МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО
СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ
УЗБЕКИСТАН

ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ БЕРУНИ

**Методические указания к выполнению
лабораторных работ по**

ИНФОРМАТИКЕ, ИНФОРМАЦИОННЫМ ТЕХНОЛОГИЯМ
часть 1

Ташкент – 2003

Составители: Якубов А.Х., Сагатов М.В., Ирмухамедова Р.М.,
Равилов Ш.М., Каримова Д.К., Бабаханова В.Ю.

Методические указания к выполнению лабораторных работ по информатике, информационным технологиям (часть 1)/ Сост.: Якубов А.Х., Сагатов М.В. и другие. Таш. гос. техн. ун-т. Ташкент, 2003, С. 60

Данные методические указания предназначены для выполнения лабораторных работ по курсу «Информатика, информационные технологии». В них, в частности, рассматриваются предназначение, основные устройства, принципы работы и технические возможности современных IBM персональных компьютеров, операционной системы MS-DOS, работа с программой - оболочкой Norton Commander, составление программ алгоритмов линейной, разветвляющейся и циклической структуры на Турбо Паскале, использование процедур и функций, а также составление программ с привлечением сложных типов ТР, таких как регулярного, файлового, записей и множеств.

Описание каждой лабораторной работы содержит сущность и краткую теоретическую часть работы. Составление программ проиллюстрировано на конкретных примерах.

Методические указания предназначены для студентов очной и заочной форм обучения. Кроме этого, они также могут быть полезны для магистров, аспирантов, преподавателей и самостоятельно изучающих алгоритмический язык Паскаль.

Кафедра «Информатика»

Печатается по решению научно-методического совета Ташкентского государственного технического университета имени Беруни.

Рецензенты: 1. Солиев Х.А. – к.т.н., заведующий кафедрой «Информатика» Ташкентского химико – технологического института.
2. Гаипназаров Т. – к.т.н., доцент кафедры «Программное обеспечение ВТ и АС», ТашГТУ.

Лабораторная работа N 1.
Знакомство с персональной электронно-вычислительной машиной.

Цель: изучить основные части ПЭВМ и технические возможности компьютера.

Задания:

1. Ознакомление со структурной схемой ПЭВМ.
2. Назначение блоков ПЭВМ.
3. Технические характеристики ПЭВМ.
4. Устройства ввода/вывода ПЭВМ.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ.

1.Структура ПЭВМ.

Общая структурная схема ПЭВМ представлена на рис.1. Компьютер состоит из следующих блоков: микропроцессор, интерфейсы для устройств ввода /вывода (ИВВ/ИВЫВ), основная память, системная магистраль (СМ).

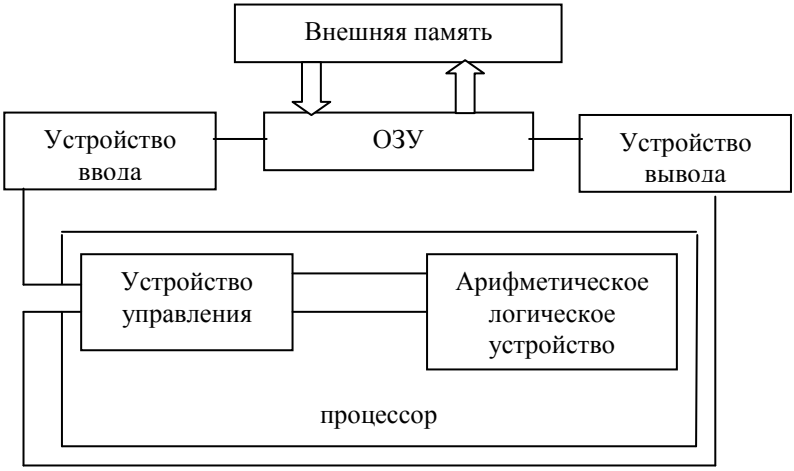

Рис.1. Блок-схема ЭВМ.

2.Блоки ПЭВМ.

МИКРОПРОЦЕССОР. Он является главным компонентом микрокомпьютера. Характеристики микропроцессора (МП) - длина разрядной сетки, набор выполняемых команд, быстродействие и другие в основном определяют

характеристики всего компьютера. МП выполняет следующие функции:

управление и координация работ всех других компонентов компьютера; выборка команд и данных из памяти; выполнение с помощью арифметико-логического устройства (АЛУ) арифметических, логических и других операций; обработка сигналов от устройств ввода /вывода.

В состав МП входят устройство управления, АЛУ и набор регистров.

Устройство управления (УУ) предназначено для управления работой всех компонентов компьютера и обеспечения должного взаимодействия различных компонентов друг с другом.

Арифметико-логическое устройство предназначено для исполнения арифметических и логических операций.

Регистры - электронные устройства для временного запоминания информации в форме двоичного числа. Запоминающим элементом в регистре является триггер, который может находится в одном из двух устойчивых состояний.

Основная память - важный компонент компьютера. Название "основная" отличает эту память от внешней, которая организуется на некотором внешнем носителе. Основная память состоит обычно из двух частей - ОЗУ и ПЗУ.

Оперативное запоминающее устройство обеспечивает чтение, хранение и запоминание данных. Хранение информации не долговечно, т.е. после выключения питания данные теряются.

Постоянное запоминающее устройство обеспечивает только чтение информации. Содержимое ПЗУ не может быть изменено микропроцессором. При выключении питания содержимое ПЗУ не стирается.

Интерфейсы для устройств ввода (ИВВ) и вывода (ИВЫВ). Они представляют собой блоки сопряжения с устройствами ввода/вывода. Термин " интерфейс" означает приблизительно " сопряжение двух устройств друг с другом". Необходимость интерфейсов, или, как их еще называют, контроллеров, вызывается тем, что нельзя непосредственно подключать к компьютеру.

Внешние устройства предназначены для ввода обрабатываемых данных, результатов обработки или

вычислений и долговременного хранения программ и данных. Внешние устройства могут быть такими как дисплей, клавиатура, принтер, дисковод, винчестер и др.

Системная магистраль (СК). Обычно она содержит магистрали адресов, данных, управления. Каждая из них состоит из набора проводников, по которым МП передает или принимает определенные электрические сигналы. Магистраль адреса предназначена для передачи цифрового адреса ячейки памяти или внешнего устройства, магистраль данных - для передачи и приема данных. Магистраль управления используется для передачи сигналов управления, которые сопровождают любую передачу адреса или данных.

3. Технические характеристики IBM компьютера.

Введение. Первый универсальный микропроцессор 4004 фирмы Intel появился в 1971 г. Длина слова этого микропроцессора составляла всего 4 бита. Через несколько лет появился другой микропроцессор 8008 и 8080 (достаточно мощный для построения небольшого компьютера). 18080 может выполнять десятичные и 16-ричные операции, адресовать память до 64Кбайт.

В 1978г. начат выпуск микропроцессоров 8086 от фирмы Intel, а через год 8088 той же фирмы. Размер слова составляет 16бит. Размер адресной шины 20бит, что позволяет адресовать память до 1Мбайт. Затем появляются 80186,80386,80486,80586.

Характеристики IBM машины (xt/at):

Наличие микропроцессора	80286
Сопроцессор	80287
Тактовая частота	8/10 MHz
ПЗУ	8(64)Кбайтов
ОЗУ	256/512/640/10024Кбайтов
Часы-календарь	Есть

Внешние устройства:

Дисплей: текстовый режим	40x25(80x25) (16 цветов)
Графический режим	320x200 (16 цветов) 640x200 (черно-белый)
Дисковод гибкого диска	0.360/1.2Мбайт

Винчестер	40Мбайт
Клавиатура	101/102 клавиши (возможно 84)

ПРАКТИЧЕСКАЯ ЧАСТЬ.

4. Устройства ввода/вывода ПЭВМ.

Клавиатура разделяется на три основные части: 1) клавиатура "пишущей машинки"; 2) клавиатура "калькулятора"; 3) функциональные клавиши.

В первой части клавиатуры клавиши расположены так же, как и на обычной пишущей машинке. Некоторые клавиши присущи только клавиатуре компьютера (используемый стандарт - QWERTY).

Клавиши управления программой:

ESC - выполняет ряд функций, документированных в описании ОС или прикладной программы. Чаще всего - выход из меню.

TAB - табуляция, продвигает курсор на 8 позиций вправо. Ctrl - в сочетании с другими клавишами используется для выдачи команд или для функций. Самое распространенное применение - Ctrl+Break прерывание выполнения команд, программ.

-"Shift" служит для перехода от прописных букв к строчным и обратно. При нажатии на небуквенную клавишу появляется символ, нанесенный на верхнюю половину клавиши.

Alt - совместно с буквенными клавишами служит для выдачи ключевых слов языка, оболочки.

- "Enter" перемещает курсор из последней позиции на текущей строке в первую позицию на следующей строке.

- "BackSpase" перемещает курсор на одну строку влево и одновременно стирает знак в этой позиции.

PrtSc - будучи нажата одновременно с клавишей "Shift", выдает на принтер копию экрана дисплея.

CapsLock - при нажатии переключает с прописных на строчные и обратно.

Цифровая клавиатура:

NumLock - однократное нажатие этой клавиши переводит оцифрованные клавиши в режим управления или наоборот.

Scroll Lock - функции определены в оболочке, либо в O S.

Ins - включает режим вставки (замены), т.е. позволяет вставлять буквы между двумя символами (писать поверх написанной строки - затирать ее).

Del - стирает знак в текущей позиции курсора.

- "Minus" вводит знак "-" в текущую позицию курсора.

+ - то же, но "+".

Home - перемещает курсор в первую позицию первой либо текущей строки.

End - перемещает курсор в последнюю позицию последней либо текущей строки.

PgUp/PgDn - программно-управляемые клавиши. Конкретные функции определяются OS. Обычно они перемещают курсор на страницу вверх/вниз соответственно.

Значение функциональных клавиш полностью определяется прикладными программами.

Гибкие дискеты. На гибких дискетах можно при малых затратах хранить "большие" объемы информации. Дискета 5.25 дюйма с двойной плотностью записи может хранить 360 килобайт информации, т.е. около 180 страниц по 2000 знаков. Для компьютера AT выпускаются дискеты 5.25 дюйма высокой плотности емкостью 1.2 мегабайта.

