

## Лаба 1. Списки 1

1. В текстовом файле заданы следующие данные (построчно): слово и некоторое ключевое число, соответствующее этому слову:

<b>Слово1</b>	<b>20</b>
<b>Слово2</b>	<b>86</b>
...	

Считать файл в список таким образом, чтобы он оставался отсортированным по ключевым числам. Вывести список слов и соответствующих им ключевых чисел на экран. Если два слова в списке имеют одно и тоже значение ключевого числа, выдать сообщение об ошибке. Ввести с клавиатуры ключевое число и проверить, есть ли в списке слово, соответствующее такому ключевому числу.

2. В текстовом файле заданы следующие данные (построчно): слово и некоторое ключевое число, соответствующее этому слову:

<b>Слово1</b>	<b>20</b>
<b>Слово2</b>	<b>86</b>
...	

Считать файл в список таким образом, чтобы он оставался отсортированным по словам в алфавитном порядке. Вывести список слов и соответствующих им ключевых чисел на экран. Если одно и то же слово встречается в списке несколько раз, записать его один раз, просуммировав все значения ключевых чисел. Ввести с клавиатуры слово и проверить, есть ли оно в списке.

Вывести на экране слово и соответствующее ему значение ключевого числа.

Ввести с клавиатуры число и выдать на экран список слов, значение ключевого числа у которых совпадает с введенным. Список слов должен быть выведен на экран в алфавитном порядке.

3. Файл содержит следующую информацию (построчно): фамилия, имя, отчество, дата рождения. Считать файл в список таким образом, чтобы он оставался отсортирован по дате рождения. Вывести список фамилий и дат рождения на экран (отсортированным по датам рождения). Ввести с клавиатуры дату рождения и проверить, есть ли в списке человек (один или несколько), родившийся в этот день.

4. Файл содержит следующую информацию (построчно): фамилия, имя, отчество. Считать файл в список таким образом, чтобы он оставался отсортирован по фамилиям (при совпадении фамилий - по именам; при совпадении и фамилий и имен - по отчествам). Вывести список фамилий на экран. Ввести с клавиатуры начальные буквы фамилии, имени и отчества и проверить, есть ли люди с такой фамилией в списке.

Пример:

1. Иванов Иван Иванович
2. Петров Иван Петрович

Запрос «И» «И» «>» выдает первую фамилию в списке

Запрос «>» «Ив» «>» выдает обе фамилии в списке

5. Считать все слова из текстового файла в список таким образом, чтобы список слов оставался отсортированным по длине слов (при совпадении длины следует сортировать список по алфавиту). Вывести на экран список всех встречающихся в файле слов, длина которых превышает N символов. Ввести с клавиатуры число и проверить, есть ли в списке слова такой длины. Вывести эти слова на экран в алфавитном порядке.

6. Файл содержит следующую информацию (построчно): дата и средняя температура в этот день. Считать файл в список таким образом, чтобы он оставался отсортированным по возрастанию средней температуры (при совпадении средней температуры проводить сортировку следует по дате). Вывести на экран информацию обо всех днях, в которых средняя температура оказалась ниже нуля. Ввести с клавиатуры число и проверить, есть ли в списке день с такой средней температурой.

7. Файл содержит следующую информацию (построчно): дата, фамилия, количество рабочих часов. Считать информацию из файла в список таким образом, чтобы он оставался отсортирован по фамилиям сотрудников. Подсчитать и вывести на экран общее рабочее время каждого отдельного сотрудника. Ввести с клавиатуры начальные буквы фамилии и проверить, есть ли сотрудники с такой фамилией в списке. Вывести список сотрудников и суммарное количество рабочих часов.

Пример:        1.09.02 Иванов 8  
                  1.09.02 Петров 9  
                  2.09.02 Иванов 9  
                  3.09.02 Петров 7

8. Файл содержит следующую информацию (построчно): дата, фамилия и имя, количество рабочих часов. Считать информацию из файла в список таким образом, чтобы он оставался отсортирован по количеству рабочих часов (при совпадении количества рабочих часов сортировать список по алфавиту). Вывести на экран информацию о тех сотрудниках (фамилия и дата), которые суммарно отработали более N часов.

