ЛАБОРАТОРНАЯ РАБОТА №1

Фундаментальные структуры данных

1. Цель работы

Знакомство с фундаментальными структурами данных и алгоритмами их обработки, развитие навыков программирования на языке высокого уровня.

2. Домашнее задание

Изучить разделы 1 и 2 конспекта лекций. Знать определение алгоритма, пять его важных особенностей, типы фундаментальных структур данных. Изучить варианты индивидуальных заданий. Знать суть алгоритма Евклида, перестановки, сочетаний, чисел и слов Фибоначчи. Нарисовать блок-схему алгоритма своего варианта задания. Подготовиться к теоретическому коллоквиуму.

3. Варианты индивидуальных заданий

Вариант 1

Реализовать *алгоритм Евклида*: даны два положительных целых числа m и n. Требуется найти их наибольший общий делитель d и два целых числа a и b, таких, что am+bn=d.

Вариант 2

Последовательность

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, …,

в которой каждое число есть сумма двух предыдущих, играет важную роль в десятках несвязных алгоритмах.

Эта знаменитая последовательность была впервые предложена в 1202 г. Леонардо Фибоначчи, поэтому и носит название *чисел Фибоначчи*. В своей “Книге о счете” величайший математик привел такое упражнение: “Сколько пар кроликов получится от одной пары за год?” При этом предполагается понять, что каждая новая пара дает приплод – пару кроликов – каждый месяц, каждая новая пара становится плодоносящей в возрасте одного месяца и, кроме того кролики никогда не мрут.

Сгенерировать максимально возможную последовательность Фибоначчи.

Вариант 3

Образуем последовательность следующим образом. В качестве начального элемента выберем четырехзначное натуральное число, цифры которого не все равны между собой. Каждый следующий элемент последовательности формируется по следующему правилу. Пусть a,b,c,d – цифры исходного числа. Расположим их в порядке убывания слева направо и получим первое число. Затем расположим их в обратном порядке и вычтем это второе число из первого. Это и есть искомый следующий член последовательности. Показать, что эта последовательность становится постоянной (начиная с некоторого элемента), равной 6174. Например:

начнем с 7815;
8751-1578=7173
7731-1377=6354
6543-3456=3087
8730-0378=8352
8532-2358=6174
7641-1467=6174

Вариант 4

Ввести натуральное число. Определить, является ли это число простым.

Вариант 5

Ввести натуральное число N>2. Выбрать все простые числа в интервале от 2 до N.

Вариант 6

Дано натуральное число n ($n \le 9999$). Определить, является ли это число палиндромом (перевертышем) с учетом четырех цифр, как, например, числа 2222, 6116, 0330 и т.д.

Вариант 7

Дано натуральное число n ($n \le 9999$). Определить, содержит ли это число ровно три одинаковые цифры (с учетом четырех цифр), как, например, числа 3222, 6616, 0006 и т.д.

Вариант 8

Дано натуральное число n ($n \le 9999$). Верно ли, что все четыре цифры числа различны?

Вариант 9

Определить количество трехзначных натуральных чисел, сумма цифр которых равна заданному n. $1 \le n \le 27$.

Вариант 10

Вывести на экран все простые делители заданного натурального числа.

Вариант 11

Найти количество «счастливых» трамвайных билетов с номерами от 000000 до 999999 включительно. Билет считается «счастливым», если сумма левых трех цифр номера равна сумме правых трех цифр. (Например, билет с номером 248077 – «счастливый»)

Вариант 12

Дано натуральное число n. Вычислить: $\sqrt{3+\sqrt{6+...+\sqrt{3(n-1)+\sqrt{3n}}}}$

Вариант 13

Известно, что:
$1^3=1$
$2^3=3+5$
$3^3=7+9+11$
$4^3=13+15+17+19$ и т.д.
Определить значение n^3, не используя операции умножения.

Вариант 14

Определить все числа Амстронга в интервале [a,b]. ЧисломАмстронга называется число, равное сумме своих цифр в степени их количества. Например: $153= 1^3+5^3+3^3$

Вариант 15

Напимсать программу перевода натурального десятичного числа в двоичное.

Вариант 16

Определить, является ли введенное число степенью тройки, если – да, то какой.

Вариант 17

Напимсать программу перевода целого двоичного числа в десятичное.