Применяются также дискеты 3.5 дюйма и 8 дюймов. Все более популярными становятся дискеты 3.5 дюйма, т.к. они могут хранить до 1.44 мегабайт информации; корпус изготовлен из достаточно прочной пластмассы, окно для доступа к магнитному слою автоматически закрывается при изъятии ее из дисковода, а размеры позволяют носить ее в нагрудном кармане.

Дискета выполнена в виде тонкого винилового диска, покрытого магнитным слоем и помещена в пластмассовый корпус или пакет с окнами для доступа магнитной головки к магнитному слою. Обычно дискету хранят в бумажных или пластмассовых конвертах для защиты от пыли и механических воздействий.

Принтер. К персональному компьютеру можно подключить любой принтер, имеющий параллельный интерфейс типа "Centronix" или последовательный интерфейс типа RS - 232-C- например, принтер Logitec FT5002. Это принтер с матричной печатью и параллельным интерфейсом; скорость печати 120 знаков в секунду. Принтер подключают

специальным кабелем к системному адаптеру параллельного принтера.

Плотность печати составляет от 5 до 16.5 знаков на дюйм (40,66,80,132 знака в строке). В правой нижней части передней панели принтера расположена панель управления с тремя кнопками и четырьмя индикаторами.

При подачи питания на принтер засветится индикатор "Power". Если принтер готов к приему данных, светится индикатор "Ready". В процессе приема принтером информации от компьютера этот индикатор мигает. Если бумага кончилась, светится индикатор "No paper out"; одновременно раздастся звуковой сигнал.

Нажатие кнопки "On Line" останавливает процесс приема данных и позволяет воспользоваться кнопками "Form Feed" (продвинуть бумагу до начала следующей страницы) и "Line Feed" (продвинуть на одну строку).

Для возобновления приема данных надо нажать еще раз кнопку "On Line".

Контрольные вопросы.

1.Что такое ЭВМ?

2.Какие основные части ПЭВМ и их функции?

3.Какие виды памяти Вы знаете?

4.Что Вы понимаете под техническими возможностями ПЭВМ?

5.Правила подготовки компьютера к работе.

Лабораторная работа N 2

Знакомство с операционной системой (О С). Работа с каталогами и файлами.

Цель: изучить структуру MS-DOS, основные файлы ОС, также ознакомиться с командами работы с дисками, директориями и файлами.

Задания:

1.Изучить структуру MS-DOS.

2.Изучить структуру диска, файлов и каталога.

3.Ознакомиться с командами работы с дисками и директориями.

4.Ознакомиться с командами работы с файлами.

5.Выполнить практическую часть лабораторной работы.

6.Подготовить отчет к лабораторной работе.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1. Структура MS DOS.

Вся MS-DOS загружается с диска со специально отведенного на этом диске места. Это IBMBIO.COM, дополняющий основное BIOS компьютера и IBMDOS.COM, где находятся подпрограммы обращения к диску и другие дополнительные подпрограммы, расширяющие возможности BIOS и программирования в целом . Для общения с пользователем служит командный процессор, загружающийся в память из файла COMMAND.COM. Этот файл содержит ряд команд, представленных пользователю для работы с внешней памятью.

2. Структура диска, файлов и каталога.

Чтобы отыскать ту или иную дорожку на диске, его специальным образом размечают. Эта разметка заключается в следующем: диск представляется в виде дорожек и секторов. Дорожки представляют собой концентрические окружности на поверхности диска; информация записывается на дорожки не сплошь, а на отдельных отрезках - секторах. В каждом секторе хранится 512 байтов данных. Для отделения из сплошного потока данных полезной информации, на диске на некотором удалении от центра имеется отверстие, называе-

мое индексным. Это отверстие указывает на начало нулевого сектора.

В MS-DOS принято форматировать дискеты по 9 секторов на дорожку.

3.Холодный и горячий запуск OS.

При изготовлении компьютера в его постоянное запоминающее устройство (ПЗУ) записывается базовая система ввода /вывода (BIOS), которая приводится в активное состояние при включении компьютера в сеть. Однако для использования максимальных его возможностей требуется более развитое программное обеспечение. Его основу составляет OS, после загрузки которой в ОЗУ, компьютер будет готов к работе. Обычно OS хранится на каком-либо внешнем носителе и с помощью начального загрузчика загружается и запускается .После включения компьютера начнет осуществляться его самотестирование систем и загружаться OS, т.е. будет происходить инициация системы, при этом все параметры OS устанавливаются в исходное состояние - происходит холодный запуск OS. Как только запустится OS, она начнет искать на диске запускаемый файл autoexec.bat, в котором хранится информация в конфигурации системы, т.е. возможность запуска различных драйверов программного обеспечения.

При выходе из какой-либо программы в OS происходит обновление только командного процессора или каких-либо измененных пользователем ее частей ,необходимых для нормального функционирования OS, однако полного сброса системы не происходит - Это так называемый теплый запуск OS.

4.Команды работы с дисками.

-CHKDSK [drive:][path][filename][/F][/V] - проверка диска и вывод на экран результатов проверки;

filename - определяет файл (файлы) для проверки на фрагментацию;

/F - фиксирование ошибок на диске;

/V - вывод на экран пути и полного имени файла;

-DISKCOMP [drive1l:drive2:][/1][/8] - сравнение содержимого двух гибких дисков;

/1 - сравнение только одной стороны дискеты;

/8 - сравнение первых 8 секторов;

-DISKCOPY [drive1l:[drive2:][/1][/v] - копирование содержимого одного на другой;

/1 - копирование только одной стороны дискеты;

/V проверка копируемой информации;

-FORMAT [/Q][/U][/B][/S] [/T:tracks][/N:sectors][/1][/4][/8]- форматирование диска для использования в MS-DOS;

/Q - устанавливает режим быстрого форматирования;

/U - режим безусловного форматирования;

/B - резервирование места при форматировании для записи системных файлов;

/S - копирование системных файлов на форматируемый диск;

/T - число дорожек на сторону диска;

/N - число секторов на дорожку;

/1 - форматирование одной стороны дискеты;

/4 - форматирование дискеты на 360Кб;

/8 - форматирование по 8 секторов на дорожку;

-CHDIR (CD) - вывод на экран текущего каталога или его смена ;

CD [drive:] [path]

-CD / - выход в корневой каталог;

-CD ..- переход в каталог более высокого уровня

-MKDIR (MD) - создание подкаталога;

-RMDIR (RD) - удаление подкаталога при условии отсутствия в нем других подкаталогов или файлов.

5.Основная информация о файлах.

Файлом называется поименованная область однотипных данных на любом внешнем носителе. Имя состоит из восьми любых символов, кроме шаблонных - * и ?, а также трех символов - расширения файла. Эти части одного имени разделяются в OS точкой.

Например: COMMAND.COM, здесь COMMAND означает имя файла, а COM - его расширение, которое в OS имеет определенное значение.

.exe - запускаемый файл;

.com - то же;

.sys - системный для MS DOS;

.obj – объектный;

.bin – двоичный;

.bat - пакетный, позволяет изменить конфигурацию системы;

.pas - признак программного файла, в данном случае файл

содержит программу на языке Паскаль;

.bas - то же, но содержит Бейсик программу;

.win - такие и им подобные расширения указывают на то, что этот файл принадлежит какому-то программному продукту.

Файла с расширениями .exe, .com, .sys, .bat запускаются в командном режиме MS DOS. Исчерпывающей информацией о месте файла на диске является указание пути файла (path), которое содержит в себе последовательное перечисление подкаталогов, начиная с корневого, и заканчивая подкаталогом, где содержится необходимый файл, иначе, если он находится в корневом каталоге, указание пути не нужно. Например, допустим, что файл содержится в подкаталоге MINUS, а сам подкаталог MINUS входит в корневой каталог. Тогда путь к файлу будет такой: MINUS далее пишется имя файла. Также при записи полного имени файла указывается логическое имя диска, на котором содержится файл, оно, как и путь может отсутствовать. Итак, полное имя файла запишется в таком формате:

[drive:][path]filename ,

где drive: это имя дисковода, квадратные скобки означают необязательность параметра, заключенного в них, т.е. один из них может отсутствовать.

6.Команды работы с файлами.

-COPY [drive:][path]filename1[+...] [drive:[path]filename2] - копирование одного или нескольких файлов

filename1[+...] -определяет файл или файлы для копирования

[drive:[path]filename2]-определяет новое место или новое имя файла.

Возможны необязательные ключи:

/A - копирование ASCII файлов;

/B - копирование двоичных файлов;

/V - верификация файла.

- DEL, ERASE - удаление одного или нескольких файлов.

- ERASE [drive:}{path}filename[/P]:

filename - определяет файлы, подлежащие удалению;

/P - перед каждым удалением файлов выводится запрос:

Delete (Y/N)?. При нажатии на Y файл будет удален, на N - нет.

Если вместо имени использовать шаблон *.* перед выполнением команды выводится вопрос об удалении всех файлов в каталоге.

- RENAME,REN - изменить имя файла или группы файлов.

- REN [drive:][path]filename1 filename2

- TYPE [drive:][path]filename - позволяет вывести на экран содержимое текстового файла.

Порядок выполнения работы:

1. Вывести на экран оглавления каталогов текущего диска.

2. Создать каталог и подкаталог и выполнить с каталогами основные команды.

3. Создать текстовый файл в собственном каталоге и выполнить все команды ОС с файлами.

4. Отформировать гибкий диск и скопировать собственные файлы.

5. Пользуясь шаблонами групповых операций удалить все текстовые файлы.

6. Удалить все собственные каталоги.

ПРАКТИЧЕСКАЯ ЧАСТЬ.

1.а) C>DIR/p

LEX <SUB-DIR> 01-01-80 12:00

KATALOG1 <SUB-DIR> 01-01-80 12:01

б) C>DIR/w

[LEX] [KATALOG1]

2.C>md katalog1

C>dir

LEX <SUB-DIR> 01-01-80 12:00

KATALOG1 <SUB-DIR> 01-01-80 12:01

NC <SUB-DIR> 01-01-80 12:00

...