9. Считать из текстового файла все строки и заполнить список, содержащий следующую информацию: текст строки и его длина. Список должен заполняться таким образом, чтобы длины строк располагались в порядке возрастания. Вывести на экран 10 самых длинных строк. Ввести с клавиатуры число и проверить, есть ли в списке строка (или строки), имеющие такую длину.

Вывести их на экран.

10. Файл содержит следующую информацию (построчно): название товара, артикул (код товара) и его стоимость. Считать файл в список таким образом, чтобы он оставался отсортирован по цене. При совпадении цены вести сортировку по артикулу. Вывести на экран 10 товаров с самой высокой ценой. Ввести с клавиатуры число и найти товары с указанной стоимостью.

Название товара и его артикул являются уникальными идентификаторами. Если в файле встретились два товара, имеющих один артикул, но разные названия, вывести сообщение об ошибке. Однако одно и то же название может иметь разные артикулы (т.е. разные производители).

11. Файл содержит следующую информацию (построчно): название товара, артикул (код) и его стоимость. Читать файл в список таким образом, чтобы он оставался отсортирован по названию товара. Вывести на экран первые 10 товаров (в алфавитном порядке). Ввести с клавиатуры слово и проверить, есть ли такой товар в списке.

Название товара и его артикул являются уникальными идентификаторами. Если в файле встретились два товара, имеющих один артикул, но разные названия, вывести сообщение об ошибке. Однако одно и то же название может иметь разные артикулы (т.е. разные производители).

12. Файл содержит следующую информацию (построчно): название товара, артикул (код) и его стоимость. Читать файл в список таким образом, чтобы он оставался отсортирован по коду. Вывести на экран первые 10 товаров (в алфавитном порядке по возрастанию артикула). Ввести с клавиатуры код и проверить, есть ли товар с таким кодом в списке.

Название товара и его артикул являются уникальными идентификаторами. Если в файле встретились два товара, имеющих один артикул, но разные названия, вывести сообщение об ошибке. Однако одно и то же название может иметь разные артикулы (т.е. разные производители).

13. Файл содержит список адресов (улица, номер дома) и количество жилых квартир в этом доме. Читать файл в список таким образом, чтобы он оставался отсортирован по адресам (при совпадении названия улицы сортировать по номеру дома). Вывести на экран список адресов. Ввести с клавиатуры адрес и проверить, есть ли такой дом в списке.

Если в файле встречается две строки, у которых один адрес и номер дома, но разные количество жилых квартир, выдать сообщение об ошибке.

14. Файл содержит список адресов (улица, номер дома) и количество жилых квартир в этом доме (если в доме находится офис, количество жилых квартир равно 0). Читать файл в список таким образом, чтобы он оставался отсортирован по количеству жилых квартир. Вывести на экран адреса домов, в которых находятся офисы. Ввести с клавиатуры количество квартир и найти все дома, удовлетворяющие данному требованию.

Если в файле встречается две строки, у которых один адрес и номер дома, но разные количество жилых квартир, выдать сообщение об ошибке.

15. Заданы два файла слов. Читать их в упорядоченные списки (использовать сортировку вставками). Построить единый упорядоченный список.

## **Лаба 2. Списки 2**

1. Написать процедуры выделения и освобождения памяти блоками одинакового размера. Вместо указателей использовать двухбайтовые индексы В блок добавить дескриптор – флаг занятости. Использовать его для контроля повторного освобождения памяти.
2. Написать процедуры выделения и освобождения памяти блоками одинакового размера. Если исходный массив заполнен – алокировать дополнительный. В блок

добавить дескриптор – флаг занятости. Использовать его для проверки целостности списка.

3. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Использовать односвязный кольцевой список и стратегию “first fit”. Выполнять контроль повторного освобождения памяти.
4. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Использовать двухсвязный кольцевой список и стратегию “first fit”. Написать процедуру проверки целостности списка.
5. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Использовать двухсвязный список и стратегию “first fit”. Поиск подходящего блока выполнять поочередно в разных направлениях. Выполнять контроль повторного освобождения памяти.
6. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Использовать односвязный список и стратегию “best fit”. Написать процедуру проверки целостности списка.
7. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Использовать двухсвязный список и стратегию “best fit”. Выполнять контроль повторного освобождения памяти.
8. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Вместо указателей использовать двухбайтовые индексы. Размер блока в дескрипторе хранить в единственном экземпляре, но в функции освобождения памяти передавать кроме указателя размер блока. Использовать односвязный кольцевой список и стратегию “first fit”. Выполнять контроль повторного освобождения памяти.
9. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Вместо указателей использовать двухбайтовые индексы. Размер блока в дескрипторе хранить в единственном экземпляре, но в функции освобождения памяти передавать кроме указателя размер блока. Использовать двухсвязный кольцевой список и стратегию “first fit”. Написать процедуру проверки целостности списка.
10. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Вместо указателей использовать двухбайтовые индексы. Размер блока в дескрипторе хранить в единственном экземпляре, но в функции освобождения памяти передавать кроме указателя размер блока. Использовать двухсвязный список и стратегию “first fit”. Поиск подходящего блока выполнять поочередно в разных направлениях. Выполнять контроль повторного освобождения памяти.
11. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Вместо указателей использовать двухбайтовые индексы. Размер блока в дескрипторе хранить в единственном экземпляре, но в функции освобождения памяти передавать кроме указателя размер блока. Использовать односвязный список и стратегию “best fit”. Написать процедуру проверки целостности списка.

12. Написать процедуры выделения и освобождения памяти блоками произвольного размера. Вместо указателей использовать двухбайтовые индексы. Размер блока в дескрипторе хранить в единственном экземпляре, но в функции освобождения памяти передавать кроме указателя размер блока. Использовать двухсвязный список и стратегию “best fit”. Выполнять контроль повторного освобождения памяти.

## **Лаба 3. Списки 3**

1. Записная книжка

Написать программу, осуществляющую работу с базой данных «Записная книжка». Элемент данных - фамилия и телефон. Ключ для поиска – фамилия. Базу данных хранить в памяти в виде массива самоорганизующихся списков проиндексированного буквами алфавита. Добавление выполнять в начало соответствующего списка. Написать процедуры поиска, удаления и сортировки заданного списка. Базу данных зачитывать и сохранять в файл.

2. Многоугольник

Написать процедуру растеризации невыпуклого многоугольника.

3. База данных для деканата

Написать программу, осуществляющую работу с базой данных «Деканат» (Группа, фамилия, имя, отчество, оценки за экзамены). Базу данных хранить в памяти в виде массива списков групп. Осуществить добавление, удаление и сортировку элементов списка, формирование списка на отчисление и на начисление стипендии.

4. Библиотека работы с полиномами

Написать библиотеку работы с полиномами произвольной степени, позволяющую осуществлять их сложение, вычитание, умножение. Исходные данные зачитывать из файла. Предполагается, что в файле степени заданы в порядке убывания. Результат сохранять в файл. Полиномы представлять в виде списка, упорядоченного по убыванию степеней.

5. Сложение разреженных матриц

Исходные матрицы зачитывать из файла. Результат выводить на экран в виде обычной матрицы с нулями. Разреженные матрицы хранить в памяти в виде массива списков.

6. Умножение разреженных матриц

Исходные матрицы зачитывать из файла. Результат выводить на экран в виде обычной матрицы с нулями. Разреженные матрицы хранить в памяти в виде массива списков.

#### 7. Объединение разреженных матриц

Считать из файлов и построить 4 разреженные матрицы размером  $n \times n$ . Построить из них одну разреженную матрицу размером  $2n \times 2n$ . Разреженные матрицы хранить в памяти в виде массива списков.

#### 8. Разбиение разреженных матриц на кленки

Считать из файлоа и построить разреженную матрицу размером  $2n \times 2n$ . Построить из них нее 4 разреженных матрицы размером  $n \times n$ . Разреженные матрицы хранить в памяти в виде массива списков.

#### 9. Транспонирование разреженных матриц

Исходные матрицы зачитывать из файла. Результат выводить на экран в виде обычной матрицы с нулями. Разреженные матрицы хранить в памяти в виде массива списков.

#### 10. Построение графа 1

Граф задан в файле матрицей смежностей. Построить список смежностей..

#### 11. Построение графа 2

Граф задан в файле списком смежностей. Построить матрицу смежностей..

#### 12. Построение графа 3

В одном файле граф задан списком смежностей в другом - матрицей смежностей. Проверить графы на совпадение.

