0 до 9, и в выражении используются только операции «+», «—», «*», «/».

6. Прочитать из входного файла последовательность слов Pop и Push. Слово Push вталкивается в стек и выводится число элементов стека, при чтении слова Pop выполняется операция исключения элемента из стека. При невозможности выполнения операции Pop вывести нужное сообщение и работу прекратить.

7. Ввести программу на языке Бейсик, состоящую из произвольного числа расположенных последовательно или вложенных инструкций:

```
10 FOR x=a TO b STEP c
40 NEXTx
```

и определить правильность вложения циклов. Для каждой строки выводить сообщение вида: «Цикл по x открыт», «Цикл по x закрыт» или сообщение об ошибке.

8. Преобразовать бесскобочную инфиксную запись выражения в постфиксную. Операнды представляют собой переменные с однобуквенным идентификатором или цифры от 0 до 9, и в выражении используются только операции «+», «—», «*», «/».

9. Сформировать дек целочисленных элементов и преобразовать его таким образом, чтобы первый элемент стал последним, второй - предпоследним и т.д. При преобразовании исключить из дека нулевые элементы.

10. Сформировать две очереди с элементами - фамилиями клиентов. Удалить из второй очереди клиентов, стоящих также и в первой.

11. Ввести программу на языке Бейсик, состоящую из произвольного числа расположенных последовательно или вложенных инструкций:

```
10 FOR x=a TO b STEP c
40 NEXT x и перевести ее в следующую форму:
10 x=a-c 20 x=x+c
50 IF x<b THEN 20
```

Номер строки перехода, переменная цикла и ее конечное значение сохраняются в стеке и выталкиваются из него при достижении строки со словом NEXT.

12. Создать очередь с элементами - строками. Исключить из очереди элементы, начинающиеся с буквы А. Вывести длину получившейся очереди и значения ее первого и последнего элементов.

13. Создать очередь клиентов и распределить клиентов по двум вспомогательным очередям (через одного).

14. Сформировать дек, каждым элементом которого являются сведения о студенте (фамилия, номер группы, год рождения). Переписать в очередь студентов заданного года рождения.

15. Создать дек, каждый элемент которого представляет собой совокупность трех вещественных значений. Исключить из дека все элементы, у которых каждое из трех значений превышает заданное. Вывести длину получившегося дека и значения его первого и последнего элементов.

16. Сформировать очередь для двух стеков вещественных значений таким образом, чтобы вершина первого стека стала началом очереди, а вершина второго - ее концом.

17. Вычислить значение выражения, записанного в префиксной форме. Операнды представляют собой переменные с однобуквенным идентификатором или (и) цифры от 0 до 9, и в выражении используются только операции «+», «—», «*». «/».

18. Сформировать очередь целых чисел. С использованием дека расположить все элементы очереди в обратном порядке.

19. Ввести программу на языке Бейсик, состоящую из произвольного числа расположенных последовательно или вложенных инструкций:

```
10 FOR x=a TO b STEP c
40 NEXTx
```

при условии, что один оператор `NEXT` может завершать одновременно несколько вложенных циклов (например: `NEXT z,y,x`), и определить правильность вложения циклов. Для каждого цикла выводить сообщение вида: «`Цикл по x открыт`», «`Цикл по x закрыт`».

20. Преобразовать выражение в инфиксной форме в префиксную. Операнды представляют собой переменные с однобуквенным идентификатором или (и) цифры от 0 до 9, и в выражении используются только операции «+», «-», «*», «/».

21. Сформировать две очереди с элементами - фамилиями клиентов. Удалить из первой очереди клиентов, стоящих в двух очередях.

22. Исключить из стека строк три нижних элемента, если в вершине содержится текст «Вершина». В противном случае изменить содержимое трех нижних элементов на «Первый», «Второй». «Третий».

23. Создать дек с элементами - двумя целочисленными значениями. Подсчитать количество элементов с заданной суммой значений, прочитать пятый и шестой элементы от начала сформированного дека.

24. Сформировать два дека с фамилиями клиентов. Слить оба дека в третий, расположив клиентов через одного, начиная с конца.

25. Сформировать очередь целых чисел. С использованием стека расположить все элементы исходной очереди в обратном порядке.

26. Создать дек с элементами - фамилиями студентов. Перенести в очередь элементы с фамилиями, начинающимися с букв «А» - «К».

27. Создать дек целочисленных значений и распределить все его элементы по двум стекам через одного, начиная с последнего элемента.

28. Сформировать дек, каждым элементом которого являются сведения о студенте (фамилия, номер группы, год рождения). Переписать в очередь студентов заданной группы.

29. Определить, имеет ли вводимая строка вид xCy, где x - строка, состоящая из букв A и B, а y - строка, обратная строке x. Если да, то определить, имеет ли вводимая строка форму aDbD...Dz, где a, b,..., z - строки вида xCy.

30. Ввести программу на языке Бейсик, состоящую из произвольного числа расположенных последовательно или вложенных инструкций:

```
10 FOR x=a TO b STEP c
40 NEXTx
```

при условии, что один оператор NEXT может завершать одновременно несколько вложенных циклов (например: NEXT z,y,x), и перевести ее в следующую форму:

```
10x=a-c
20 x=x+c
50 IFx<b THEN 20
```

Номер строки перехода, переменная цикла и ее конечное значение сохраняются в стеке и выталкиваются из него при достижении строки со словом NEXT.

1.4.2 Требования к содержанию практической части по лабораторной работе № 4

1. При выполнении лабораторной работы для каждого задания требуется написать программу, в которой выполняется формирование стека, очереди или дека ***в соответствии с постановкой задачи***, ввод данных элементов структуры с учетом разработанного интерфейса, их обработка и вывод на экран (или в файл) в указанном формате. Для хранения данных структур следует использовать ресурсы динамической памяти.

2. Проверить выполнение основных операций (для заданного типа информационного поля всех элементов):

- создание структуры;
- печать (просмотр) структуры;
- вставка элемента в структуру (в вершину (для стека), в конец (для очереди), в начало и в конец (для дека));
- удаление элемента из структуры (из вершины (для стека), из начала (для очереди), из начала и из конца (для дека));
- проверка пустоты структуры;
- очистка и удаление структуры.