C>cd katalog1

C>dir

Volume in drive C is program

Directory of C:\KATALOG1\

. . .

O file (s) o byte

4112384 bytes free

C>md katalog 2

C>dir

Volume in drive C is PROGRAM

Directory of C:\KATALOG1\

KATALOG2 <SUB-DIR> 01-01-80 12:04

O file (s) 400 byte

4111984 bytes free

C>cd\

3. а) Создание текстового файла.

C>copy con myfile.txt

Прежде чем приступить непосредственно к работе на компьютере, пользователь может по своему вкусу изменить некоторые характеристики видеомонитора.

^Z

1 file(s) copied.

C>dir

. . .

LIBRO.TXT 20456 01-01-80 12:10

MYFILE.TXT 50 01-01-80 12:12

NEXT.TXT 4510 01-01-80 12:08

3 file(s) 25016 byte

30456823 bytes free

б) Копирование файла.

C>copy myfile.txt d: { копирование файла myfile на диск D}

C>copy myfile.txt+libro.txt+next.txt+con full.txt

{копирование файлов myfile.txt,libro.txt,next.txt и добавление ко всем этим файлам данных, вводимых с клавиатуры, после нажатия ^Z все это скопируется в один файл full.txt}.

в) Удаление файла:

C>del next.txt

C>dir

. . .

LIBRO.TXT 20456 01-01-80 12:10

MYFILE.TXT 50 01-01-80 12:12

FULL.TXT 25189 01-01-80 12:20

3 file(s) 45645 byte

30453599 bytes free

г) Переименование файла:

C>ren full.txt newfile.txt

4. Форматирование гибкого диска:

c>format a:

5. Удаление текстовых файлов :

c> DEL D:*.txt

6. Удаление каталога:

C>rd katalog1\ katalog2

C>rd katalog1

Контрольные вопросы.

1. Основные файлы операционной системы и их назначение.
2. Назначение гибких и жестких дисков.
3. Как загружается операционная система?
4. Какие команды используются при работе с директориями?
5. Что такое дерево каталогов и файлов?
6. Основные типы файлов .
7. Основные команды работы с файлами.

Лабораторная работа N 3

Работа с программной оболочкой Norton Commander

Цель: изучить основные операции и возможности оболочки.

Задания:

1. Изучить теоретическую часть лабораторной работы.
2. Ознакомиться с основными командами главного меню NC.
3. Изучить собственное меню NC.
4. Ознакомиться с функциями комбинаций управляющих клавиш в NC.
5. Подготовить отчет к лабораторной работе.

Теоретическая часть

Операционная оболочка NC выводит информацию в наглядном виде посредством панелей, вспомогательных меню. Все управление осуществляется управляющими клавишами с клавиатуры или мышью. Стрелки перемещают курсор по именам файлов или по пунктам интерактивного меню. Запуск файла осуществляется нажатием клавиши Enter, запускаются только файлы с расширениями и .com,.exe, .bat, .sys.

В нижней области экрана выводится строка - подсказка, расшифровывающая значение функциональных клавиш.

[F1] Help - при нахождении на диске пс.hlp выводит информацию о системе;

[F2]Menu - выводит окно с перечнем средств, используемых в данной конфигурации системы;

[F3] View - просмотр текстовых файлов;

[F4] Edit - редактирование текстовых файлов;

[F5] Copy - появляется окно с опросом, куда копировать файл;

[F6] RenMov -все файлы переписываются на новое место с возможным одновременным переименованием. При переносе все файлы будут удалены с текущего каталога;

[F7] MkDir -создание подкаталога;

[F8] Del - удаление файлов либо подкаталога

При нажатии на [F9] появляется собственное меню NC:

LEFT FILES COMMANDS OPTIONS RIGHT

1 LEFT/RIGHT.

NCD tree [Alt+F10] - быстрый переход по дереву каталогов;

Find file [Alt+F7] - поиск файлов;

History [Alt+F8] - список 15 последних введенных команд в командной строке**.**

Brief - отображаются только имена файлов;

Full - имена файла с дополнительной информацией;

Info - включение панели информации;

Tree - включение дерева каталогов;

Link - панель связи on/off - вкл/выкл панели;

Name - сортировка по именам;

Extension - сортировка по расширению;

Eime - сортировка по времени;

Unsorted - выдача как по команде dir;

Re-read - обновить информацию на панели;

Filter - выводятся только файлы, которые определены этой опцией;

Drive - выбор диска.

2.Files:

- дублирует функциональные клавиши [F1]-[F8]**;**

File Attributes - позволяет устанавливать атрибуты файла;

Send files - использование электронной почты;

Select Group - выделение файла по маске;

Unselect Group - отмена режима выделения файла;

Invert Group - выделение с инвертированием файлов;

Restore selection - восстановление исходного режима выделения;

Quit - выход из NC.

3.COMMANDS:;

EGA lines [Alt+F9] - переключение числа строк;

Swap panels [Ctrl+U] обмен панелями Panels on/off ;

[Ctrl+O] вкл/выкл панелей;

Compare directories сравнение каталогов обеих панелей;

Send/Receive mail команда запуска электронной почты;

Menu fill edit- меню пользователя;

4.OPTIONS:

- в приглашении MS-DOS выводится значение пути, в противном случае - только имя диска;

Key bar ([Ctrl+B])-включая Cofiguration - устанавливает основные режимы работы оболочки NC.

Advanced Options - включение дополнительных средств по конфигурации NC;

EXtension file edit - редактирование расширения файлов команд пользовательского меню([F2]);

Editor - устанавливает тип редактора, вызываемого по [F4];

Auto menus - при запуске NC автоматически появляется пользовательское меню;

Path prompt т вывод на экран строки подсказки значений функциональных клавиш;

Full screen - при включенном режиме панели занимают весь экран, при выключенном - половину;

Mini status - в нижней части экрана выводится информация о выбранном файле;

CLock - в правом верхнем углу выводится текущее время;

Save setup([Shift+F9]) - сохранение установок NC (при выключенном режиме Auto save setup).

Порядок выполнения работы:

1. Изучить и просмотреть оглавление каталогов на панели NC, а также другие виды панелей.
2. Создать каталог и подкаталог и выполнить с каталогами основные команды.
3. Создать текстовые файлы в каталоге и выполнить все основные команды с файлами, пользуясь меню NC.
4. Удалить все собственные файлы.

ПРАКТИЧЕСКАЯ ЧАСТЬ

1.Создание текстового файла:

Для создания текстового файла необходимо использовать команду редактирования файла с заданием имени файла [Shift+F4].

Например, необходимо создать текстовый файл text.txt, для этого нажимаем на клавиатуре клавиши:

Shif+F4 после этого на экране появляется запрос об имени файла, на который следует набрать: text.txt

далее загружается редактор, устанавливаемый меню NC и можно вводить информацию: information of tne file...

По окончании ввода нужно нажать клавишу [ESC] и ответить на запрос о сохранении файла "Y".

2.Создание подкаталога осуществляется нажатием клавиши [F7] и вводом имени подкаталога - TXTFILE.

3.Копирование файла в подкаталог(левая панель активна, правая подготовлена для принятия файла)при этом достаточно нажать F5 и Enter на запрос о пути и изменении файла ,если не предполагается изменение таковых

4.Просмотр файла - стандартная функция в NC:

F3 при этом курсор установлен на нужный файл

5.Удаление файла:

F8. Возможно использование шаблонов – нажатием серой клавиши "+" выбрать группу файлов или нажатием lns выбрать нужные файлы по своему усмотрению (в меню NC в Options/Configuration можно включить или выключить режим передвижения курсора вниз по нажатии lns. Для этого нужно нажать F9, если строка меню не выводится все время, выбрать пункт Options, затем - Configuration... и далее пометить или отменить соответствующий файл).

6.Работа с меню пользователя. Нажатием двух клавиш можно вызвать любую последовательность команд. Чтобы создать или отредактировать файл nc.mnu (в котором содержится информация о командах пользователя), нужно выбрать опцию Menu file edit из меню Commands и в появившемся окне вписывать команды по следующим правилам:

Сначала вводится имя клавиши, по которой активируется последовательность команд, затем ставится двоеточие (:) и пишется комментарий о выполняющихся командах. На следующей строке можно ввести последовательность необходимых команд.

Для сохранения новой последовательности нужно нажать F10 ,если файл редактировался через меню NC. Например , нужно осуществить проверку на вирус при нажатии клавиши F2 и при нажатии 1 запустить игру. Для этого введем следующую информацию в файл nc.mnu:

F2 проверка на вирус диска C c:/aids/aidstest c/g/s/p

1: Запуск игрушки d:/game/myth/myth

Для активирования этих команд в NC достаточно выбрать меню пользователя по команде F2 и после появления окна, нажать соответствующую клавишу.

Контрольные вопросы

1.Расскажите об основных файлах обеспечения NC.

2.Как создать текстовой файл?

3.Для чего используются клавиши F5,F8,F9 ?

4.Основные функции главного меню NC и назначение собственного меню.

Лабораторная работа N 4

Ознакомление с интегрированной средой Турбо Паскаля. Программирование линейных алгоритмов

Цель: изучить интегрированную среду Турбо Паскаля (ТР). Реализовать простейшую программу в среде ТР.

Задания:

1. Ознакомиться с интегрированной средой.
2. Изучить основные операторы по вводу исходных данных, оператор присваивания и вывод результатов в простейшей линейной программе.
3. Составить и реализовать линейную программу по заданному варианту.
4. Подготовить отчет к лабораторной работе.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ.

Интегрированная среда (ИС) Турбо Паскаля даёт возможность ввода программы на языке Паскаль, редактирования, сохранения результатов, выполнение программы и т.д.

Панель интегрированной среды состоит из:

- основного меню;
- окна редактирования;
- строки подсказок (функции функциональных клавиш).

Основное меню состоит из 7 пунктов.

I. FILE.

Меню File предлагает разнообразные средства для загрузки существующих файлов, создание новых файлов и сохранение файлов на диске. При загрузке файл автоматически помещается в буфер текстового редактора. В состав этого меню входят:

Load (Open) ([F3]).