Вариант 18

Задано натуральное число N≤1000. Определить первую цифру этого числа, последнюю и, если N≥10, предпоследнюю.

Вариант 19

Доказать, что любую целочисленную сумму, большую 7 руб. , можно выплатить без сдачи трешками и пятерками. Для данного n>7 найти такие целые неотрицательные a и b, что 3a+5b=n.

Вариант 20

Дано натуральное число N≤99. Выяснить, верно ли, что N^2 равно кубу суммы цифр числа N.

Вариант 21

Дано натуральное число N. Получить все пифагоровы тройки натуральных чисел, каждое из которых не превосходит N, т.е. все такие тройки натуральных чисел a, b,c , что $a^2+b^2=c^2$ ($a \le b \le c \le N$) .

Вариант 22

Треугольником Паскаля называется числовой треугольник вида:

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1

В этом треугольнике по краям стоят единицы, а каждое число внутри равно сумме двух стоящих над ним в ближайшей строке сверху.

Дано натуральное число N. Получить первые N строк треугольника Паскаля.

Вариант 23

Дано натуральное число N. Найти все числа Мерсена, меньшие N. Простое число называется числом Мерсена, если оно может быть представлено в виде $2^p - 1$, где p – тоже простое число.

Вариант 24

Дано натуральное число N. Среди чисел 1, …. , N найти все такие числа, запись которых совпадает с последними цифрами записи их квадрата (как, например, 62=36, 252=625 и т.д.).

Вариант 25

Дано натуральное число N. Выяснить, можно представить N! в виде произведения трех последовательных целых чисел.

Вариант 26

Дано натуральное число N (N>5). Получить все пятерки натуральных чисел x_1,x_2,x_3,x_4,x_5 такие, что $x_1 \ge x_2 \ge x_3 \ge x_4 \ge x_5$ и $x_1+x_2+x_3+x_4+x_5=N$.

Вариант 27

Получить все четырехзначные натуральные числа, в записи которых нет двух одинаковых цифр.

Вариант 28

Даны натуральные числа N и M (M≥N). Определить, сколько из чисел N, N+1, … , M являются номерами високосных годов.

Вариант 29

Даны натуральные числа a, b,c, которые обозначают число, месяц и год. Определить, сколько полных дней осталось до конца года.

Вариант 30

Два числа называются дружественными, если каждое из них равно сумме всех делителей другого, кроме самого этого числа. Найти все пары дружественных чисел, лежащих в диапазоне от 200 до 300.

Вариант 31

Натуральное число называется палиндромом (перевертышем), если его запись читается одинаково с начала и с конца (например, 2222, 6116, 323 и т.д.). Найти все числа-палиндромы, меньше 1000, которые при возведении в квадрат дают также палиндром.

Вариант 32

Даны натуральные числа N и M. Получить сумму M последних цифр числа N.

Вариант 33

Дано натуральное число N. Вычислить

$$1\cdot 2+2\cdot 3\cdot 4+\ldots+N\cdot(N-1)\cdot\ldots\cdot 2N$$

Вариант 34

Дано натуральное число N. Получить наименьшее число вида 2^r, превосходящее N.

Вариант 35

Дано натуральное число N. Получить наименьшее целое k, при котором $4^k<N$.

Вариант 36

Дано натуральное число N. Указать такие неотрицательные целые x, y, z, t, что $N=x^2+y^2+z^2+t^2$.

Вариант 37

Найти натуральное число от 1 до 10 000 с максимальной суммой делителей.

Вариант 38

Даны натуральные числа $m \le n$ и $r \le s$. Вычислить

$$\sum_{j=m}^{n} \sum_{k=r}^{s} j \cdot k$$

Вариант 39

Дано натуральное число n. Вычислить

$1 \cdot 2 + 2 \cdot 2^2 + 3 \cdot 2^2 + \ldots + n \cdot 2^n$

Определить максимальное значение n, для которого можно высислить значение этого выражения, если для вывода результата использовать переменную

a) типа Word; b) типа Byte

Вариант 40

Замечена следующая закономерность:

9 x 1+2=11
9 x 12 + 3 =111
9 x 123 + 4 =1111
9 x 1234 + 5 =11111

Написать программу, подиверждающую эту закономерность.

Вариант 41

Найти наименьшее натуральное число N, представимое двумя различными способами в виде суммы кубов двух натуральных чисел x^3+y^3.