#### 13. Топологическая сортировка

В файле задан список имен объектов и отношения “предшествует” между ними.

#### 14. Топологическая сортировка

В файле заданы только отношения “предшествует” между объектами (в виде  $a < b$ ).

## Лаба 4. Алгоритм с возвратом

### Задание 4.1

#### Арифметика

##### Условие

Заданы два целых положительных числа  $A$  и  $B$ . Расставьте знаки арифметических операций (+, -, \*, /) между цифрами числа  $A$ , чтобы получить  $B$ .

## Исходные данные

В текстовом файле input.txt записаны два числа  $A, B$ .  $A \leq 10^{100}$ ,  $B \leq 2^{32}$ . Выведите в текстовый файл output.txt арифметическое выражение, результатом которого является число  $B$ , или 0 если такого выражения построить нельзя

### Пример

input.txt	output.txt
123456789 1214	12+34*56-78*9

## Задание 4.2

### Лабиринт

Лабиринт задан матрицей: 1 означает стену, 0 - проход. Найти самый короткий выход из лабиринта (проход до любого края лабиринта) из указанной точки.

## Задание 4.3

### Кратчайший путь

На шахматной доске размером  $N \times N$  найти кратчайший путь ходами коня из поля  $A$  в поле  $B$ .

## Задание 4.4

### Аналог задачи коммивояжера

Задано  $N$  городов, связанных дорогами. Для каждой дороги задано время и стоимость проезда. Требуется посетить все города за минимальное время истратив не более фиксированной суммы.

## Задание 4.5

### Гамильтонов путь

#### Условие

Гамильтонов путь (или гамильтонова цепь) — путь (цепь), содержащий каждую вершину графа ровно один раз. Гамильтонов путь, начальная и конечная вершины которого совпадают, называется гамильтоновым циклом.

Зада граф  $G = (V, E)$ . Определите, существует ли гамильтонов путь в этом графе.

#### Исходные данные

В текстовом файле input.txt храниться граф заданный списком смежности. В первой строке указано количество вершин в графе  $|V| < 1000$ , в последующих строках содержаться списки номеров смежных вершин. Вершины нумеруются начиная с 1. Выведите в текстовый файл output.txt номера вершин в том порядке, в котором они встречаются в гамильтоновом пути на заданном графе. Если такого пути нет, выведите 0.

### Пример

В примере рассмотрен граф с тремя вершинами, образующих треугольник

input.txt	output.txt
3 2 3 1 3 1 2	1 2 3

## Задание 4.6

### Самый длинный путь

#### Условие

Задан неориентированный граф  $G = (V, E)$ , две вершины  $s, t \in V$  и целое положительное число  $K$ . Существует ли в данном графе простой путь (путь в котором нет повторяющихся ребер) от  $s$  к  $t$  длинна которого (количество ребер в пути) больше или равна  $K$ ?

#### Исходные данные

В первой строке текстового файла записаны четыре целых положительных числа:  $|V|$ ,  $s$ ,  $t$  и  $K$ .  $|V| < 1000$ ,  $1 \leq s, t \leq |V|$ ,  $K \leq |V|^2$ . В последующих строках содержатся списки номеров смежных вершин. Вершины нумеруются начиная с 1. Найдите путь в графе от вершины  $s$  до вершины  $t$ , длинны не меньше  $K$ . Выведите в текстовый файл output.txt номера вершин в том порядке, в котором они встречаются в найденном пути. Если такого пути нет, выведите 0.

### Пример

В примере рассмотрен граф с тремя вершинами, образующих треугольник

input.txt	output.txt
3 1 3 2 2 3 1 3 1 2	1 2 3

## Задание 4.7

### Самый длинный цикл

#### Условие

Задан неориентированный граф  $G = (V, E)$  и целое положительное число  $K$ . Существует ли в данном графе цикл количество ребер в котором не меньше  $K$ ?



## Исходные данные

В первой строке текстового файла записаны два целых положительных числа:  $V$  и  $K$ .  
 $V < 1000$ ,  $K \leq V^2$ . В последующих строках содержатся списки номеров смежных вершин. Вершины нумеруются начиная с 1. Найдите в графе цикл длинны не менее  $K$  и выведите в текстовый файл output.txt вершины, которые образуют этот цикл. Вершины вывести в том порядке, в котором они встречаются в цикле. Если такого цикла не существует, выведите 0.