Команда Load используется для загрузки файла в текстовый редактор.

При выборе элемента Load на экране появляется окно "Load File Name". В этом окне вы можете ввести:

а) имя файла, либо

б) шаблон, определяющий группу файлов.

Если попытаться загрузить новый файл, а в буфере есть корректировавшийся, но не сохраненный файл, то на экране появляется окно Verify с предупреждением

File Name not saved. Save? (Y/N)

Pick ([Alt+F3])

Команда pick позволяет загрузить файл, выбранный из списка указания, который содержит:

*имена 7 последних загружавшихся файлов,

*имя файла, находящегося в текущий момент в редакторе,

*элемент -load file-

New.

По команде New создается новый файл. По умолчанию этому файлу присваивается имя NONAME.PAS. При записи нового файла на диск вы можете изменить это имя.

Save(F2)

Команда save используется для сохранения на диске текущего редактируемого файла.

Write to (Save as…)

По команде Write to выполняется запись файла на диск. В отличие от команды Save, сохраняющей файл с тем же именем, с которым он был загружен в редактор, команда Write to позволяет дать файлу новое имя. Если при вводе имени файла вы не укажете расширение, то редактор автоматически добавит к имени расширения .PAS.

Directory

Эта команда выводит список файлов указанного каталога. При этом можно ввести шаблон, определяющий группу файлов.

Change Dir

Эта команда выводит имя текущего каталога и позволяет сделать текущим другой каталог.

Os Shell

По команде Os Shell выполняется временный выход из интегрированной среды в среду MS-DOS (в текущий каталог). Для возврата в ИС введите команду MS-DOS Exit.

Quit (Alt+x).

Завершает работу и возвращает управление операционной системе MS-DOS.

II. EDIT.

По команде Edit основного меню вызывается встроенный редактор, предназначенный для создания исходных текстов программ в ИС.

Редактор позволяет вводить до 64К текста, при этом можно использовать любые символы из основного и расширенного набора. Размер строки - до 248 символов.

Редактор предоставляет широкий набор команд для работ с фрагментами текста(блоками). Блоки визуально выделяются на экране

1. Отметить начало блока - [Ctrl+K + B]

По этой команде редактор помечает начало блока в текущей позиции курсора.

Блок выделяется цветом только после того, как установлен маркер конца блока.

2. Отметить конец блока - [Ctrl+K+K]

По этой команде редактор помечает конец блока в позиции слева от курсора.

Блок выделяется цветом только после установки маркера начала блока.

3. Отметить одиночное слово - [Ctrl+K+T]

По этой команде редактор помечает одиночное слово как блок. Слово определяется как последовательность символов, ограниченная с двух сторон одним из следующих символов:

пробел <> , ; . () [] ^ ' * + - / $

4. Скопировать выделенный блок - [Ctrl+K +C]

По этой команде ранее помеченный блок копируется в область, начинающуюся с текущей позиции курсора.

5. Переместить выделенный блок - [Ctrl+K+V]

По этой команде ранее выделенный блок перемещается из его первоначального положения в область, начинающуюся текущей позиции курсора.

6. Удалить выделенный блок - [Ctrl+K+Y]

По этой команде ранее выделенный блок удаляется.

7. Отменить выполнение блока/Выделить предыдущий блок -[Ctrl+K+H]

Включается/Выключается режим выделения блока. Команды манипуляции с блоками выполняются только в том случае, если блок

выделен.

8. Читать файл, с диска - [Ctrl+K+R]

При вводе этой команды на экране появляется окно Read File Name (имя файла, который нужно прочитать с диска).

9. Записать выделенный блок в файл на диске - [Ctrl+K+W]

ПРИ ВВОДЕ ЭТОЙ КОМАНДЫ НА ЭКРАНЕ ПОЯВЛЯЕТСЯ ОКНО WRITE FILE NAME.

10. Распечатать выделенный блок - [Ctrl+K+P]

11. Удалить текущую строку - [Ctrl+Y]

12. Найти в тексте образец - [Ctrl+Q+F]

По этой команде осуществляется поиск строки длиной до 30 символов. Строка поиска может содержать любые символы, включая управляющие.

Для прерывания выполнения команды поиска используйте команду прерывания [Ctrl+U].

13. Найти в тексте заданный образец и заменить его - [Ctrl+Q+A]

Эта команда выполняется аналогично команде [Ctrl+Q+F], за исключением того, что она позволяет заменить найденную строку любой другой строкой, имеющий длину до 30 символов.

14. Прервать выполнение команды - [Ctrl+U]

Эта команда позволяет прервать выполнение любой команды, когда та делает паузу для ввода.

15. Перейти на позицию ошибки - [Ctrl+Q+W]

По этой команде редактор выводит в окне редактирования последнее по времени сообщение об ошибке и помещает курсор в позицию этой ошибки.

III. RUN.

Run (Ctrl+F9)

По этой команде выполняется компиляция программы, если в нее были внесены изменения со времени последней компиляции, и ее запуск.

Program Reset ([Ctrl+F2])

Команда Program Reset завершает текущий сеанс отладки, выводит из режима отладки и удаляет с экрана текущую границу выполнения. Кроме того, по этой команде освобождается память, занимаемая программой, и закрываются все открытые файлы.

Go To Cursor ([F4])

Эта команда позволяет начать/продолжить выполнение программы от текущей строки до той строки в окне редактирования, на которой находится курсор.

Trace Info ([F7])

Выполняется следующий оператор. Если это оператор вызова подпрограммы, то трассировка продолжается на следующем операторе подпрограммы.

Step Over ([F8])

Эта команда работает так же, как команда Trace Info, но с одним важным отличием: если встречается оператор вызова подпрограммы, то вся эта процедура выполняется за один шаг, и отладчик делает паузу на операторе, следующем за оператором вызова.

User Screen ([Alt+F5])

Команда User Screen позволяет увидеть на экране пользователя выходную информацию программы. Для возврата в ИС следует нажать любую клавишу.

IV. COMPILE.

Compile ([Alt+F9])

Выполняется компиляция текущего редактируемого файла. Во время компиляции на экране отображается окно, в котором выводится следующая информация: имя основного файла, имя компилируемого файла, количество откомпилированных строк, размер доступной оперативной памяти. При успешном завершении компиляции в окне выводится сообщение

Success : Press any key

Make ([F9])

По команде Make выполняется компиляция

а) текущего редактируемого файла

либо

б) основного файла (если он задан с помощью Compile/Primary File).

Build

Выполняется компиляция либо текущего редактируемого файла, либо основного. Кроме того, если в компилируемом модуле имеется список ссылок (Uses) на другие модули, то заново компилируются ВСЕ модули, указанные в этом списке.

Destination

Используется для указания места сохранения выполняемого кода. При установке Disk выполняемый код сохраняется на диске, в файле типа EXE.

При установке **Memory** выполняемый код сохраняется в оперативной памяти.

Если код сохраняется на диске, то в качестве имени выполняемого файла выбирается имя основного файла (Primary File), либо имя последнего откомпилированного файла.

Find Error

Позволяет найти в программе место расположения ошибки, возникшей на этапе выполнения.

Адрес ошибки указывается в формате "сегмент : смещение".

Primary File

Используется для установки имени файла, который будет компилироваться по командам Make и Build. Если задан основной файл, то независимо от того, какой файл редактируется, команды Compile/Make, Compile/Build всегда работают с основным файлом.

Get Info

По команде Get Info отображается окно с информацией о текущем файле.

V. OPTIONS.

5.1. Compile - содержит:

Range Cheking - позволяет установить или отменить проверку диапазона. При установке в значение On компилятор генерирует код, который проверяет, соответствуют ли индексы массивов и строк границам диапазона.

Stack Cheking - позволяет установить или отменить проверку стека. При установке в значение On происходит проверка на доступное пространство в стеке для локальных переменных. Если проверка дает отрицательный результат, то программа прекращает работу из-за ошибки времени выполнения.

I/O Cheking - позволяет установить или отменить проверку ошибок во время выполнения операций ввода/вывода.

Force Far Calls - при установке этой опции все вызовы процедур функций интерпретируются как межсегментные вызовы.

Align Data - позволяет переключаться между выравниванием переменных и типизированных констант на границу байта и на границу слова.

Var-String Cheking - эта команда позволяет выбрать строгую или ослабленную проверку ошибок в строковых параметрах. При строгой проверке компилятор сравнивает размер

формального строкового параметра типа Var с размером фактически передаваемого параметра.

Boolean Evaluation - позволяет выбрать полную или короткую схему вычислений булевского выражения.

Numeric Processing - при установке сопроцессора генерируется код сопроцессора и допускается использование типов данных с плавающей точкой по стандарту IEEE. При установке программной обработки допускается использование только стандартного типа Real.

Emulation - разрешает или запрещает компоновку с библиотекой программ, которые будут эмулировать работу сопроцессора в случае его отсутствия. При компиляции модуля (Unit) эта установка не используется.

Debud Information - позволяет дать указание компилятору генерировать информацию для отладки компилируемой программы. При установке в значение On компилятор создает таблицу с номерами строк каждой процедуры. Эта таблица позволяет поставить в соответствие объектному коду номера строк исходного текста.

Local Symbols - позволяет дать указание компилятору генерировать информацию о локальных символах. Если элемент Local Sumbols установлен в On, то при работе в ИС можно будет проверять и модифицировать локальные переменные модуля.

Conditional Defines - определяет символы, ссылки на которые содержатся в директивах условной компиляции в исходном коде.

Memory Sizes - позволяет определить конфигурацию карты памяти для генерируемого исполняемого кода (.EXE).

5.2. Linker:

Map File - определяет объем информации, помещаемой в табличный файл, который генерируется компоновщиком. В зависимости от установки, табличный файл содержит:

* только информацию о сегментах, либо

* информацию о сегментах; список всех символических имен и соответствующих им адресов; список точек входа; таблицы модулей, содержащие номера строк исходного текста и соответствующие строкам адреса.

Link Buffer - позволяет указать тип памяти, используемой в качестве буфера компоновщика. При установке Memory для

буферизации используется оперативная память, при установке Disk - используется диск.