3. Ввод данных должен осуществляться с клавиатуры через средства интерфейса или из файла - с учетом требований к входным данным, изложенным в вашей задаче.

4. Вывод результатов должен выполняться на экран или в файл (согласно условию решаемой задачи) помощью интерфейсных средств в соответствующем формате.

1.5 Лабораторная работа № 5 «Бинарные деревья»

1.5.1 Задание к лабораторной работе №5

Задание 1. Опишите структуру (дерево), необходимую для решения задачи, указанной в вашем варианте индивидуальных заданий, и реализуйте ее методы.

Задание 2. Составьте программу решения задачи, указанной в вашем варианте задания.

Варианты индивидуальных заданий

1. Найти дубликаты в списке чисел с использованием дерева поиска.

2. На основе операции обхода сверху вниз реализовать операцию, определяющую, подобны ли два бинарных дерева (два бинарных дерева подобны, если они оба пусты, либо их левые и правые поддеревья подобны).

3. По бесскобочной постфиксной записи арифметического выражения с операндами, записанными в виде строк символов, построить дерево выражения и получить полноскобочное инфиксное выражение.

4. Реализовать операцию определения уровня узла с заданным указателем в бинарном дереве, построить дерево с узлами - символами и определить минимальный и максимальный уровни листа.

5. На основе функции обхода сверху вниз реализовать операцию поиска узла с заданным значением в дереве, не являющемся деревом поиска. Построить дерево минимальной высоты с элементами-символами. Используя функции Brother и Value, найти адрес брата узла с введенным с терминала значением и вывести его значение.

6. Построить дерево поиска с элементами - строками. Используя функции Addr (возвращает адрес первого элемента с заданным ключом), Father (возвращает адрес левого предка элемента с заданным ключом) и Value (возвращает значение информационного поля первого элемента с заданным ключом), найти узел, являющийся самым «молодым» общим предком двух заданных узлов, и вывести его значение.

7. По постфиксной записи арифметического выражения с операндами-строками построить дерево выражения и получить инфиксную запись выражения, содержащую только необходимые скобки.

8. Реализовать дерево поиска, содержащее элементы с одинаковыми ключами. Функция Addr возвращает адрес первого элемента с заданным ключом. Функция Delete исключает первый элемент с заданным ключом.

9. Построить дерево поиска с элементами - вещественными числами. Определить количество элементов дерева на каждом уровне. Удалить элементы с заданными значениями.

10. Построить дерево поиска с элементами - числами. С использованием операций Addr (возвращает адрес первого элемента с заданным ключом) и Delete Left (исключает левое поддерево первого элемента с заданным ключом) найти узел с заданным значением и исключить его левое поддерево. Вывести число узлов в дереве до и после исключения.

11. На основе просмотра текста построить дерево поиска, элементами которого являются символы, а ключами - количества вхождений этих символов в текст. Исключить из дерева символы с заданным числом повторений.

12. Построить дерево поиска с элементами - строками. Исключить из дерева все узлы с заданными словами. При каждом исключении выводить текущее состояние дерева и сообщение о том, каким потомком является исключаемый узел: левым (функция IsLeft) или правым (IsRight).

13. На основе функции обхода дерева снизу вверх реализовать операцию поиска узла с заданным значением в дереве, не являющемся деревом поиска. Из двух последовательностей символов построить два бинарных дерева

минимальной высоты. В первом дереве найти элемент с заданным значением и подключить второе дерево в качестве его левого поддерева, если оно пусто, или левого поддерева первого из его крайних левых потомков, имеющих пустое левое поддерево.

14. На основе функции обхода сверху вниз реализовать операцию, определяющую, являются ли два бинарных дерева зеркально подобными (два бинарных дерева зеркально подобны, если они оба либо пусты, либо левое поддерево каждого дерева подобно правому дереву другого).

15. По бесскобочной постфиксной записи арифметического выражения с операндами - строками построить дерево выражения и получить полноскобочное инфиксное выражение.

16. По бесскобочной префиксной записи арифметического выражения с операндами - строками построить дерево выражения и получить выражение в инфиксной записи, содержащее только необходимые скобки.

17. Реализовать дерево поиска, содержащее элементы с одинаковыми ключами. Функция AddrFirst возвращает адрес первого элемента с заданным ключом. Функция DeleteAH исключает все элементы с заданным ключом.

18. Построить бинарное дерево с элементами - символами. Вывести элементы дерева по уровням.

19. Реализовать дерево выражения с операндами - числами и написать метод вычисления значения арифметического выражения. По бесскобочной префиксной записи построить дерево выражения и вычислить его значение.

20. Реализовать операцию определения уровня узла с заданным указателем в дереве поиска, построить дерево с заданными числовыми значениями узлов и определить минимальный и максимальный уровни листа.

21. На основе функции обхода дерева слева направо реализовать операцию поиска узла с заданным значением в дереве, не являющемся деревом поиска. Построить дерево минимальной высоты с числовыми значениями узлов. Используя необходимые операции, удалить из дерева поддерево, корнем которого является отец символа с заданным значением.

22. Построить два дерева поиска для студентов двух групп с полученными на экзаменах оценками. Сформировать дерево для студентов, получивших отличные оценки, причем расположить их в дереве по алфавиту.

23. Реализовать дерево поиска, содержащее элементы с одинаковыми ключами. Функция AddrLast возвращает адрес последнего элемента с заданным ключом. Функция DeletcFirst исключает первый элементе заданным ключом.

24. С использованием дерева поиска удалить из заданного текста дубликаты слов.

25. Построить бинарное дерево с элементами - словами. Сформировать предложение из слов дерева, получающееся при обходе слева направо. Удалить поддерево, начинающееся с заданного слова.

26. Построить дерево поиска с элементами - целыми числами. Удалить из дерева элементы с заданным значением, а все отрицательные элементы заменить нулями.