### Пример

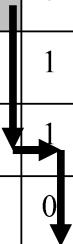
input.txt	output.txt
4 3 2 3 4 1 3 1 2 2 3	1 2 3

## Задание 4.8

### Максимальный бонус

Поле задано двумерным массивом, в заданной точке стоит человек. Каждая ячейка массива содержит 0 - на эту ячейку можно ступить левой ногой или 1 - на эту ячейку можно ступить правой ногой. Человек может ступить с одной клетки на другую, меняя ногу.

1	1	0	1	1
1	0	0	0	1
0	0	1	1	1
0	1	0	1	1
0	1	0	0	1



Дополнительно в  $K$  ячейках содержат 2 (на рисунке не показано). На эти ячейки можно ступить любой ногой. Более того, за это назначается бонус. Найти путь с максимальным числом таких ячеек.

## Задание 4.9

### Минимизация числа невыполненных заданий

#### Описание

От сортировочной железнодорожной станции каждый день в 18.00 отходит грузовой состав. Маневровый тепловоз должен составить вагоны в состав. Для каждого вагона известно, сколько времени требуется, чтобы подцепить вагон к составу. Некоторые вагоны будут отцепляться по ходу следования состава на промежуточных станциях, поэтому вагоны, которые будут отцепляться раньше, должны оказаться в конце состава (чтобы на промежуточных станциях отцепить только хвост состава). Как организовать работу

тепловоза, чтобы после отправления состава на сортировочной станции осталось, как можно меньше вагонов?

### Условие

Задано множество вагонов  $W$ , время  $t(w) \in \mathbb{Z}^+$ , которое надо затратить, чтобы пристыковать вагон к составу, частичный порядок  $\preceq$  на множестве вагонов ( $w_1 \preceq w_2$  - если вагон  $w_1$  должен быть пристыкован раньше, чем  $w_2$ ), время до отправления состава  $D$ , и целое число  $K \leq |W|$ . Существует ли такая последовательность составления вагонов

$w_1, w_2, \dots, w_n$ , удовлетворяющая условию частичного порядка, что:  $\sum_{i=1}^n t(w_i) \leq D$ , и  $n \geq |W| - K$  (на станции осталось не более  $K$  вагонов)

### Исходные данные

В первой строке текстового файла input.txt записаны три целых числа:  $|W|$ ,  $D$  и  $K$ .  $|W| \leq 1000$ ,  $0 \leq D \leq 2^{32}$ ,  $0 \leq K \leq |W|$ . В следующей строке записаны веса  $t(w_i), i = 1..|W|$ . В третьей строке указано целое  $N$  – количество пар, и дальше  $N$  пар чисел – суть отношение порядка на множестве вагонов. пара “ $i j$ ” – означает  $w_i \preceq w_j$ . Индексация начинается с 1. Выведите в текстовый файл последовательность вагонов, в составе, удовлетворяющую условию задачи, начиная с головного. Если построить состав невозможно выведите 0.

### Пример

input.txt	output.txt
10 40 3	1 3 4 2 5 6 7
5 3 3 3 6 7 7 9 10 10	
4	
3 2	
4 2	
10 7	
7 9	

## Задание 4.10

### Минимизация максимальных затрат

#### Условие

Задано множество заданий  $T$ , частичный порядок  $\leq$  на множестве  $T$  (одни задания должны быть выполнены раньше), затраты  $c(t) \in \mathbb{Z}$  для каждого задания  $t \in T$  (если  $c(t) < 0$  их можно рассматривать как доход) и константа  $K \in \mathbb{Z}$ . Существует ли однопроцессорное расписание  $\sigma$ , которое удовлетворяет условию частичного порядка, и имеет следующее свойство: для каждого задания  $t' \in T$ :  $\sum_{t: \sigma(t) \leq \sigma(t')} c(t) \leq K$  ?