5.3. Environment:

Config Auto Save - позволяет сохраняет изменения, производимые в опциях компилятора и ИС.

Edit Auto Save - помогает предотвратить потерю исходного файла: при выполнении команд File/Os Shell, Run/Run (Ctrl+F9), Trace Over (F8) текущий редактируемый файл автоматически сохраняется. Backup Files - при выполнении File/Save ИС автоматически сохраняет резервную копию исходного файла (в файле .BAK).

Tab size - устанавливает жесткий размер табуляции для редактора (от 2 до 16 позиций).

Zoom Windows - активное окно увеличивается до размеров полного экрана.

Screen Size - позволяет выбрать изображение в 25 строк, в 43 строки (для EGA), либо в 50 строк (для VGA). Доступно только при аппаратной поддержки.

5.4. Directories:

Turbo directory - устанавливается имя каталога, в котором производится поиск файла конфигурации (.TP) и файла подсказки.

EXE & TPU Directory - устанавливается имя каталога, в который записываются файлы с расширением .EXE и .TPU.

Include Directories - устанавливаются имена каталогов, которые содержат включаемые файлы.

Unit Directories - указывает каталоги, которые содержат файлы модулей. При указании нескольких каталогов их имена разделяются точкой с запятой.

Object Directories - указывает каталоги, содержащие объектные модули.

Pick File Name - с помощью данной опции можно установить имя файла указания.

Current Pick File - Содержит имя текущего файла. Обычно этот элемент меню выключен и используется только в информационных целях.

5.5. Parameters:

Позволяет определить параметры, которые передаются программе, выполняемой в ИС.

5.6. Save Options:

Позволяет сохранить в файле конфигурации все выбранные опции.

5.7. Retrieve Options:

Загружает файл конфигурации, сохраненный ранее по команде Save Options.

VI. DEBUG:

Evaluate - открывается окно, позволяющее увидеть значение переменной или выражения, а также задать новое значение переменной.

Call Stack (Ctrl+F3) - открывается окно, в котором отображается состояние стека вызовов. Эту команду можно использовать только в режиме отладки.

Find Procedure - позволяет выполнить поиск процедуры или функции в текущей программе. Используется только в режиме отладки.

Integrated Dubuggind - при установке этой опции в On появляется возможность отлаживать программу в ИС: устанавливать точки останова выполнять программу по шагам и работать с командами отладки в меню Run.

Standalone Debugging - одновременно с установкой опции Compile/Destination в значение Disk в выполняемый файл .EXE добавляется информация для отладки, которая может использоваться автономным Turbo-отладчиком.

Display Swapping - если установлено значение Smart, то отладчик будет выполнять переключение экрана при выполнении вызова процедуры или функции, даже если подпрограмма не выполняет вывода на экран.

Если установлено значение Always, то переключение экрана происходит при выполнении каждого оператора.

Если установлено значение None, то отладчик не будет выполнять переключения экрана.

Refresh Display - эту команду можно использовать для восстановления изображения на экране ИС.

VII. BREAK/WATCH:

Add Watch (ctrl+F7) - позволяет добавить элемент в окно просмотра. Новый элемент включается перед текущим выражением просмотра.

Delete Watch - эта команда удаляет текущее просматриваемое выражение (только в том случае, если окно просмотра является видимым).

Edit Watch - выводит текущее выражение просмотра в окне Input.

Remove All Watch - удаляет все элементы из окна просмотра .При этом окно Watch сокращается до минимального размера.

Toggle Breakpoint - устанавливает в текущей строке точку останова. Если там уже имеется точка останова, то она отменяется.

Clear All Breakpoint - команда отменяет все ранее установленные точки останова.

View Next Breakpoint - перемещает в окне редактирования курсор к следующей точке останова. При этом никакие операторы не выполняются.

Изучение интегрированной среды начнем с реализации простой программы на ТР. В качестве примера возьмем программу линейной структуры.

При программировании линейного процесса используются следующие операторы:

Оператор присваивания, операторы ввода/вывода

1. Оператор присваивания. Оператор имеет вид:

a:=в;

где ":=" - знак присвоения;

a - имя переменной;

b - константа, выражения, переменная.

При выполнении оператора значение В присваивается переменной А.

При этом тип В должен соответствовать типу А. Допускается присваивание, когда А вещественного, В целого типа.

Примеры:

K:=ROUND (EXP(16*LN(X)));

Y:=A*K+7*K;

2. Ввод исходных данных можно организовать различными способами:

- в разделе констант задаются соответствующие значения;
- с помощью оператора присваивания присваивается значения переменным.
- с помощью оператора ввода. Оператор имеет следующие виды:

а) Read (список переменных);

При выполнении оператора переменным указанных в списке вводятся значения с клавиатуры в том же порядке, как они указаны в списке. При этом тип значений должен соответствовать типу переменной.

Пример: Read(A,B,C)

б) Readln (список переменных);

При выполнении этого оператора после ввода значения курсор переводится на начало следующей строки.

Например: Readln(B,Д); Read(Z);

Значения B и Д вводятся на одной, значение Z на другой строке.

3. Вывод данных.

Для вывода данных используется следующий оператор вывода:

а) Write(a);

где a - выражения, переменная и константа.

При выполнении оператора значение A выводится на экран в соответствии с типом A.

Например:

Write('C=',C,7+x,y);

б) Writeln(a);

При выполнении оператора после вывода значения A курсор переносится на начало следующей строки.

в) Write(в:m:n),

где в - переменная, m - количество позиций, отведенных для печати B;

n - количество позиции дробной части B.

Например: Write ('Y=',Y:8:3)

Решение одного варианта.

Задание. Составить программу вычисления K по формуле:

$$k = e^{a} + b$$

где

$$a = \frac{2 * \cos(x - \frac{\pi}{6})}{0.5 + \sin^2 y}; \quad b = 1 + \frac{z}{3 + \frac{z^2}{5}};$$

x = 3,4; y = 2,21; z = 1,09;

Текст программы:

```
program linproc;
const
pi=3.14125;
  var
a,b,k,x,y,z : real;
  Begin
  readln(x,y,z);
 a:=(2*cos(x-pi/6)/(0.5+sqr(sin(y)));
        b:=1+(z/(3+z*z/5));
       k:=exp(a)+b;
   writeln('x=',x:4:2,'y=',y:4:2,'z=',z:4:2);
   writeln('a=',a:6:4,'b=',b:6:4,'k=',k:6:4);
    End.
```

Результат программы:

```
3.4 2.21 1.09
X = 3.40 Y = 2.21 Z = 1.09
```

A = - 1.6870 B = 1.3669 K = 1.5520

Контрольные вопросы

1. Какие команды включены в пункты основного меню интегрированной среды Турбо Паскаль?
2. Как осуществляется загрузка программного файла в интегрированной среде?
3. Последовательность выполнения программного файла в ТП.

4. Какие основные команды используются при программировании линейного процесса?

5. Как осуществляется просмотр результатов программы.

Лабораторная работа N 5.

Программирование и реализация алгоритмов разветвляющихся структур

Цель: Приобретение навыков составления программ разветвляющейся структуры с помощью операторов условия и выбора языка Паскаль .

Задания.

1.Ознакомьтесь с теоретической частью.

2. Перепишите свой вариант из таблицы.

3. Напишите программу решения задачи и реализуйте ее на ЭВМ.

4. Подготовить отчет к лабораторной работе.

Теоретическая часть.

Программа разветвляющейся структуры обязательно содержит условия, в зависимости от которых предусматривают выбор одного из вариантов последовательностей операторов из нескольких заданных.

Для организации разветвлений в программах используются операторы условия и выбора.

Оператор условного перехода.

Оператор условного перехода относится к числу управляющих операторов Паскаля. Его работа нарушает естественный порядок выполнения операторов в программе . Существует две формы записи - краткая и полная. Краткая форма записи оператора имеет общий вид:

IF <УСЛОВИЕ> THEN <ОПЕРАТОР>

Полная форма записи оператора имеет следующий общий вид:

IF <условие> THEN <оператор>
ELSE <оператор>

<Оператор> может иметь вложенную структуру, т.е. иметь в себе условие. В качестве оператора могут быть использованы любые операторы языка Паскаль. В качестве условия в общем

случае может быть логическое выражение. Работа короткого условного оператора заключается в том, что в первую очередь проверяется условие, заданное логическим выражением. Если значение логического выражения равно true (истина),то выполняется оператор следующей после THEN, если же значение логического выражения равно false (ложь), то управление передается следующему оператору программы после условного.

В полной форме оператор условного перехода записывается:

IF <условие> then <S1> else <S2>;

где S 1 - любой оператор Паскаль или составной оператор;

S 2 - любой оператор Паскаль или составной оператор.

При выполнении оператора вычисляется логическое выражение. Если значение его равно "TRUE", то управление передается оператору S1, в случае "FALSE" - S2. Особенностью условного оператора в языке Паскаль является то, что он может быть вложенным. Т.е. операторы внутри условного могут, в свою очередь, быть условными операторами только полной формы.

Например.

```
if  x>0  then  if  x=1  then  y=5
else  y=sqr (x)  else  y=abs(x);
```

Пример 1. Написать программу для решения следующей задачи, используя оператор условного перехода.

$$y = \begin{cases} \ln(3) * \ln\left|a + b + t\right| & \text{если } \frac{1}{t^2} > \frac{1}{k}; \\ e^{t+k} & \text{если } \frac{1}{t^2} \le \frac{1}{k} \end{cases}$$

где $t = e^{\frac{\ln 0.83}{3}} \cos a + e^{\ln 2/3} \sin b$, k=cos(-0,5arctg b + arctg a)

Текст программы:

```
PROGRAM  Razvetvlenie1;
  VAR a,b,t,k,y : real;
BEGIN   READLN (a,b);
   t:=exp(ln(0.83)/3) * cos(a)+ exp(ln(2)/3) * sin(b);
```

```
  k:=cos (-0.5*arctan(b) + arctan(a));
  WRITE ('выполнилось условие');
  IF (1/sgr(t))>(1/k) THEN BEGIN y:=ln(3)*ln(abs(a+b+t));
  WRITELN ('первое...',y)
 End
  ELSE BEGIN y:=exp (t+k); WRITELN ('второе...',y) end;
END.
```

Результат программы:

```
1   10
Выполнилось условие первое                ... 2.7298353200E
10   1
выполнилось условие второе                 ....2.1050896900E
```

Оператор выбора.