Вариант 42

Найти все n, для которых Fn =n. (Fn - числа Фибоначчи)

Вариант 43

Найти все n, для которых $Fn = n^2$. (Fn - числа Фибоначчи)

Вариант 44

Определить количество совершенных чисел на интервале [a, b]. Совершенным называется число, равное сумме своих делителей за исключением самого себя. Например, 6=1+2+3.

Вариант 45

Образуем последовательность следующим образом. В качестве начального элемента выберем любое натуральное число, кратное трем. Каждый следующий элемент последовательности равен сумме кубов всех цифр предыдущего элемента. Показать, что любая такая последовательность становится постоянной (начиная с некоторого элемента), равной 153. Например:

33
$3^3+3^3=54$
$5^3+4^3=189$
$1^3+8^3+9^3=1242$
$1^3+2^3+4^3+2^3=81$

$8^3+1^3=153$
$1^3+5^3+3^3=153$

4. Результаты работы

В результате выполнения лабораторной работы студент должен описать алгоритм, продемонстрировать преподавателю работу программы и исходные тексты, написанные на любом языке высокого уровня

ЛАБОРАТОРНАЯ РАБОТА №2

Обработка строковых данных

1. Цель работы

Изучение возможностей и овладевание навыками обработки строковых данных.

2. Домашнее задание

Изучить разделы 1 и 2 конспекта лекций. Ознакомиться с описанием и заданием по лабораторной работе. Подготовиться к теоретическому коллоквиуму.

3. Экскурс в язык Паскаль

Тип STRING (строка) в Турбо Паскале широко используется для обработки текстов. Он во многом похож на одномерный массив символов ARRAY [0..N] OF CHAR, однако, в отличие от последнего, количество символов в строке-переменной может меняться от 0 до N, где N – максимальное количество символов в строке. Значение N определяется объявлением типа STRING[N], но не больше 255. По умолчанию значение N=255.

Строка в Турбо Паскале трактуется как цепочка символов. К любому символу в строке можно обратиться точно так же, как к элементу одномерного массива , например:

```
Var
    St : string;
……..
if st[5] = ‘A’ then ……..
```

Самый первый байт в строке имеет индекс 0 и содержит текущую длину строки.

Все действия над строками реализуются с помощью встроенных процедур и функций;

CONCAT(S1 [,S2, …, SN]) – функция типа STRING; возвращает строку, представляющую сцепление строк – параметров S1, S2, …, SN.

COPY(ST, INDEX, COUNT) – функция типа STRING; копирует из строки ST COUNT символов, начиная с символа с номером INDEX.

DELETE (ST, INDEX, COUNT) – процедура; удаляет из строки ST COUNT символов, начиная с символа с номером INDEX.

INSERT(SUBSTR, ST, INDEX) – процедура; вставляет подстроку SUBST в строку ST, начиная с символа с номером INDEX.

LENGTH(ST) – функция типа INTEGER; возвращает длину строки ST.

POS(SUBST, ST) - функция типа INTEGER; отыскивает в строке ST первое вхождение подстроки SUBST и возвращает номер позиции, с которой она начинается; если подстрока не найдена, возвращается ноль.

STR(X [:WIDTH [:DECIMALS]], ST) – процедура; преобразует число X любого вещественного или целого типов в строку символов ST.

4. Варианты индивидуальных заданий

Вариант 1

Ввести строку текста. Определить, содержит ли она символы, отличные от букв и пробела.

Вариант 2

Дана строка; выяснить, является ли она идентификатором переменной или десятичной записью числа.

Вариант 3

Подсчитать, сколько раз входит символ 'a' в текст из пяти строк, сколько раз символ 'b' и сколько раз символ ‘c’. Выдать информацию, какой символ из этих трех встречается наиболее часто.

Вариант 4

Дан текст из десяти строк. Заменить в четных строках все вхождения символа 'a' на 'b', а в нечетных наоборот. Исходные и измененные строки вывести на экран.

Вариант 5

Определить количество вхождений заданной подстроки в строку. Ввод строки и подстроки организовать из файла или с клавиатуры.

Вариант 6

В строке заменить все вхождения подстроки ‘ASU’ на ‘АСУ’. Ввод строки и подстроки организовать из файла или с клавиатуры.

Вариант 7

Подсчитать количество различных символов в вводимой строке.