### Исходные данные

В первой строке текстового файла input.txt записаны два целых числа  $T$  и  $K$ ,  $T \leq 1000$ ,  $K < 2^{32}$ . В следующей строке следует  $T$  целых чисел. Начиная с третьей строки идут пары целых чисел, по одной паре в каждой строке, за исключением последней, в которой записан 0. Пары чисел представляют собой пары множества частичного порядка  $\leq$  на множестве  $T$ . Если  $(i, j) \in \leq$ , то  $t_i$  предшествует  $t_j$ . Нумерация заданий начинается с 1. Если существует последовательность выполнения заданий, удовлетворяющая условию задачи, выведите ее в текстовый файл output.txt. Нумерация заданий начинается с 1. Если такой последовательности не существует, выведите 0.

### Пример

input.txt	output.txt
5 1 1 -1 1 1 -1 1 2 2 3 3 5 3 4	1 2 3 5 4

## Задание 4.11

### Сумма размеров

#### Условие

Заданы конечное множество  $A$ , размеры  $s(a) \in \mathbb{Z}^+$  всех элементов  $a \in A$  и положительное целое число  $B$ . Существует ли такое подмножество  $A' \subseteq A$ , что сумма размеров его элементов равна  $B$ ?

### Исходные данные

В первой строке текстового файла input.txt записано целое положительное число  $B$ ,  $0 < B < 2^{32}$ . Далее записано целое неотрицательное число  $N$ ,  $0 < N < 1000$  - количество элементов множества  $A$ , и  $N$  натуральных чисел  $a_i \leq 2^{16}$ . Выведите в текстовый файл output.txt те элементы множества  $A$ , сумма которых равна  $B$ . Если таких элементов нет, выведите 0. Если задача имеет не единственное решение, выведите любое из них.

### Пример

input.txt	output.txt
10 5 2 3 4 5 9	2 3 5

## Задание 4.12

## Упаковка в контейнеры

### Условие

Задано конечное множество предметов  $U$ , размер каждого предмета  $s(u) \in \mathbb{Z}^+, u \in U$ , положительное целое число  $B$  - вместимость контейнера и положительное целое число  $K$ . Существует ли такое разбиение множества  $U$  на непересекающиеся множества  $U_1, U_2, \dots, U_K$ , что сумма размеров предметов из каждого множества не превосходит  $B$ ?

### Исходные данные

В первой строке текстового файла input.txt записаны три целых положительных числа  $|U|, B, K$ .  $|U| < 1000$ ,  $B < 2^{32}$ ,  $K < 1000$ . В следующей строке записано  $|U|$  целых положительных чисел – веса предметов. Если искомое разбиение существует, выведите в текстовый файл output.txt «Yes», в противном случае выведите «No»

### Пример

input.txt	output.txt
10 11 5 1 2 3 4 5 6 7 8 9 10	Yes

## Задание 4.13

### Расстановка ферзей

#### Условие

На шахматной доске размером  $N \times N$  расставить  $N$  ферзей, так, чтобы не какие два не били друг друга. Для сокращения перебора использовать метод "изменение порядка перебора".

#### Исходные данные

В текстовом файле input.txt записано число целое неотрицательное число  $N < 1000000$ . Выведите в текстовый файл output.txt координаты ферзей – номер ряда и номер колонки, начиная нумерацию с 0

#### Пример

input.txt	output.txt
4	0 2 1 0 2 3 3 1

## Задание 4.14

### Всех бить

На шахматной доске размером  $N \times N$  установлено  $K$  фигур противника и свой конь. Найти минимальную последовательность ходов коня бьющую все фигуры противника.  
Замечание. Конь может возвращаться в уже пройденные поля.