Если алгоритм предусматривает вычисление более чем двух или трех вариантов решения задачи, в Паскале предусмотрено использование оператора выбора CASE.

Оператор выбора CASE обеспечивает организацию разветвлений путем выбора одного из нескольких операторов.

Оператор выбора состоит из служебного слова CASE, с - селектора и служебного слова OF , и далее задаются варианты идентифицирующих меток и последовательности операторов, соответствующих только выбранному варианту. Заканчивается оператор служебным словом END.

Селектором в операторе выбора является выражение значения, которое может принадлежать типам целый, литерный, логический, либо типу, определяемому пользователем, а также перечисляемому и ограниченному. Оператор выбора содержит список операторов, перед которыми записывается одна или несколько констант, отделяемых двоеточием. Константы должны быть одного типа с селектором. Выбор оператора определяется совпадением значения селектора и константы, стоящей перед оператором. Общий вид записи оператора имеет вид:

```
CASE c OF
 n 1 : P 1
  n 2 : P 2
 .......
  n n :  P n;
end;
```

где c - селектор; ni - метки операторов, Pi операторы. В качестве оператора может быть использован любой оператор языка, а также составной оператор.

Рассмотрим вышеприведенный пример 1, используя оператор выбора CASE.

```
PROGRAM  razvet 2;
 VAR a,b,y,k,t:real; d:boolean;
   BEGIN  READLN (a,b);
      t:=exp(ln(0.83)/3) * cos(a)+ exp(ln(2)/3) * sin(b);
      k:=cos (-0.5*arctan(b) + arctan(a));
      d:=(1/sqr(t))>(1/k);
  CASE  d  OF
  TRUE : BEGIN y:=ln(3)*ln(abs(a+b+k));
WRITE (' условие 1-',y);end;
  FALSE : BEGIN y:=exp (t+k);
WRITELN ('условие 2-',y)
   END; END;
END.
```

Результат программы.

```
1   10
условие первое              ... 2.7298353200E+00
10   1
условие второе              ....2.1050896900E+00
```

Контрольные вопросы

1. Что такое разветвляющийся вычислительный процесс?
2. Какие виды условного операторов вы знаете?
3. Когда используется оператор варианта?
4. Какого типа константы используется в качестве меток в операторе варианта?

Лабораторная работа N 6

Программирование и реализация алгоритмов циклических структур

Цель работы: приобретение навыков составления программы циклической структуры.

Задания:

1. Ознакомиться с теоретической частью.
2. Ознакомиться с постановкой задачи.
3 . Написать программу для решения задачи на ЭВМ и выполнить ее.
4 . Подготовить отчет к лабораторной работе.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Программа циклической структуры позволяет многократно повторять вычисление группы операторов при изменении одновременно одного или нескольких параметров. В зависимости от постановки задачи различают циклы с известным и неизвестным числом повторений. В языке Паскаль имеются специальные операторы цикла: FOR, WHILE, REPEAT , позволяющие организовать циклы с известным и неизвестным числом повторений. Оператор цикла FOR - называются оператором цикла с параметром. Он служит для организации цикла с известным числом повторений.

Общий вид записи оператора:

FOR i:=m to n DO S ;

или

FOR i:=n DOWNTO m DO S ;

где i - параметр цикла (не может быть величиной действительного типа);

m и n - начальное и конечное значения параметра цикла;

S - тело цикла (простой или составной оператор)

Значения m и n записываются константами или выражениями того же типа, что и тип переменной цикла.

При ключевом слове TO шаг изменения параметра цикла равен "+1", а при DOWNTO равен "- 1".

Работа оператора заключается в следующем. Вычисляются выражения m и n. Параметр цикла i принимает начальное зна-

чение- m и вычисляется тело цикла, изменяясь на шаг, сравнивается с конечным значением - n. При повторении цикла, если параметр цикла не превышает - n, то управление передается в тело цикла и вычисления повторяются. При каждом повторении параметр цикла меняет свое значение на единицу и работа оператора повторяется. Выход из цикла осуществляется после того, как параметр цикла превышает конечное значение, при этом параметр цикла не сохраняет свое значение. Например, для вычисления значения **y**, где

y=1+1/2 + 1/3 + ... 1/n

можно использовать следующий фрагмент программы:

```
y: = 0;  n:=5 ;
FOR  I:= 1  to n do  y:=y+1/I;
```

или

```
FOR I : =n  DOWNTO  1 DO  y:=y+1/I;
```

Оператор цикла WHILE - называются оператором цикла с предусловием, позволяет организовать цикл, в котором число повторений вычислений зависит от заданного условия.

Общий вид записи оператора:

```
WHILE  B DO  S;
```

где B - логическое выражение,

S - тело цикла содержащее простой или составной оператор.

Сначала вычисляется значение логического выражения, и если оно равно true (истина) , то выполняется тело цикла и снова проверяется логическое выражение до тех пор пока не станет равным false (ложь).

В этом случае оператор обеспечивает выход из цикла. Так как оператор WHILE управляет циклом по условию, он используется для организации цикла с неизвестным числом повторений.

Оператор цикла REPEAT - оператор цикла с постусловием позволяет организовать цикл с неизвестным числом повторений. В отличие от оператора WHILE в операторе REPEAT проверка условия производится после каждого повторения. Организация цикла при помощи этого оператора состоит из двух служебных слов, которые также являются операторными скобками, внутри которых располагается само тело цикла.

Общий вид записи оператора имеет вид:

REPEAT S UNTIL B;

где S - оператор или последовательность операторов;

B - логическое выражение .

Если логическое выражение имеет значение true (истина), происходит выход из цикла иначе при значении false (ложь) - операторы цикла выполняются ещё раз.

Рассмотрим пример с использованием операторов цикла.

Вычислить:

$$y = \frac{e^{3\cdot\ln(x^2+2)}}{\sqrt{(5+x)^2-1}} + \frac{\sin 2\cdot x}{\ln\left|x^3-215\right|},$$

где $x_н$=7, $x_к$=10, шаг изменения h_x=0,2

```
PROGRAM   tsikl1;
 VAR n : INTEGER;
      A,b,x,h,y   : REAL ;
   BEGIN  a:=7; в : = 10 ; h: = 0.2;
      n:= trunc ((в-a)/h)+1;
x:=a;
     FOR i: = 1  TO n  DO  BEGIN
       y: = exp (3*Ln(x*x+2)) / s q r t(s q r (5+x)-1);
       y:= y+sin (2*x)/ln(abs(x*x*x-215) );
     x:=x+h;
WRITELN  (y);
END;
END.
```

Решение одного варианта .

Задание: Выполнить заданную функцию в промежутке от 7 до10 с заданным шагом h=0,2 с использованием оператора Repeat.

```
PROGRAM   tsikl2;
 VAR  в,h,y,x  : real;
 BEGIN  read ln (x,в,h);
 REPEAT
  y: = exp (3*Ln(x*x+2)) / sqrt (sqr (5+x)-1);
 y:= y+sin (2*x)/ln(abs(x*x*x-215)) ;
  WRITELN  (y) ;
  x: = x + h;
 UNTIL  x > в;
 END.
```

Результат программы:

1193.0381
14793.4012
19445.9165
25220.0070
32314.0171
40946.3972
51358.9150
63817.9023

Контрольные вопросы.

1.Что такое циклический вычислительный процесс?

2.Какие операторы цикла вы знаете ?

3.Какие типы переменных используются в качестве параметра в операторе цикла с параметром.

4.Различие между операторами цикла WHILE и REPEAT.

Лабораторная работа N 7

Работа с регулярными типами данных

Цель работы: освоение практических навыков составления программ на алгоритмическом языке Паскале с использованием регулярных типов (массивов).

Задания.

1. Ознакомиться с теоретической частью. Изучить методику решения задач с индексированными переменными на ЭВМ.
2. Получить вариант задания .
3. Выполнить реализацию задания согласно этапам решения задач на ЭВМ.
4. Оформить отчет.

Теоретическая часть.

Массивом называется упорядоченное множество конечных чисел, снабженных номерами (или индексы).

Массив является одномерным, если его элемент имеет один индекс. Например:

$A=\{a_1,a_2,a_3,a_4,a_5\}$- одномерный массив из пяти элементов.

$X=\{x_1,x_2,x_3,x_4\}$ - одномерный массив из 4-х элементов.

Массив является двумерным, если его элемент имеет два индекса(это матрица чисел).

Например:

A(3,4) - двумерный массив, состоящий из 12-ти элементов.

Элементом массива называется переменная с индексом, которая состоит из имени массива и одного (или двух) индекса, значение которого определяет положение элемента в массиве. В программе для обращения к элементу одномерного массива указывается имя массива, в квадратной скобке номер элемента в массиве.

Например, массив B состоит из пяти элементов. Для обозначения четвертого элемента массива B в программе указывается имя массива B и в квадратной скобке номер элемента 4,т.е. B[4].

Для обращения к элементу двумерного массива указывается имя массива, а в квадратной скобке номер строки и столбца, на пересечении которых находится данный элемент. Номера разделяются запятой.

Например, элемент, который находится на пересечении второй строки и четвертого столбца матрицы M, обозначается в программе следующим образом: M[2][4] или M[2,4].

В качестве индекса могут быть использованы имя простой переменной или арифметическое выражение.

Например, D[K] - элемент одномерного массива D, номер которого равен значению переменной K.

S[2*I+K] - порядковый номер элемента массива S определяется значением выражения 2*I+K.

M[2,I] - элемент двумерного массива M находится на пересечении второй строки и столбца, номер которого равен значению переменной I.

Переменные с индексами могут быть использованы в программе наравне как простые переменные. Индексированных переменных как и простых до применения их в операторном разделе надо описывать либо в разделе TYPE, либо в разделе VAR.