27. Построить дерево поиска с заданными числовыми значениями. С использованием функция (возвращает адрес первого элемента с заданным ключом), Father (возвращает адрес левого предка элемента с заданным ключом) найти узел с заданным значением и удалить из дерева поиска отца этого узла.

28. Построить дерево поиска с элементами - символами. Определить число повторяющихся символов в дереве и удалить дубликаты.

29. Реализовать дерево поиска, содержащее элементы с одинаковыми ключами. Функция AddrLast возвращает адрес последнего элемента с заданным ключом. Функция DclctcAll исключает все элементы с заданным ключом.

30. На основе просмотра текста построить дерево поиска, элементами которого являются символы, а ключами - количества вхождений этих символов в текст. Исключить из дерева символы с одинаковым числом повторений, используя строку для хранения последовательности узлов дерева (результатов обхода)

1.5.2 Требования к содержанию практической части лабораторной работы № 5

1). При выполнении лабораторной работы для каждого задания требуется написать программу, в которой выполняется формирование структуры данных в соответствии с постановкой задачи, ввод данных элементов структуры с учетом разработанного интерфейса, их обработка и вывод на экран (или в файл) в указанном формате. Для хранения данных структур следует использовать ресурсы динамической памяти.

2. Проверить выполнение основных операций (для заданного типа информационного поля всех элементов):

- создание структуры;
- печать (просмотр) структуры;
- обход узлов сверху вниз, слева направо и снизу вверх;
- добавление поддерева;
- исключение поддерева;
- проверка пустоты структуры;
- очистка и удаление структуры.

3. Ввод данных должен осуществляться с клавиатуры через средства интерфейса или из файла - с учетом требований к входным данным, изложенным в вашей задаче.

4. Вывод результатов должен выполняться на экран или в файл (согласно условию решаемой задачи) помощью интерфейсных средств в соответствующем формате.

1.6 Лабораторная работа № 6 «Ориентированные графы»

1.6.1 Задание к лабораторной работе №6

Задание 1. Опишите структуру (ориентированный граф), необходимую для решения задачи, указанной в вашем варианте индивидуальных заданий, и реализуйте ее методы. Исходный граф может содержать петли и циклы.

Задание 2. На основе разработанной структуры в задании 1, составьте программу решения задачи, указанной в вашем варианте индивидуального задания.

Варианты индивидуальных заданий

1. Определить кратчайший путь между узлами N1 и N2 и его длину во взвешенном графе.

2. Задана карта дорог в виде графа. Найти города, расположенные на расстоянии более X от заданного узла.

3. Определить, является ли один граф подграфом другого.

4. Реализовать граф с помощью односвязных циклических списков (включение и исключение узла).

5. Найти все узлы графа, к которым можно добраться из заданного узла.

6. Определить кратчайший путь между узлами N1 и N2 и его длину во взвешенном графе.

7. Определить, является ли граф несвязным.

8. Составить список узлов, расстояние от которых до заданного узла не превышает величину X.

9. Определить, является ли один граф суграфом другого.

10. Реализовать граф с помощью односвязных циклических списков (определение начального и конечного узлов дуги).

11. Найти все узлы графа, из которых можно добраться до заданного узла.

12. Определить расстояние между узлами N1 и N2 во взвешенном циклическом графе.

13. Определить, является ли один граф частью другого.

14. На основе обхода в ширину найти все узлы заданного графа, недостижимые из заданного узла.

15. На основе поиска в ширину определить, является ли граф несвязным.

16. Перечислить узлы графа по возрастанию расстояния до них от заданного узла.

17. Реализовать графе помощью двусвязных списков (включение и исключение узла).

18. Задана карта дорог в виде графа. Найти города, расположенные на расстоянии X от заданного узла.

19. На основе обхода списка узлов найти узел с заданным значением в графе и исключить его.

20. Определить, является ли граф полным.

21. Найти узел взвешенного графа, ближайший к заданному узлу.

22. Определить, является ли один граф дополнением другого.

23. На основе поиска в ширину определить, является ли граф связным.

24. Найти все циклы в графе.

25. Задана карта дорог в виде графа. Найти города, расположенные на расстоянии не более X от заданного узла.

26. На основе обхода в глубину найти все узлы заданного графа, недостижимые из заданного узла.

27. Найти все узлы графа, к которым можно добраться из заданного узла по пути не длиннее заданного значения.

28. Определить длины всех путей между узлами N1 и N2 в невзвешенном графе.

29. Реализовать граф с помощью циклических списков (включение и исключение дуги, определение числа узлов).

30. Найти узел графа, наиболее удаленный от заданного узла.

1.6.2 Требования к содержанию практической части лабораторной работы № 6

1. При выполнении лабораторной работы для каждого задания требуется написать программу, в которой выполняется формирование структуры данных (орграф) в соответствии с постановкой задачи, ввод данных элементов структуры с учетом разработанного интерфейса, их обработка и вывод на экран (или в файл)

в указанном формате. Для хранения данных структур следует использовать ресурсы динамической памяти.

2. Проверить выполнение основных операций (для заданного типа информационного поля всех элементов):

- создание графа;
- просмотр (печать) графа;
- включение дуги между узлами N1 и N2;
- исключение дуги между узлами N1 и N2;
- включение узла со значением v в граф;
- исключение из графа узла N с исключением инцидентных ему дуг и возвращение значения информационного поля узла;
- проверка графа на пустоту;
- возвращение адреса узла со значением v;
- проверка смежности узлов N1 и N2;
- возвращение начального узла дуги Arc;
- возвращение конечного узла дуги Arc;
- очистка графа.

3. Ввод данных должен осуществляться с клавиатуры через средства интерфейса или из файла - с учетом требований к входным данным, изложенным в вашей задаче.

4. Вывод результатов должен выполняться на экран или в файл (согласно условию решаемой задачи) с помощью интерфейсных средств в соответствующем формате.

2 АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ

2.1. Лабораторная работа № 7 «Сортировка данных»

2.1.1 Задание к лабораторной работе №7

Задание 1. Составить программу сортировки заданной последовательности, используя алгоритм в соответствии с вариантом.

Исходные, промежуточные и итоговые данные хранить в виде соответствующих структур данных (внутренняя сортировка).