Вариант 8

Строка содержит несколько слов, между соседними словами не менее одного пробела, за последним словом – точка. Выбрать все слова, имеющие нечетную длину, вывести их на экран в обратном порядке (например, слово «мама» в обратном порядке – «амам»).

Вариант 9

Дана строка, содержащая несколько слов. Выбрать то слово, которое содержит наибольшее количество гласных звуков.

Вариант 10

Дано натуральное число n. Получить символьное представление этого числа в виде последовательности цифр и пробелов, отделяющих группы по три цифры, начиная справа. Например, n=12354376, должно получиться 12 354 376.

Вариант 11

Дано натуральное число n (n≤1000). Записать это число русскими словами. Например, пятнадцать, двести тридцать и т.п.

Вариант 12

Для большинства существительных, оканчивающихся на *–онок* и *–енок*, множественное число образуется от другой основы. Как правило, это происходит по образцу: цыпленок – цыплята, мышонок – мышата и т.д. В новой основе перед последней буквой *т* пишется *а* или *я* в зависимости от предыдущей буквы: если это шипящая, то – *а*, иначе – *я*. Преобразовать подобные существительные единственного числа в существительные множественного числа.

Вариант 13

Ввести строку. Из символов этой строки составить две новые строки: одну – содержащую только цифры; другую – знаки арифметических действий. Если какая либо из строк пустая, вывести сообщение об этом.

Вариант 14

Исходная строка содержит сведения о человеке: фамилию, инициалы, год рождения, рост в см. Эти сведения расположены в произвольном порядке, отделены друг от друга пробелами. Например:

Иванов И.И. 1976 187
И.И. Иванов 187 1976
187 И.И. 1976 Иванов и т.п.

Вывести эти сведения на экран в следующем виде:

Иванов И.И. 1976 года рождения имеет рост 187 см.

Вариант 15

Определить, является ли введенная строка палиндромом, т.е. читается одинаково слава направо и справа налево.

Вариант 16

Определить стоимость телеграммы, если известно, что слово, длиннее К букв, стоит в два раза дороже обычного.

Вариант 17

Ввести строку. Определить наибольшее количество одинаковых символов, идущих в ней подряд.

Вариант 18

Ввести строку, содержащую несколько слов. Определить самое длинное и самое короткое слово.

Вариант 19

Ввести строку, содержащую несколько слов. Определить слово, содержащее наибольшее количество гласных русских букв.

Вариант 20

Строка S содержит фамилию, имя, отчество. Необходимо преобразовать ее в строку, содержащую фамилию и инициалы.

Вариант 21

Ввести строку, содержащую несколько слов. Составить слово из последних букв слов, введенной строки.

Вариант 22

Ввести строку, содержащую несколько слов. Выбрать те слова, в которых первая буква этого слова встречается еще хоть один раз.

Вариант 23

Ввести строку, содержащую несколько слов. В словах, имеющих нечетную длину, удалить среднюю букву.

Вариант 24

Задано целое число от 1 до 999. Вывести его римскими цифрами.

Вариант 25

После каждого слова в строке поставить знак препинания: если слово заканчивается на гласный, то поставить «!», в противном случае поставить «;»

Вариант 26

Ввести строку текста. Определить каких букв – гласных или согласных – больше в этом тексте.

Вариант 27

Ввести два символа. На их основе сгенерировать строку Фибоначи (последеющий элемент строки является конкатенацией двух предыдущих

а б аб баб аббаб бабаббаб аббаббабаббаб…….

Вариант 28

Разработать программу, преобразующую вводимое с клавистуры число в строку символов, от-

деляя группы по 3 разряда пробелами.

Вариант 29

Ввести строку, содержащую несколько слов. Переписать слова в ней в обратном порядке (последнее слово записать первым)

Вариант 30

Заменить в строке все заглавные буквы на строчные и наоборот.

Вариант 31

К строке добавить двоеточие и все первые буквы входящих в нее слов.

Вариант 32

Ввести букву. Удалить ее из всех четных слов заданной строки.

Вариант 33

Ввести две строки. Создать третью следующим образом: четные слова этой строки - слова 1-й строки, нечетные слова – слова 2-й строки, записанные в обратном порядке.

Вариант 34

Подсчитать количество одинаковых слов в предложении.

Вариант 35

Определить, есть ли в строке слова, отличающиеся от заданного не более чем на 1 символ.