Общая форма записи описания индексированных переменных одномерного массива в разделе имеет следующий вид:

VAR

<имя переменной>: array[тип индекса] of <тип элементов массива>

В качестве индекса применяются все типы кроме real, longint, integer.

Часто в программе в качестве типа индекса применяются ограниченные типы: ограниченный целый, ограниченный символьный.

Например: Описание в разделе VAR одномерного массива A, состоящий из 100 вещественных чисел имеет следующий вид

```
VAR   a:array [1..100]of real;
```

Особенностью использования массивов в языке Паскаль является то, что эта конструкция описывается, как тип и может быть описан в разделе типов.

Тогда описание этого же массива имеет следующий вид

```
TYPE   massiv=array[1..100]of real;
 VAR   a:masiv;
```

В случае применения в качестве типа индекса перечисляемых типов имеется возможность увеличить читаемость и наглядность программы.

Например:

```
TYPE  mecyaz=(yanvar, fevral,...dekabr)
VAR   t,r:mecyaz;
```

дает возможность применения частичных переменных, обозначающих температуру конкретного месяца

t[январь], t[февраль], t[март] и т.д.

Двумерные массивы описываются в программе подобно как одномерные .

Общая форма записи описания двумерного массива в разделе VAR имеет следующий вид

VAR

<имя двумерного массива>:array[тип индекса, тип индекса] of <тип элемента двумерного массива>

Например, двумерный массив A, имеющий 10 строк и 20 столбцов, составленный из целых чисел, в разделе VAR списывается следующим образом:

```
VAR   A:array [1..10,1..20]of integer;
```

Этот же массив в разделе TYPE и VAR описывается следующим образом:

```
TYPE   matriza=array[1..10,1..20]of integer;
VAR    a:matriza;
```

Решение одного варианта

Задача: Вычислить значения функции

$$Z = \frac{Xi}{i}$$

где Xi - элемент массива ($x_1, x_2, \ldots, x_{20}$)

Текст программы:

```
program FUN2(input, output);
var  Z,X:array[1..20]of real;
     i:integer;
          BEGIN
            FOR i:=1 to 20 do  read(X[i]) ;
            FOR i=1 to 20 do   Z[I]:=X[i]/i
            FOR i:=1 to 20 do  writeln('Z(',I,')= ',Z[i])
          END.
```

Контрольные вопросы.

1. Что такое индексная переменная? Виды массивов и их различия.

2. Описание регулярного типа (массива) в Турбо Паскале.

3. Как описываются многомерные массивы в языке Паскаль?

4. Какие типы используются для типа индексов и типов компонент массива?

Лабораторная работа №8

Программирование и реализация на ЭВМ алгоритмов с использованием конструкций функций и процедур

Цель работы:

1.Освоение практических навыков составления программ с процедурой.

2. Освоение практических навыков составления программ с функцией.

Задания:

1. Ознакомится с теоретической частью
2. Получить вариант задания
3. Составить программу на языке Паскаль
4. Реализовать ее на компьютере
5. Оформить отчет

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Существуют такие задачи, где некоторые совокупности действий должны выполнятся в нескольких различных местах программы. Чтобы избежать повторения, указанную группу операторов можно записать один раз и обращаться к ней, когда в этом возникает необходимость.

Обособленную группу операторов, которую можно выполнять многократно, обращаясь к ней из различных мест программы, называют процедурой.

1.Место процедуры в программе и синтаксис ее описания

процедура в программе пишется в промежутке между разделом описания переменных - VAR и разделом операторов.

Синтаксис описания процедуры в программе имеет следующую общую форму записи:

<описание процедуры>::=<заголовок процедуры>;<блок>

Блок (или тело процедуры) является тем фрагментом программы, который является процедурой, причем этот блок определятся точно так же, как и блок, являющийся телом Паскаль программы.

<заголовок процедуры>::= procedure <имя процедуры> [(список формальных параметров)]

Имя процедуры есть идентификатор, (например: INTEGRAL, MAX и др.) а список формальных параметров определяется следующим образом:

<секция формальных параметров>::=<имя>{,<имя>}:<имя типа>{var<имя>},<имя>}:<имя типа>},

где <имя>- идентификатор, используемый в качестве фактического параметра

Обращение к процедуре выполняется в разделе операторов программы и имеет следующий общий вид:

<обращение к процедуре>::=<имя процедуры> [(список фактических параметров)].

При составлении программ на Паскале используются следующие виды процедур:

1. Процедура с параметром.
2. Процедура без параметра.
3. Процедура с параметром, относящийся к сложному типу.

Процедура без параметра в программе имеет следующий общий вид:

```
PROCEDURE <имя процедуры>
Begin
<раздел операторов>
end;
```

Различные особенности описаний процедур и способы их использования можно объяснить на примере следующей задачи

```
U = MAX(X+Y,X*Y)
V = MAX(0.5,U)
```

Программу решения этой задачи без использования процедур можно записать в виде

```
program maxa(input,output);
 var   x,y,u,v:real;
BEGIN        read(x,y);
  if (x+y)>(x*y) then u:=x+y else u:=x*y;
  if 0.5>u then v:=0.5 else v:=u;
 writeln ('u=',u, 'v=',v)
END.
```

В данной программе два условных оператора, каждый из которых предназначен для решения одной задачи: нахождение большего из двух заданных вещественных значений и присваивания некоторой переменной полученного результата. По-

этому алгоритм решения этой частичной задачи целесообразно объявить процедурой для переменных a, b и s.

If a>b then s:=a else s:=b

С использованием данной процедуры программа имеет следующий вид:

```
Program maxG(input, output);
Var  x, y, u, v real;  a, b, s:real;
Procedure max2A;
Begin
    If a>b then s:=a else s:=b
End;

Begin       read (x,y);
            a:=x+y; b:=x+y' max2A; u:=S;
            a:=0.5; b:=U; max2A; v:=S;
            writeln(`v=`,v);
end.
```

Max2A является процедурой без параметра.

2. Процедура с параметрами

Процедура с параметрами имеет следующий общий вид:

Procedure <имя процедуры>(список формальных параметров);

Begin <раздел операторов>

End;

На месте списка фактических параметров могут быть параметры значения и переменных, иначе называемые параметрами входа и выхода. Параметры переменных пишутся после слово VAR.

Решение одного варианта.

Рассмотрим пример использования процедуры с параметром значения, относящегося к сложному типу.

Пример: В программе после определения значений переменных X и У регулярного типа требуется найти

U = MAX {Xi}

V = MAX {Yi} (i=1,2,3,...n)

Решение

Из условий задач видно, что дважды должен использоваться частичный алгоритм нахождения наибольшей компоненты вектора, то этот алгоритм удобно оформить в виде процедуры,описание которой может иметь вид

```
procedure maxar (a:vekt;var s :real);
  begin
   s:=a[1];
   for i:=1 to n do
   if a[i]>s then s:= a[i];
end;
```

С использованием процедуры maxar программа решения задачи имеет следующий вид

```
program max (input,output);
  const n=20;
    type        vekt=array[1..n] of real;
      var u,v:real; i:integer; x,y:vекt;
procedure maxar (a:vekt;var s:real);
  begin
  s:=a[1];
  for i:=1 to n do
  if a[i]>s then s:=a[i];
end;
BEGIN
  for i:=1 to n do
  read (x[i],y[i]);
  maxar(x,u), maxar(y,v);
writeln(`u=`, u,`v=`, v)
END.
```

Правила записи и применение процедур

1. Число фактических параметров в операторе процедуры должно быть равно числу формальных параметров в описании процедуры. Если у процедуры нет параметров, то оператор состоит из имени процедуры без круглых скобок.

2. При записи оператора процедуры необходимо иметь в виду, что соответствие между фактическими и формальными параметрами устанавливается путем их сопоставления слева направо в соответствующих списках.

3. Тип каждого фактического параметра должен соответствовать типу формального параметра.

4. При задании каждого очередного фактического параметра надо особенно внимательно следить за тем, что представляет в процедуре соответствующей ему формальный параметр – значение или переменную.

Место расположения процедуры функции в программе и ее синтаксис

Если необходимость использования какой-либо функциональной зависимости встречается в нескольких местах программы, то было бы нерационально каждый раз выписывать соответствующий алгоритм. Как в случае процедур операторов, удобнее однажды определить требуемую функциональную зависимость, дав ей некоторое имя, а в случае необходимости использовать эту зависимость путем указания ее имени и задания конкретных значений аргументов. Процедура функция размещается как процедура оператора между блоками описания переменных var и разделом операторов.

Синтаксис описания функции имеет следующий вид

<описание функции>::=function<имя функции> (<формальный параметр>:<имя типа>,...):<имя типа>;

Описание процедур функции имеет следующие отличия от описания процедуры оператора

1.Описание начинается служебным словом function.

2.В заголовке функции указывается имя типа значения описываемой функции.

3.В теле процедуры функции должен присутствовать хотя бы один оператор присваивания, в левой части которого фигурирует имя описываемой функции, причем хотя бы один оператор такого вида должен быть выполнен.

Например f(n)=n! можно описать на Паскале следующим образом

```
 function fact(n:integer):integer;
var
  i,k:integer;
begin
k:=1;
for i:=1 to n do k:=k*i;
fakt:=k;
end;
```

Вызов функции

Синтаксически вызов функции определяется точно также, как и оператор процедуры - это имя функции, за которым следует взятый в круглые скобки список фактических параметров. Например, fakt (9) - вызывает процедуру функ-

цию fakt, подставляя вместо формального параметра n фактический параметр, значение которого равен 9.

Контрольные вопросы:

1. Что называется процедурой и характерные особенности ее использования?
2. Где помещается процедура в программе?
3. Основные отличия конструкции процедуры от функции.
4. Чем отличается вызов процедуры в основной программе от вызова функции?
5. Что такое формальный параметр и фактический параметр?

Лабораторная работа N9.
Работа с комбинированными типами данных.

Цель работы: изучить структуру записи и программирование алгоритмов с комбинированным типом.

Задания:

1. Изучить теорию работы и использование записей.
2. Для заданного варианта составить программу с использованием записей.
3. Реализовать программу и получить результат на ЭВМ.