Варианты для выполнения задания 1

Вариант	Метод сортировки
1	Модифицированный метод подсчета
2	Метод выбора из дерева
3, 4*	Метод простых вставок
5	Метод бинарных вставок
7	Метод двухпутевых вставок
8, 9*	Сортировка с вычислением адреса по старшей цифре
10	Пирамидальная сортировка
11	Метод обмена с запоминанием количества обменов
12	Шейкер-сортировка
13	Метод четных и нечетных транспозиций
14, 15*	Метод простого выбора
16	Быстрая сортировка
17*	Сортировка простым слиянием
18, 6*	Сортировка с вычислением адреса по младшей цифре
19	Вставки в список
20	Метод подсчета
21	Метод обмена с запоминанием адреса последнего обмена

Задание 2. Составить программу сортировки заданной последовательности, используя алгоритм в соответствии с вариантом. Исходные, промежуточные и итоговые данные хранить в файлах (внешняя сортировка).

Варианты для выполнения задания 2

Вариант	Задание
1, 4, 7, 10, 13, 16, 19	Дан полный перечень всех стран, который включает в себя название, континент, столицу, площадь, численность населения. Указать сведения о государствах заданного континента в порядке возрастания численности населения. Использовать двухпутевое однофазное простое слияние.
2, 5, 8, 11, 14, 17, 20	Даны сведения о химических веществах, которые включают в себя: класс вещества; название вещества; молекулярную массу вещества. Упорядочить по возрастанию молекулярных масс все вещества указанного класса. Использовать двухпутевое двухфазное естественное сбалансированное слияние.
3, 6, 9, 12, 15, 18	В файле хранится последовательность русских слов. Упорядочить ее в алфавитном порядке. Использовать внешнюю сортировку. Учесть, что порядок кодов букв русского алфавита не соответствует порядку букв в алфавите.

2.1.2 Требования к содержанию практической части лабораторной работы № 7

Задание 1.

Программу необходимо выполнить с двумя вариантами исходных данных:

• отладочный (10 элементов), демонстрирующий результаты всех проходов алгоритма сортировки;

• рабочий (50 элементов), в результате обработки которого выводятся отсортированная последовательность, а также значения параметров эффективности: число сравнений ключей *C* и число перестановок элементов *P* (или число переключений связей V для списка).

Исходные данные и результаты работы программы (демонстрационные и рабочие) разместить в текстовых файлах.

В вариантах с нечетными номерами сортируются записи, состоящие из двух полей: строки и числового ключа.

В вариантах с четными номерами сортируется по алфавиту последовательность строковых переменных.

В вариантах, отмеченных звездочкой, исходная последовательность организована в виде связанной структуры — списка. В остальных вариантах сортируемая последовательность — массив.

В программе необходимо рассчитать характеристики сложности реализуемых методов. Выводы о соответствии полученных практических оценок теоретическим **внести в отчет.**

Для создания таблицы рекомендуется создать текстовый файл, содержащий порядка 100 элементов.

Сначала выполнить работу в отладочном (демонстрационном) режиме, для чего прочитать из этого файла первые 10 элементов, создать таблицу и отсортировать ее, демонстрируя все шаги алгоритма (индексы и результаты сравнений).

Затем выполнить работу в рабочем режиме. Для этого открыть тот же файл с исходными данными для чтения, прочитать из него все элементы, создать таблицу и отсортировать ее, подсчитывая количество операций сравнения ключей и перестановок на каждом шаге сортировки. В файл результатов вывести отсортированную таблицу и результаты исследования процедуры сортировки (число сравнений ключей и перестановок,

потребовавшихся для сортировки). Результаты сравнить с теоретическими характеристиками сложности исследуемого метода.

Задание 2

Реализовать программу, в которой выполняется алгоритм внешней сортировки двухпутевым двухфазным простым слиянием или алгоритм внешней сортировки двухпутевым двухфазным естественным слиянием, в соответствии с заданием варианта.

Программа должна иметь интерфейсную часть, обеспечивающую ввод исходных данных и вывод результатов в файл.

В программе необходимо рассчитать характеристики сложности реализуемых методов. Выводы о соответствии полученных практических оценок теоретическим **внести в отчет.**

2.2. Лабораторная работа № 8 «Поиск и расстановка»

2.2.1 Задание к лабораторной работе №8

Задание 1. Составить программы формирования заданными методами (в каждом варианте 2 метода) таблицы поиска и последующего поиска в этой таблице элементов из входной последовательности.

Варианты индивидуальных заданий

Вариант	**Метод поиска**	**Метод устранения конфликтов при расстановке**
1,2*,13, 19*	Линейный поиск с барьером	
3*,9*,14*,20*	Метод перемещения в начало	
4,5*,10,15*, 21	Метод транспозиции	
6,7*,11*,16	Индексно-последовательный поиск	
8,12,17,18	Бинарный поиск	
1, 13	Расстановка методом деления	Линейное опробование

2, 14	Расстановка методом деления	Случайное опробование
3, 15	Расстановка методом деления	Двойная расстановка
4, 16	Расстановка методом середины квадрата	Двойная расстановка
5,12	Расстановка методом середины квадрата	Раздельное сцепление
6, 17	Расстановка методом свертки	Внутреннее сцепление
7, 18	Расстановка методом свертки	Раздельное сцепление
8, 19	Расстановка методом свертки	Двойная расстановка
9, 20	Расстановка граничной свертки	Двойная расстановка
10, 21	Мультипликативная функция	Внутреннее сцепление
11	Преобразование системы счисления	Двойная расстановка

2.2.2 Требования к содержанию практической части лабораторной работы № 8

Программу необходимо выполнить с двумя вариантами исходных данных:

• отладочный (10 элементов), демонстрирующий работу алгоритма поиска;

• рабочий (50 элементов), в результате обработки которого выводятся значения параметров эффективности, определенных для заданного метода поиска.

Исходные данные и результаты работы программы (демонстрационные и рабочие) разместить в текстовых файлах.

В вариантах с нечетными номерами каждый элемент — строковая переменная.