Вариант 36

Поставить знаки препинания в предложении: если слово начинается с гласной буквы, перед ним поставить знак “ ; ” , в противном случае – “!” (Пробелы между слов сохранить).

Вариант 37

Задан текст. Вывести все слова, записанные с заглавной буквы.

Вариант 38

Дан текст. Вывести все собственные имена, имеющиеся в нем. (Будем считать, что в начале предложения их нет).

Вариант 39

Задан текст из 5 предложений. Заменить порядок предложений в тексте следующим образом: 1, 3, 5, 2, 4.

Вариант 40

В заданном тексте после четного предложения поставить “ ! ”, после нечетного – “ ? ”

Вариант 41

В заданном тексте все точки заменить не восклицательные знаки, запятые – на вопросительные.

Вариант 42

Перевести число в привычные номер телефона, например:
324567 записать 32-45-67; 2345236 записать 234-52-36

Вариант 43

Сформировать символьную запись цены , например
32,34 – тридцать два рубля 43 коп;
1324,00 – одна тысяча триста двадцать четыре руб, 00 коп.

Вариант 44

Задано математическое выражение. Переписать его в следующем виде:
Сначала математические функции, затем знаки операций, буквы, цифры.

Вариант 45

Ввести число. Найти в тексте все слова, состоящие на более чем из этого числа букв.

Вариант 46

Вывести все сочетания трех заданных букв (сочетания могут быть с повторами или без).

Вариант 47

Определить количество слов в тексте, имеющих заданную длину. Составить строку из первых букв полученных слов.

Вариант 48

Определить количество предложений в тексте.

Вариант 49

Заменить текст следующим образом: последнее слово в предложении сделать первым, предпоследнее – вторым и т.д.

Вариант 50

Составить программу “Циклический сдвиг предложения”. Ввести число. Осуществить циклический сдвиг слов в предложении на указанное число позиций.

5. Результаты работы

В результате выполнения лабораторной работы студент должен разработать и описать алгоритм решения задачи, продемонстрировать преподавателю работу программы и исходные тексты, написанные на любом языке высокого уровня.

ЛАБОРАТОРНАЯ РАБОТА №3

Сортировка массивов и анализ алгоритмов

1. Цель работы

Изучение различных методов внутренней и внешней сортировки, исследование и анализ алгоритмов на примере алгоритмов сортировок массивов.

2. Домашнее задание

Изучить раздел 3 конспекта лекций. Знать методы внутренней и внешней сортировок. Ознакомиться с описанием и заданием по лабораторной работе.

3. Варианты индивидуальных заданий

Для всех вариантов необходимо реализовать два алгоритма сортировки массивов чисел. Проанализировать работу этих алгоритмов в лучшем и худшем случаях. Определить **количество сравнений и перестановок** в каждом из методов**. Сделать выводы** о целесообразности применения методов для различных наборов данных.

Вариант 1

Создать программу, реализующую сортировку массива методами пузырька, и быструю сортировку (метод Хоора)

Вариант 2

Создать программу, реализующую сортировку массива простыми включениями и методом максимумов.

Вариант 3

Создать программу, реализующую сортировку массива простым выбором и сортировку включениями с убывающим приращением (сортировку Шелла).

Вариант 4

Создать программу, реализующую шейкер - сортировку массива и сортировку с помощью дерева.

Вариант 5

Создать программу, реализующую сортировку массива методами предсортировки и слияния, и пирамидальную сортировку.

Вариант 6

Создать программу, реализующую сортировку массива методами цифровой распределяющей сортировки, и сортировку простым выбором.

4. Результаты работы

В результате выполнения лабораторной работы студент должен представить алгоритмы сортировок, продемонстрировать работу программы и исходные тексты, написанные на любом языке высокого уровня, протестировать программы на различных наборах данных (упорядоченные массивы, частично упорядоченные массивы, массивы, упорядоченные в обратном порядке), выводы из анализа результатов работы алгоритмов на различных наборах данных.

ЛАБОРАТОРНАЯ РАБОТА №4

Динамические структуры данных. Организация данных в списковые структуры

1. Цель работы

Изучение и создание списковых структур данных .

2. Домашнее задание

Изучить раздел 4 конспекта лекций. Ознакомиться с заданием по лабораторной работе. Подготовиться к теоретическому коллоквиуму.