Теоретическая часть.

Запись представляет собой совокупность ограниченного числа данных (элементов) различного типа, называемые полями.

При описании записей все элементы заключаются в операторные скобки RECORD и END, где задаётся имя поля (элемента) и его тип. На тип полей не накладываются ограничения. Поэтому поля записей также могут быть записями.

Описать записи в программе можно двумя способами:

1. В разделе описания типов. Общий вид описания:

```
TYPE <имя типа>=RECORD
<имя 1-го поля>:<тип>;
<имя 2-го поля>:<тип>;
   . . . . . .
<имя n-го поля>:<тип>
END;
```

Пример:

```
TYPE ZAPIS=RECORD
       FAM:String[15];
       GR: 1965..1977;
       GP: Integer;
       NGR: 20..56
     END;
VAR  STUD:ZAPIS;
```

где STUD-переменная комбинированного типа и она задаёт имя записи.

2. В разделе описания переменных. Общий вид описания:

```
VAR <имя переменной>: RECORD
                        <имя 1-го поля>:<тип>;
```

<имя 2-го поля>:<тип>;
. . . . . .
<имя n-го поля>:<тип>
END;

Пример: VAR ST:RECORD
FAM:String[15];
NGR:Integer;
OTSEN:Record MAT,FIZ,INF:Integer End
END;

При обращении к элементу записи используется имя записи (имя переменной типа запись) и имя поля, разделенные точкой, например:

STUD.NGR, ST.OTSEN.INF, STUD.FAM.

Над элементом записи можно выполнять действия, допустимые для данных его типа.

Например:

READLN(STUD.FAM,ST.OTSEN.MAT);

STUD.FAM:='Ахмедов'; ST.OTSEN.INF:=4; STUD.NGR:=34;

При использовании типа запись получаются длинные операторы, так как постоянно приходится писать полное имя записей при обращении к элементу данной записи. Можно сократить текст, воспользовавшись оператором WITH. Он позволяет один раз указать имя записи, которое выносится в заголовок оператора, а в блоке используются только имена элементов.

Общий вид оператора:

WITH <список имен переменных> DO

Begin <операторы, содержащие имена элементов> End;

Например:

WITH STUD DO
Begin Readln(FAM,NGR,GP) End;

или

WITH ST,OTSEN DO
Writeln(FAM,MAT,INF);

При определении комбинированного типа в него можно включать вариантную часть. Это позволяет размещать в одних и тех же полях записи компоненты различных типов, обычно определяемые значением поля признака.

Вариантная часть записи включает поле признака (пере-

ключателя) скалярного типа. Выбираемый вариант записи определяется значением поля признака. За полем признака следуют метки, соответствующие каждому допустимому значению этого поля. Каждая метка является заголовком для списка полей, который определяет тип варианта, соответствующего данной метке.

Общий вид описания записи с вариантами:

```
TYPE RT=RECORD
  A:T1;
  B:T2;
  . . . . .
  CASE N:TN OF
  C1:(R11:TP11;. . .);
  . . . . . .
  CK:(RK1:TPK1;. . .)
  END;
VAR  Z:RT;
```

где N - переменная-признак (переключатель);

TN- тип переменной N;

этому же типу должны принадлежать метки C1,C2,...,CK. Каждой метке соответствует набор полей R11,R12,... . Эти поля являются компонентами варианта. Если какой либо метке вообще не соответствует поля, то записывают:

```
CL:0;
```

Следует обратить внимание на следующие особенности использования записей с вариантами:

1. Любая запись может иметь одну вариантную часть (CASE);

2. Вариантная часть должна размещаться только после постоянной части, поэтому эта часть не закрывается со словом END;

3. Среди имен полей не должно быть одинаковых.

4. Перед засылкой информации в запись необходимо переключателю присвоить соответствующее значение.

5. Допускается вложение вариантов в типе запись.

Решение одного варианта

Постановка задачи: Каждый компьютер характеризуется названием и техническими характеристиками : скорость работы в тыс.оп/сек, объем ОЗУ, длины машинного слова, количество пикселей в графическом режиме. Отсор-

тировать компьютеры по объему памяти (в порядке возрастания).

```
program Com;
 Type
  computer=record
        Name  : string[10];
        Speed : real;
        Razr  : byte;
        Pix   : word;
        Ram   : word
        end;
 Var EVM:array[1..10] of computer;
     k:computer; i,j,m:integer;
 Begin Write('Количество компьютеров:'); Readln(m);
     for i:=1 to m do
      with EVM[i] do
        begin
     Write('Введите имя компьютера....'); Readln(Name);
     Write('Введите скорость работы...'); Readln(Speed);
     Write('введите разрядность.....'); Readln(Razr);
     Write('количество пикселов.....'); Readln(Pix);
     Write('объём ОЗУ.....'); Readln(RAM);
        end;
     for j:=1 to m-1 do
      for i:=1 to m-1 do
       begin
       if  EVM[i].RAM>EVM[i+1].RAM  then  begin
                          k:=EVM[i];
                          EVM[i]:=EVM[i+1];
                          EVM[i+1]:=k;
                           end
        end; writeln;
            writeln('I Name I Speed I Razr I PIX I RAM I');
            writeln('I------I-------I------I-----I-----I');
         for I:=1 to m do
         with EVM[i] do
       begin
          writeln('I',Name,'I',Speed,'I',Razr,'I',PIX,'I',
RAM,'I');
       end;
```

```
        writeln('I---------------------------------I');
    End.
```

Контрольные вопросы.

1. Дайте определение комбинированному типу. Что такое запись, элементы записи, поля записи?
2. Описание записи в программе (на примерах).
3. Правила корректного представления элементов записи в программе.
4. Какие операции выполняются над полями записи?
5. Объясните оператор присоединения.
6. Комбинированный тип с вложенной структурой. Правила доступа к полям записи вложенной структуры.

Лабораторная работа N 10

Работа с множественными типами данных.

Цель: изучить использование множественного типа при программировании задач на PASKALe.

Задания:

1.Изучить теоретическую часть.

2.Для заданной задачи составить программу.

3.Реализовать программу и получить результат.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Под множеством языка Паскаль понимают ограниченный, неупорядоченный набор различных элементов одинакового типа. Все элементы множества должны быть значениями одного типа. Тип элементов множества называется базовым типом этого множества.

Базовыми типами множеств на Паскале могут быть только скалярные типы. При этом базовый тип множества не должен иметь более 256 возможных значений, а порядковые значения нижней и верхней границы базового типа должны находиться в диапазоне 0..255.

Значением множественного типа является множество. Конкретные значения множественного типа задаются с помощью так называемого конструктора множества, представляющего собой список элементов множества, заключенный в квадратные скобки. Множество может не иметь вообще элементов, то есть быть пустым.

При определении множественного типа значений прежде всего необходимо задать некоторый базовый тип, из значений которого создаются конкретные значения множественного типа.

В программе множества задаются следующим образом

```
SET OF<базовый тип>;
```

Описать множества можно в разделе TYPE либо в разделе VAR .

Например

```
TYPE  MNOG = SET OF 1..50;
   DETAL = (PP,TRANZ, KON1, REG);
VAR  CHIS10: MNOG; N: DETAL;
```

LB: SET OF 'A'..'Z',

где CHISLO,N,LB - имена множеств.

Переменные множественного типа называют также переменными множествами. Как и любые переменные, они описываются в разделе переменных.

С множествами возможны следующие двуместные операции:

Объединением двух множеств называется множество, состоящее из элементов, входящих хотя бы в одно из этих множеств. Например, если в качестве множества A взять [1,2,3,4,5] а в качестве множества B-[2,5,6,7,8],то объединением этих множеств будет [1,2,3,4,5,6,7,8].

Для обозначения этой операции используется знак "+".

Пересечением двух множеств называется множество, состоящее из элементов, одновременно входящих и в первое, и во второе множества. Например, пересечением рассмотренных выше множеств будет множество [2,5]. Для обозначения операции пересечения используется знак "*".

Разностью двух множеств называется множество, состоящее из элементов первого множества, не входящих во второе множество. В этом случае используется знак "-" .

Наряду с операциями *,+, - над значениями множественного типа определены и некоторые операции отношения.

Среди операций отношения над значениями множественного типа особое место занимает специальная операция проверки вхождения элементов в множество, обозначаемая служебным словом in. В отличие от остальных операций отношения, в операции in первый операнд должен принадлежать базовому типу, а второй - множественному типу значений, построенному на основе этого базового типа.

Результатом операций отношения, как обычно, является булевское значение.

Выполнение операцией < и > над операндами множественного типа недопустимо.

При значении True в операциях: A=B - множества совпадают, A<>B-множества не совпадают ,A <=B - множество A входит в множество B,A>=B - множество B входит в множество A.

Решение одного варианта

Составить множество Lb из латинских букв и множество PR содержащих знаки препинания из введенной строки исходного текста.

```
Program rr;
  type simvol=set of 'a'..'z'; znaki='!'..'?'
  var LB:simvol; PR:znaki; C,I,J:char;
    Begin
     readln;
     LB:=[]; PR:=[];
     repeat read(c);
           if C in ['a'..'z'] then LB:=LB+[C]
           else if C in ['!'..'?'] then PR:=PR+[C];
     until eoln;
        writeln('Латинские буквы');
      for I:='a' to 'z' do if I in LB then write(I:2);
        writeln;
      writeln('Знаки препинания');
     for J:='!' to '?' do if J in PR then write(J:2)
     end.
```

Введенная строка:

FYWA.*12 PRLLLLO;ORPA:PR?8

Полученный результат:

Латинские буквы

A F L O P R W Y

Знаки препинания

. : ; ?

Контрольные вопросы.

1. Дайте определения типу множества и его элементам. Какие типы могут быть базовыми?
2. Как описывается множество в разделе описания Паскаль программы?
3. Задание значений переменным множественного типа (возможные варианты).
4. Как выполняются операции сложения, вычитания и умножения множеств (на примерах)?
5. Объясните на примерах использования операций отношения для множеств.
6. Операция “IN” и ее использования.

Оглавление

Редактор Ахметжанова Г.М.