В вариантах с четными номерами элемент состоит из двух полей: строковой переменной и числового ключа.

При расстановке строковые ключи необходимо предварительно свернуть каким-либо методом.

В вариантах, отмеченных звездочкой, таблица организована в виде списка, в остальных вариантах таблица — массив.

Для создания таблицы и последующего поиска в ней рекомендуется создать текстовый файл, содержащий порядка 100 элементов.

Сначала выполнить работу в отладочном (демонстрационном) режиме, для чего прочитать из этого файла первые 10 элементов и создать таблицу. Ввести в диалоговом режиме значение элемента, присутствующего в таблице, и выполнить его поиск, демонстрируя все шаги алгоритма (индексы и результаты сравнений). Затем ввести значение элемента, которого нет в таблице, и выполнить поиск с демонстрацией шагов алгоритма. Результаты поиска вывести в текстовый файл.

Затем выполнить работу в рабочем режиме. Для этого открыть тот же файл с исходными данными для чтения, промигать из него первые 50 элементов и создать таблицу. Затем вновь открыть этот файл для чтения, последовательно прочитать из него и выполнить поиск всех 100 элементов, подсчитывая количество операций сравнения ключей на каждом шаге поиска. При этом в файл результатов вывести только результаты исследования процедуры поиска (число сравнений ключей, потребовавшихся для поиска).

Характеристики эффективности поиска, полученные с помощью разработанной программы, сравнить со значениями, вычисленными по приведенным для заданных методов формулам.

Чтобы правильно оценить эффективность поиска, необходимо, чтобы во второй половине текстового файла содержались элементы, отсутствующие в таблице.

Для методов разрешения конфликтов при расстановке необходимо отдельно оценивать эффективность успешного и неуспешного поиска.

Вывод о соответствии характеристик эффективности поиска,

полученные с помощью разработанной программы значениям, вычисленным по соответствующим для заданных методов формулам **внести в отчет**.

2.3. Лабораторная работа № 9 «Эвристические алгоритмы»

2.3.1 Задание к лабораторной работе №9

Задание 1. Изучить теоретическую часть работы.

Задание 2. Реализовать алгоритм, согласно варианту.

Задание 3. Для функций двух видов: вогнутой и с вторичными минимумами применить реализованный метод, оценить скорость сходимости (для генетического алгоритма провести оценку сходимости в зависимости от размера начальной популяции и скорости мутаций) и возможность нахождения глобального минимума.

Варианты индивидуальных заданий

1,4,7,10,13,16,19	Метод градиентного спуска
2,5,8,11,14,17,20	Метод моделирования отжига
3,6,9,12,15,18,21	Генетический алгоритм

Примеры функций для тестирования работы программы

№	Целевая функция	Глобальный минимум
1	$f(x)=\sin(x)+\sin\left(x^2\right)$	
2	$f(x)=\dfrac{6\cdot\sqrt[3]{6\left(x-3\right)^2}}{\left(x-1\right)^2+8}$	
3	$f(x)=-2x-8+3\cdot\sqrt[3]{6\left(x+4\right)^2}$	
4	$f(x)=\sqrt{3+x^2}+3\cos x$	
5	$f(x,y)=100\left(y-x^2\right)^2+\left(1+x\right)^2$ - функция Розенброка	$f(1,1)=0$

6	$f(x,y)=\left(y-x^2\right)\left(\frac{5,1}{4\pi^2}+x\frac{5}{\pi}\right)+10\left(1-\frac{1}{8\pi}\right)\cos x+10$	0,397887
7	$f(x,y)=20+e-20\exp\left(-0,2\sqrt{0,5\left(x^2+y^2\right)}\right)-\exp\frac{1}{2\left(\cos\left(2\pi x+\cos 2\pi y\right)\right)}$	0
8	$f(x,y)=\lvert x\rvert^2+\lvert y\rvert^3$	0
9	$f(x,y)=102+\left(x^2-10\cos(2\pi x)\right)+\left(y^2-10\cos(2\pi y)\right)$	0

2.3.2. Требования к содержанию практической части лабораторной работы № 9

Метод градиентного спуска.

1. Провести анализ метода градиентного спуска. Для чего построить графики зависимости точности решения от числа итераций и сопоставить эти графики для функций различных типов. Для этого необходимо последовательно выполнить следующие действия.

1.1. Реализовать метод градиентного спуска и описать детали выбранной реализации.

1.2. Для нескольких вогнутых (обладающих единственным минимумом) функций построить графики зависимости $\left|x_i-x^*\right|$ (где x^* - истинное положение глобального минимума) от номера итерации i. Выбранные для исследования функции должны различаться средним значением модуля градиента (крутизной). Следует обратить внимание на выбор начальной точки которая должна отличаться от искомого минимума.

1.3. Для нескольких функций с многими локальными минимумами определить условия сходимости к глобальному минимуму. Определить, с какой вероятностью метод сходится к глобальному минимуму при случайном выборе (из некоторого фиксированного диапазона) начальной

точки. Для исследования следует брать функции с разным числом минимумов.

1.4. Проанализировать полученные результаты. Определить предпочтительность использования метода при поиске минимума функции с одним и многими минимумами, опираясь на скорость сходимости и на легкость нахождения глобального минимума.

2. Сделать выводы по работе. Оформить отчет.

Метод моделирования отжига

1. Провести анализ метода моделирования отжига. Для чего построить графики зависимости точности решения от числа итераций и сопоставить эти графики для функций различных типов. Для этого необходимо последовательно выполнить следующие действия.

1.1. Реализовать метод моделирования отжига и описать детали выбранной реализации.

1.2. Для нескольких вогнутых (обладающих единственным минимумом) функций построить графики зависимости $\left|x_i - x^*\right|$ (где x^* - истинное положение глобального минимума) от номера итерации i. Выбранные для исследования функции должны различаться средним значением модуля градиента (крутизной). Следует обратить внимание на выбор начальной точки которая должна отличаться от искомого минимума. В методе моделирования отжига следует также обратить внимание на задание начальной температуры и исследовать ее влияние на скорость сходимости.