3. Экскурс в язык Паскаль

Описание любой переменной простого или структурированного типа служит двум целям:

1) определяется идентификатор как имя этой переменной;
2) выделяется память для хранения этой переменной, которая связывается с ее идентификатором.

Все переменные, обладающие этими свойтсвами, называются статическими. Компилятор может обработать все статические переменные без выполнения программы.

С другой стороны, есть переменные, которые создаются и уничтожаются в процессе выполнения программы. Они не входят в явное описание переменных и, следовательно, к ним нельзя обратиться с помощью идентификаторов. Память для хранения таких переменных нельзя выделить при просмотре статического текста программы. Это можно сделать лишь динамически в процессе выполнения программы. Поэтому такие переменные называются *динамическими.* Доступ к динамическим переменным осуществляется с помощью указателей (ссылок), которые становятся определенными после создания динамического объекта.

Динамическая память – это фактически единственная возможность обработки массивов данных большой размерности. Многие практические задачи трудно или невозможно решить без использования динамической памяти.

3.1. Указатели

Оперативная память ПК представляет собой совокупность элементарных ячеек для хранения информации – байтов, каждый из которых имеет собственный номер. Эти номера называются адресами, они позволяют обращаться к любому байту памяти.

Турбо Паскаль предоставляет в распоряжение программиста гибкое средство управления динамической памятью – так называемые указатели.

Указатель – это переменная, которая в качестве своего значения содержит адрес байта памяти.

Как правило, указатель связывается с некоторым типом данных. Такие указатели будем называть *типизированными*. Для объявления типизированного указателя используется значок ^, который помещается перед соответствующим типом.

```
Var
    P1 : ^integer;
    P2 : ^ real;
Type
    RPointer = ^PRecord;
    PRecord = record
            Name : string;
            Job : string;
            Next : Rpointer;
            End;
```

При объявлении типа RPointer мы сослались на тип PRecord, который предварительно в программе объявлен не был. Это исключение сделано для указателей, которые могут ссылаться на еще не объявленный тип данных.

В Турбо Паскале можно объявлять указатель и не связывать его при этом с каким-либо конкретным типом данных. Для этого служит стандартный тип POINTER.

VAR P : POINTER;

Указатели такого рода будем называть *нетипизированными*.

Значение NIL (пустой указатель) принадлежит всем типам указателей.

Один тип указателей разрешается использовать только для ссылки на однотипные элементы.

Для создания динамических переменных используется стандартная процедура **NEW.** Ее называют процедурой динамического размещения. Если задано описание переменной P

Var P : ^T;

то при обращении к процедуре динамического размещения

NEW(P)

создается новая, полностью неопределенная переменная типа T и новое значение типа - указатель для переменной P, которое указывает на вновь созданную переменную.

Таким образом процедура динамического размещения выполняет два действия:

1) размещает переменную типа T в памяти;
2) присваивает указателю P адрес этой переменной.

Динамическую память можно не только забирать из кучи, но и возвращать обратно. Для этого используется процедура DISPOSE.

```
Const
        P : ^ real = NIL;
……..
if  p = NIL then new(p);
……..
dispose (p);
p := NIL;
……..
```

Следует учесть, что начальное значение указателя (при его объявлении) может быть произвольным, но значение NIL предпочтительнее.

Чередование обращений к процедурам NEW и DISPOSE обычно приводит к “ячеистой” структуре памяти. Другая возможность состоит в освобождении целого фрагмента кучи. С этой целью перед началом выделения динамической памяти текущее значение указателя запоминается в переменной – указателе с помощью процедуры MARK. Теперь можно в любой момент освободить фрагмент кучи, начаная от запомненного адреса до конца динамической памяти, используя процедуру RELEASE.

```
Var
        P, p1, p2, p3, p4, p5 : ^integer;
Begin
        New(p1);
        New(p2);
        Mark(p3);
        New(p4);
        New(p5);
……
        release(p);
```

В этом примере обращение к RELEASE(P) освободило динамическую память от помеченного места до конца кучи.

3.2. Использование указателей

Указатели обычно используют для построения списковых структур данных. Рассмотрим примеры построения некоторых наиболее распространенных видов списков.