## Лаба 5. Деревья

1. Записать в узлах бинарного дерева разности высот в поддеревьях. Напечатать полученное дерево. При печати использовать вспомогательную матрицу.
2. Записать в узлах бинарного дерева число листьев в поддеревьях. Напечатать полученное дерево. При печати использовать вспомогательную матрицу.
3. Записать в узлах бинарного дерева  $\min$  высоту листьев в поддеревьях. Напечатать полученное дерево. При печати использовать вспомогательную матрицу.
4. Задано бинарное дерево, содержащее слова разной длины. Назовем шириной узла число символов, требуемое для печати данных этого узла. Записать в узлах дерева ширину соответствующих поддеревьев. При печати вычисленные данные печатать под словами. При печати использовать вспомогательную матрицу.
5. Задано бинарное дерево, содержащее целые. Назовем шириной узла число символов, требуемое для печати данных этого узла. Записать в узлах дерева ширину соответствующих поддеревьев. При печати вычисленные данные печатать под исходными. При печати использовать вспомогательную матрицу.
6. Бинарное дерево задано в файле в соответствии со следующим определением: Дерево::=слово (дерево) (дерево) Если "слово"=' ', то это пустое дерево. Напечатать заданное дерево. При печати использовать вспомогательную матрицу.
7. Напечатать дерево в обычном виде и в виде: узел (левое поддерево) (правое поддерево)
8. Написать процедуру удаления поддерева с корнем, содержащим заданный ключ.
9. Дерево задано матрицей (как для печати). Построить дерево.
10. Написать самообучающуюся экспертную систему.
11. Написать программу интерактивного создания бинарного дерева (подобно самообучающейся экспертной системе). Обеспечить сохранение дерева в файл, его зачитывание и печать.
12. Построить сильноветвящееся дерево. Обеспечить возможность его просмотра как дерева каталогов в Norton-e. Найти узел, имеющий наибольшее число сыновей.
13. Построить сильноветвящееся дерево. Обеспечить возможность его просмотра как дерева каталогов в Norton-e. Определить ширину и высоту дерева
14. Построить сильноветвящееся дерево. Обеспечить возможность его просмотра как дерева каталогов в Norton-e. Написать процедуру удаления поддерева с корнем, содержащим заданный ключ.
15. Написать редактор бинарных деревьев.
16. Написать редактор сильноветвящихся деревьев.

## Лаба 6. Деревья (2)

Замечание. Если не оговорено обратное, ссылки на предков отсутствуют и следует использовать рекурсию.

1. Построить таблицу перекрестных ссылок с использованием дерева двоичного поиска.
2. Построить таблицу перекрестных ссылок с использованием дерева двоичного поиска. Для ускорения поиска в дереве использовать барьер.

3. Написать процедуры добавления в бинарное дерево и удаления из него. Добавление выполнять в поддереву с меньшим числом узлов. В узлах поддеревьев хранить число узлов.
4. Написать процедуры добавления в бинарное дерево и удаления из него. Добавление выполнять в поддереву с меньшим числом узлов. В узлах поддеревьев хранить разность числа узлов.
5. Написать процедуры добавления в бинарное дерево и удаления из него. Добавление выполнять в поддереву с меньшим числом узлов. В узлах поддеревьев хранить число узлов. Узлы дерева должны содержать указатели на предков. Рекурсию использовать запрещается.
6. Написать процедуры добавления в бинарное дерево и удаления из него. Добавление выполнять в поддереву с меньшим числом узлов. В узлах поддеревьев хранить разность числа узлов. Узлы дерева должны содержать указатели на предков. Рекурсию использовать запрещается.
7. Используя дерево двоичного поиска с дополнительной информацией реализовать структуру данных с операциями словаря с приоритетами + k-й наименьший. Узлы дерева должны содержать указатели на предков. Рекурсию использовать запрещается.
8. Используя дерево двоичного поиска с дополнительной информацией реализовать структуру данных с операциями словаря с приоритетами + порядковый номер заданного элемента. Узлы дерева не должны содержать указатели на предков. Рекурсия должна быть раскрыта.
9. Для дерева двоичного поиска с дополнительной информацией реализовать процедуру, которая находит + K-й наименьший элемент и, если он оказывается четным, печатает элементы с порядковыми номерами меньшими K.
10. Для дерева двоичного поиска с дополнительной информацией реализовать процедуру, которая по заданному элементу находит элемент, порядок которого на K единиц меньше.
11. Используя дерево двоичного поиска с дополнительной информацией реализовать дерево промежутков (добавить, удалить, найти). Найти все отрезки пересекающиеся с заданным.
12. Используя дерево двоичного поиска с дополнительной информацией реализовать дерево промежутков (добавить, удалить, найти). Найти все отрезки пересекающиеся с заданным. Узлы дерева должны содержать указатели на предков. Рекурсию использовать запрещается.
13. Используя дерево двоичного поиска с дополнительной информацией реализовать дерево промежутков (добавить, удалить, найти). Найти все отрезки пересекающиеся с заданным. Узлы дерева не должны содержать указатели на предков. Рекурсия должна быть раскрыта.
14. BSP-tree
15. Oct-tree