1.3. Для нескольких функций с многими локальными минимумами определить условия сходимости к глобальному минимуму. Определить, какая начальная температура обеспечивает сходимость к глобальному минимуму. Для исследования следует брать функции с разным числом минимумов.

1.4. Проанализировать полученные результаты. Определить предпочтительность использования метода при поиске минимума функции с одним и многими минимумами, опираясь на скорость сходимости и на легкость нахождения глобального минимума.

2. Сделать выводы по работе. Оформить отчет

Генетические алгоритмы и эволюционные стратегии поиска

1. Проанализировать зависимость характеристик работы эволюционных стратегий (скорости их сходимости и возможности обнаружения глобального экстремума фитнесс-функции) от установочных параметров - размера начальной популяции, скорости и типа мутаций, типа скрещивания и способов отбора родительских особей, а также от вида оптимизируемой функции. Для этого необходимо выполнить следующую последовательность действий.

1.1. Выполнить реализацию эволюционной стратегии и описать ее детали.

1.2. Для нескольких выпуклых (обладающих единственным максимумом) функций построить графики зависимости $\left|x_i - x^*\right|$ (где x^* - истинное положение глобального минимума, x_i – максимальное по текущей популяции значение фитнесс-функции) от номера популяции i при различных значениях скорости мутации и размера начальной популяции. Следует обратить внимание на выбор диапазона $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$. Графики необходимо построить для двух случаев: когда точка глобального экстремума попадает в этот диапазон и когда она находится вне него.

1.3. Для нескольких функций с многими локальными экстремумами провести те же исследования, что и для функций с единственным экстремумом.

1.4. По результатам проведенных экспериментов, построить таблицы и графики зависимостей погрешности результата от размера начальной популяции, скорости и типа мутаций, при различных установочных параметрах: типа скрещивания, способов отбора родительских особей. Проанализировать полученные результаты.

1.5. Определить, какое именно влияние оказывает скорость мутаций на функционирование алгоритма поиска. Определить, может ли большая скорость мутаций быть заменена большим размером популяции.

1.6. Установить различия в характере работы алгоритма поиска для функций с единственным и многими экстремумами.

2. Сделать выводы по работе. Оформить отчет

Теоретическая часть

Метод градиентного спуска.

Метод градиентного спуска - это классический метод поиска минимума дифференцируемой функции с аргументами, принимающими вещественные значения. Данный метод, как правило, применяется для многомерных функций, поскольку в одномерном случае существуют более эффективные методы поиска.

Как известно, градиент некоторый функции $f(x_1, x_2, \ldots, x_n)$ в некоторой точке показывает направление локального наискорейшего увеличения функции. Этот факт используется в методах градиентного спуска (подъема).

Эти методы описываются следующей последовательностью действий:

1. Установить номер итерации: i=0. Выбрать начальную точку $\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right)$.

2. Для текущей точки определить значение градиента:

$$\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) = \left(\frac{\partial f}{\partial x_1}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right), \frac{\partial f}{\partial x_2}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right), ..., \frac{\partial f}{\partial x_n}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right)$$

В случае если градиент не может быть вычислен аналитически, его компоненты могут быть оценены:

$$\frac{\partial f}{\partial x_j}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) \approx \frac{f\left(x_1^{(i)}, ..., x_{j-1}^{(i)}, x_j^{(i)} + \Delta x, x_{j+1}^{(i)}, ..., x_n^{(i)}\right) - f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\Delta x}$$

3. Определить положение следующей точки:

$$\left(x_1^{(i+1)}, x_2^{(i+1)}, ..., x_n^{(i+1)}\right) = \left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - d\frac{\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\left|\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right|}.$$

где d - параметр, определяющий скорость спуска, и положить i=i+1.

4. Перейти к шагу 2. если не выполнен критерий останова.

Существует несколько способов ввода критерия останова.

1) Наложить ограничение на количество итераций.

2) Проверять, что текущая точка или значение функции f меняются мало.

2.1.) При фиксированном шаге d изменение положения текущей точки происходит всегда на одну и ту же величину. В этом случае можно проверять изменение за несколько итераций и сравнивать с d:

$$\frac{\left|\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - \left(x_1^{(i-k)}, x_2^{(i-k)}, ..., x_n^{(i-k)}\right)\right|}{kd}.$$

3) Адаптивный выбор шага d. Для этого на каждой итерации осуществляется выбор такого значения из $d_0 = d, d_1 = \frac{d}{w}, d_2 = d \cdot w$ (где w - некоторый параметр, как правило $w \in (1,2]$), что значение функции в точке

$$\left(x_1^{(i+1)}, x_2^{(i+1)}, ..., x_n^{(i+1)}\right) = \left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - d \frac{\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\left|\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right|}$$

минимально.

Таким образом, если при большом d метод градиентного спуска «проскакивает» минимум, то d будет уменьшаться. Уменьшение d ниже заданного порога также служит критерием остановки. Напротив, на пологих участках значение d будет увеличиваться.

При условии существования глобального минимума функции метод градиентного спуска обычно сходится (за исключением случаев, когда вдоль некоторого направления функция, монотонно убывая, стремится к некоторому конечному пределу при $|x| \to \infty$). Сходимость метода обеспечивается тем, что на каждой итерации выбирается такая точка $\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)$, что $f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) < f\left(x_1^{(i-1)}, x_2^{(i-1)}, ..., x_n^{(i-1)}\right)$. Метод, однако, не гарантирует нахождения глобального минимума, поскольку при достижении любого локального минимума метод не в состоянии определить направление на более глубокий минимум (и вообще обнаружить его существование) и останавливается в соответствии с выбранным критерием останова.

В связи с этим, выбор начальной точки может существенным образом сказываться на получаемом результате.

Метод моделирования отжига

Метод моделирования отжига предназначен для поиска глобального минимума некоторой функции $f : S \to R$, где S - некоторое пространство (необязательно непрерывное), элементы которого интерпретируются как состояния некоторой воображаемой физической системы, а значения самой

функции - как энергия этой системы $E=f(x_1, x_2, \ldots, x_n)$ в состоянии $(x_1, x_2, \ldots, x_n) \in S$.