Пример 1. *Представление стеков*

Каждый узел списка должен содержать по крайней мере два поля: одно поле типа указатель, а второе – для хранения данных (INFO).

```
Type
    L1 = ^Node;
    Node = record
            Info : char;
            Link : L1;
            End;
```

Предположим, что входной файл содержит некоторое число литер. Тогда формирование стека можно изобразить следующей последовательностью операторов:

```
Var
    Top, K : L1;
    Ch : char;
……. Top := NIL;
While not eof Do
    Begin
            Read (Ch);
            New (K);
            K^. Link :=Top;
            K^. Info :=Ch;
            Top := K;
    End;
```

Включение нового узла можно изобразить с помощью операторов:

```
Var
        Newnode : L1;
……..
        read (Ch);
        new (Newnode);
        Newnode^ . Link := Top;
        Newnode ^ . Info := Ch;
        Top := Newnode;
```

А удаление – с помощью одного оператора

```
Top := Top^ . Link;
```

Пример 2. *Представление очередей.*

Запишем операторы по формированию очереди:

```
Var
        L, R, K : L1;
        Ch : Char;
……..
read (Ch);
New (k);
K^ . Link := NIL;
K^ . INFO := Ch;
R := K; L := K;
While not eof  Do
Begin
        Read (Ch)
```

```
        New (K);
        K^. Link :=L;
        K^. Info :=Ch;
        L := K;
End;
```

Добавление нового узла к очереди происходит справа (используется указатель R):

```
Var
        Newnode :  L1;
……..
read (Ch);
New (Newnode);
R ^ . Link := Newnode;
Newnode^. Link := Nil;
Newnode ^. Info := Ch;
R := Newnode;
```

Исключение узла из очереди происходит слева (используется указатель L) и осуществляется одним оператором

```
L := L^ . Link;
```

Рассмотрим еще один вопрос, связанный с поиском в списке. Предположим, что необходимо найти какую-то определенную литеру, скажем ‘N’. Для описания этих действий воспользуемся следующей схемой:

```
Var Point : L1;
……
Point := L;
A:= TRUE;
While (Point <> NIL) AND A DO
        If  Point ^. Info = ‘N’
        Then A := FALSE
        Else Point := Point ^ . Link;
```

Если литера ‘N’ имеется в списке, то указатель Point будет указывать на тот узел, в поле Info которого записана эта литера. В противном случае значением указателя Point будет NIL.

4. Варианты индивидуальных заданий

В ходе работы для всех вариантов необходимо написать подпрограмму создания в ОП связанного однонаправленного списка, содержащего целые числа в поле данных (числа вводятся с клавиатуры или из файла по выбору). Также в программах всех вариантов должна быть подпрограмма распечатки списка по адресу его первого элемента. Используя эту подпрограмму необходимо вывести список в наглядной форме после его создания и модификации. Основное содержимое программы должно соответствовать варианту.

Примечание: В конце программы необходимо освободить всю захваченную память.

Вариант 1

Написать программу нахождения порядкового номера элемента в списке по его значению.

Вариант 2

Написать программу нахождения максимального по значению элемента списка.

Вариант 3

Написать программу уменьшения всех значений элементов списка на их порядковый номер.

Вариант 4

Написать программу нахождения минимального по значению элемента списка.

Вариант 5

Написать программу определения количества элементов списка с заданным значением. Увеличить все указанные значения элементов списка на 5.

Вариант 6

Удалить из списка первый нулевой элемент, если он есть.

Вариант 7

Написать программу удаления элемента с определенным порядковым номером

Вариант 8

Написать программу вставки нового элемента в список за некоторым заданным порядковым номером элементом (вставка осуществляется не в конец списка).

Вариант 9

Написать программу удаления из списка элементов с четными номерами.

Вариант 10

Написать программу удаления из списка элементов, значения которых больше некоторого заданного.

Вариант 11

Написать программу переноса первого элемента списка в его конец.

.

Вариант 12

Написать программу циклического сдвига элементов списка на одну позицию вправо.

.

Вариант 13

Написать программу переноса в начало списка его последнего элемента.

Вариант 14

Написать программу переворачивания списка, т.е. сменить ссылки так, чтобы его элементы оказались расположенными в обратном порядке.

Вариант 15

Написать программу удаления из списка всех положительных элементов.

Вариант 16

Найти среднее арифметическое элементов списка

Вариант 17

Написать программу циклического сдвига элементов списка влево на одну позицию.