В методе моделирования отжига система в каждый момент времени находится в некотором состоянии $\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$ а также обладает некоторой температурой T, которая является управляемым параметром.

На каждой итерации система случайным образом переходит в новое состояние $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$. Механизм выбора нового состояния состоит из двух частей:

1. Выбирается $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ в соответствии с некоторой функцией распределения $g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right)$. Как правило, эта функция зависит только от расстояния $\left|\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)\right|$, причем с увеличением этого расстояния вероятность перехода понижается.

2. После случайного выбора $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ проверяется вероятность перехода в это новое состояние, исходя из разности энергий и текущей температуры: $h(\Delta E, T)$,

$$\Delta E = f\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - f\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right).$$

Здесь $h(\Delta E, T)$ показывает вероятность перехода в состояние с другой энергией. Проверка производится следующим образом: выбрасывается случайное число из диапазона [0, 1]. Если это число оказывается меньше, чем значение вероятности $h(\Delta E, T)$, то новое состояние $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ принимается, в противном случае шаг 1 повторяется. Функция $h(\Delta E, T)$, как правило, стремится к 1 при $\Delta E \to -\infty$ и стремится к 0

при $\Delta E \to +\infty$ (то есть предпочтение в среднем отдается состояниям с меньшей энергией).

Поскольку метод моделирования отжига базируется на физических принципах, то и функции распределения вероятностей $g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right)$ и $h\left(\Delta E, T\right)$ также часто заимствуются из физики. В частности, достаточно популярен больцмановский отжиг, в котором:

$$g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right) =$$

$$= \left(2\pi T\right)^{-\frac{n}{2}} \exp\left(\frac{-\left| \left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right) \right|^2}{2T} \right)$$

$$h\left(\Delta E, T\right) = \frac{1}{1+\exp\left(\frac{\Delta E}{T}\right)} \approx \frac{-\Delta E}{T}.$$

Таким образом, температура T определяет, насколько в среднем может меняться текущее состояние $\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$, а также то, насколько в среднем может меняться энергия системы при переходе в новое состояние.

Поскольку переход в состояния с меньшей энергией более вероятен. чем переход в состояния с более высокой энергией, то система будет больше времени проводит именно в низкоэнергетических состояниях.

Чтобы обеспечить сходимость системы к некоторому состоянию с наименьшей энергией, температуру системы понижают с переходом к следующей итерации. В больцмановском отжиге применяется следующий закон понижения температуры:

$$T_i = \frac{T_0}{\ln\left(1+i\right)},$$

где номер итерации $i>0$. Такой закон может, однако, потребовать большое число итераций, особенно при больших значениях начальной температуры T_i - в связи с чем используется более быстрое понижение температуры:

$$T_i = \frac{T_0}{1+i}.$$

Начальная температура неявно задает область, в которой будет осуществляться поиск глобального минимума, а также определяет необходимое для сходимости число итераций.

Генетические алгоритмы и эволюционные стратегии поиска

Генетические алгоритмы (ГА) предназначены для нахождения экстремумов функций от произвольных объектов, используя при этом приемы, заимствованные у естественной эволюции. Сами объекты трактуются как некоторые организмы, а оптимизируемая функция - как приспособленность организмов, или фитнесс-функция. Множество возможных объектов взаимно однозначно отображается на некоторое подмножество множества битовых строк (обычно фиксированной длины). Эти строки трактуются как хромосомы (геномы).

Особенность генетических алгоритмов заключается в том, что они работают с битовыми строками, не опираясь на структуру исходных объектов, что позволяет применять ГА без модификации для любых объектов. Единственно, что требуется для такого применения. - это перекодирование объектов в геномы.

Эволюционные стратегии отличаются от генетических алгоритмов лишь в том, что в них используются структурированные описания объектов, то есть такие описания, элементы которых имеют вполне четкий смысл в той предметной области, к которой относятся данные объекты.

Двумя основными механизмами эволюции, наиболее часто моделируемыми в генетических алгоритмах и эволюционных стратегиях, являются скрещивание и мутации. При этом схема эволюционных методов поиска выглядит следующим образом.

1. Сгенерировать начальную популяцию (случайную совокупность объектов).

2. Выбрать родительские пары.

3. Для каждой родительской пары с использованием оператора скрещивания породить потомство.

4. В хромосомы порожденного потомства внести случайные искажения оператором мутации.

5. Произвести отбор особей из популяции по значению их фитнесс-функции.

6. Повторять шаги 2-4, пока не выполнится критерий остановки.

Рассмотрим каждый из шагов чуть подробнее.

1. Генерация начальной популяции обычно производится равномерно по пространству генов (или по пространству описаний объектов). Размер популяции - установочный параметр.

2. Выбор родительских пар может осуществляться различными способами. Выбор родителей осуществляется в два этапа: выбор первого родителя и формирование пары. При выборе одного родителя обычно используется один из следующих способов:

- с равной вероятностью выбирается любая особь из имеющейся популяции:

- особь выбирается случайно с вероятностью, пропорциональной значению фитнесс-функции: то есть в этом случае значение фитнесс-

функции сказывается не только на том, какие особи останутся в популяции в результате отбора, но и на то, сколько потомства они произведут.

Выбор второго родителя осуществляется по одному из следующих критериев:

- независимо от уже выбранного родителя (то есть второй родитель выбирается абсолютно так же, как и первый),

- на основе ближнего родства,

- на основе дальнего родства.

В последних двух случаях выбор одного родителя влияет на выбор другого родителя: с большей вероятностью формируются пары, состоящие из особей, которые больше похожи друг на друга (то есть ближе находятся в пространстве геномов или описаний объектов) в случае использования ближнего родства и меньше похожи - в случае дальнего родства. В генетических алгоритмах в качестве меры близости обычно используется *расстояние Хемминга*.

3. Оператор скрещивания - это оператор, который определяет, как из хромосом родителей формировать хромосомы их потомства. Часто применяется следующий оператор скрещивания: хромосомы делятся в некоторой случайной точке и обмениваются этими участками (то есть. все. что идет до этой точки, берется от одного родителя, а все, что после, - от другого). Это одноточечный кроссинговер. В многоточечном кроссинговере таких участков обмена больше.

При равномерном скрещивании каждый бит хромосомы берется от случайного родителя.

4. Мутации обычно осуществляются как случайная замена одного бита хромосомы. Скорость мутаций выражается в том, как часто они осуществляются. Это управляемый параметр, который влияет на скорость сходимости и вероятность попадания в локальный экстремум.

5. Отбор особей в новую популяцию чаще всего осуществляется одной из двух стратегий:

- пропорциональный отбор, при котором вероятность того, что некая особь останется в следующей популяции, пропорциональна значению фитнесс-функции этой особи:

- элитный отбор, при котором из популяции отбираются лучшие по значению фитнесс-функции особи, и они детерминированным образом переходят в следующую популяцию.

Формирование новой популяции может осуществляться как на основе потомков и родителей, так и на основе только потомков в зависимости от конкретной реализации.

6. Основные критерии останова базируются либо на числе сменившихся поколений (количестве выполненных итераций), либо на некотором условии стабильности популяции. Число поколений не является адаптивным по отношению к виду фитнесс-функции. поэтому используется обычно в качестве не основного критерия. Проверка стабильности популяции в общем виде, как правило, требует значительных вычислений, поэтому чаше используется проверка того, что максимальное по популяции значение фитнесс-функции перестает заметно расти от поколения к поколению. Все эти критерии останова соответствуют критериям останова в методе градиентного спуска, но учитывают ту специфику генетических алгоритмов, что в них на каждой итерации одновременно рассматривается не одно, а несколько решений.

Особенности реализации генетических операторов в эволюционных стратегиях.

Рассмотрим особенности реализации генетических операторов в эволюционных стратегиях на примере объектов, описаниями которых являются двухкомпонентные векторы: (x, y).

1. Генерация начальной популяции может осуществляться путем выбора случайных векторов из области $[x_{\min}, x_{\max}]\times[y_{\min}, y_{\max}]$, где величины $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ задают ожидаемые минимальные и максимальные значения переменных x и y искомого положения экстремума фитнесс-функции.

2. При выборе родителей особенность эволюционных стратегий выражается в способе задания меры родства. В данном случае, мерой родства двух особей (x_1, y_1) и (x_2, y_2) может служить евклидово расстояние: $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$.

3. Результатом скрещивания двух особей в рассматриваемом случае будет являться особь $\left(\varepsilon x_1+(1-\varepsilon)x_2, \varepsilon y_1+(1-\varepsilon)y_2\right)$, где $\varepsilon\in[0,1]$ - случайная величина.

4. Результатом мутации для особи (x, y) будет являться особь $\left(x+\delta_x, y+\delta_y\right)$, где δ_x, δ_y - случайные величины. Их распределение вероятностей может быть выбрано гауссовым или, для простоты программной реализации, равномерным в некотором интервале. Дисперсия этих величин определяет скорость мутаций.

5 и 6. Операторы отбора и критерии останова в эволюционных стратегиях не имеют особых отличий от тех, которые используются в генетических алгоритмах.

3. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА И ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ

Отчет оформляется индивидуально каждым студентом в виде документа, содержащего:

1) титульный лист;

2) текст индивидуального задания, включающий требования к практической части;

3) алгоритмы решения в виде блок-схемы;

4) тексты программ на языке программирования;

5) контрольные примеры в моменты ввода исходных данных и получения результатов.

6) необходимые выводы по проведенному исследованию.

При защите лабораторной работы студент должен:

1) представить отчет;

2) продемонстрировать работающие программы;

3) ответить на вопросы преподавателя.

4. РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

4.1 Основная литература

1. Белов, В.В. Алгоритмы и структуры данных [Электронный ресурс]: Учебник / В.В. Белов, В.И. Чистякова - М.:КУРС, НИЦ ИНФРА-М, 2017. - 240 с. Режим доступа: http://znanium.com/catalog.php?bookinfo=766771

2. Гладков, Л. А. Генетические алгоритмы [Электронный ресурс]: Учебное пособие / Под ред. В. М. Курейчика. - 2-е изд., исправл. и доп. - М.: ФИЗМАТЛИТ, 2010. - 368 с. Режим доступа: http://znanium.com/catalog.php?bookinfo=544626

4.2 Дополнительная литература

1. Колдаев, В.Д. Структуры и алгоритмы обработки данных

[Электронный ресурс]: Учебное пособие / В.Д. Колдаев. - М.: ИЦ РИОР: НИЦ ИНФРА-М, 2014. - 296 с. Режим доступа: http://znanium.com/catalog.php?bookinfo=418290

2. Вирт, Н. Алгоритмы и структуры данных. Новая версия для Оберона. [Электронный ресурс] : учеб. пособие — Электрон. дан. — М. : ДМК Пресс, 2010. — 272 с. — Режим доступа: http://e.lanbook.com/book/1261

3. Петрушин, В.Н. Информационная чувствительность компьютерных алгоритмов [Электронный ресурс]: учебное пособие / В.Н. Петрушин, М.В. Ульянов. - М. : Физматлит, 2010. - 224 с. Режим доступа: http://biblioclub.ru/index.php?page=book&id=75708

4. Быкова, В.В. Теоретические основы анализа параметризированных алгоритмов [Электронный ресурс] / В.В. Быкова. - Красноярск : Сибирский федеральный университет, 2011. - 181 с. Режим доступа: http://biblioclub.ru/index.php?page=book&id=229255

5. Костюкова, Н.И. Комбинаторные алгоритмы для программистов [Электронный ресурс]/ Н.И. Костюкова. - 2-е изд/, исправ/. - М. : Национальный Открытый Университет «ИНТУИТ», 2016. - 217 с. Режим доступа: http://biblioclub.ru/index.php?page=book&id=429067

8. Савельев, В.А. Распараллеливание программ: учебник [Электронный ресурс]/ В.А. Савельев, Б.Я. Штейнберг - Ростов-на-Дону:Издательство ЮФУ, 2008. - 192 с. Режим доступа: http://znanium.com/catalog.php?bookinfo=556119