Вариант 18

Написать программу вставки и удаления элемента в начало списка.

Вариант 19

Написать программу вставки и удаления последнего элемента списка.

Вариант 20

Написать программу, формирующую два списка, запоняя их числами из файлов. Объединить оба списка в один, вставляя элементы поочередно то из одного, то из другого списка. (Первый эл-т списка №1; первый эл-т списка №2; второй эл-т списка №1; второй эл-т списка №2 и т.д.).

Вариант 21

Написать программу, формирующую два списка, запоняя их числами из файлов. Получить новый список, значения каждого из элементов которого, равны сумме значений соответствующйх элементов исходных списков.

5. Результаты работы

В результате выполнения лабораторной работы студент должен продемонстрировать преподавателю работу программы и исходные тексты, написанные на любом языке высокого уровня.

ЛАБОРАТОРНАЯ РАБОТА №5

Рекурсивные структуры данных. Графы

1. Цель работы

Изучение возможностей и овладение навыками работы с древовидными структурами данных.

2. Домашнее задание

Изучить раздел 4 конспекта лекций. Ознакомиться с описанием и заданием по лабораторной работе.

Нарисовать на бумаге граф, содержащий не менее 8 вершин. Задать его матрицами смежности и инциденций.

Дерево - одна из самых распространнных структур., используемых для представления данных в ЭВМ. А вообще, дерево-конечное множество, состоящее из одного или более элементов, называемых узлами.

Между узлами существует отношение типа "исходный-порождённый". Корень - узел, не имеющий исходного (отца).Все узлы, кроме корня, имеют только один исходный (исх.отца). Есть деревья, состоящие из одного корня. Каждый узел может иметь несколько порождённых (сыновей). Отношение "исходный-порождённый" действует только так: не бывает отношения "порождённый-исходный", т.к потомок узла никогда не станет его предком.

Древовоидная структура очень часто представляется в списковой форме.

Граф обычно определяется как некоторое множество точек (называемых *вершинами*) и некоторое множество линий, называемых *ребрами*, соединяющих определенные пары вершин. Каждая пара вершин соединяется не больше чем одним ребром. Дуга, соединенная с вершиной, называется *инцидентной этой вершине*. Две вершины называются *смежными* если существует ребро, соединяющее их. Две дуги называются *смежными*, если они инцидентны одной и той же вершине.

Для задания графов существует несколько классов матриц, основные из которых класс матриц инциденции и класс матриц смежности **.**

3. Варианты индивидуальных заданий

Вариант 1

Для заданного графа вывести все пары связных вершин, сосчитать их количество.

Вариант 2

Для заданного графа найти и вывести путь между заданными вершинами.

Вариант 3

Сосчитать количество ребер заданного графа.

Вариант 4

Сосчитать количество ребер заданного *ориентированного* графа.

Вариант 5

Создать программу поиска в графе вершины, из которой исходит наибольшее число ребер.

Вариант 6

Для заданного графа найти и вывести кратчайший путь между заданными вершинами.

4. Результаты работы

В результате выполнения лабораторной работы студент должен продемонстрировать преподавателю работу программы и исходные тексты, написанные на любом языке высокого уровня. Протестировать программу на нескольких вариантах исходных данных. Дополнить отчет графическим представлением структуры.

СПИСОК ЛИТЕРАТУРЫ

1. Вирт Н. Алгоритмы и структуры данных. М.: Мир,1989.
2. Сибуя М., Ямамото Т. Алгоритмы обработки данных. М.: Мир,1986.
3. Костин А.Е., Шаньгин В.Ф. Организация и обработка структур данных в вычислительных системах: Учеб. пособ. для вузов. М.: Высш. шк., 1987. 248 с.: ил.
4. Кнут Д. Искусство программирования для ЭВМ. Т.1: Основные алгоритмы.: Пер. с англ. М.: Мир, 1978.
5. Кнут Д. Искусство программирования для ЭВМ. Т. 3: Сортировка и поиск.: Пер. с англ.– М.: Мир, 1978.
6. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. М.: МЦНМО, 2000. 960 с.: ил.
7. Фаронов В. В. Турбо Паскаль. (В 3 кн.). Кн.1. Основы Турбо Паскаля. – М.: Учебно-инженерный центр “МВТУ – ФЕСТО ДИДАКТИК”, 1992. 304 с., ил.