# Міністерство освіти і науки України

## Черкаський Державний Технологічний Інститут

# Кафедра інформатики

### Факультет економіки і управління

#### Контрольна робота

З курсу інформатики і компютерної техніки

*Перевірив виконав студент*

*викладач кафедри 1курсу феу групи зу-112*

*Олійник Г.Т. Шкарбан С. В.*

№ 10

Определить время падения камня на поверхность земли с высоты h :

10 CLS

20 INPUT "Высота h"; h

30 t = SQR(h / 9.81)

40 PRINT t

50 END

№ 79

Вычислить: (1+sin0,1)(1+sin0,2)…(1+sin10)

10 CLS

20 m = 1

30 FOR k = .1 TO 10 STEP .1

40 m = m \* (1 + SIN (k))

50 NEXT k

60 PRINT "m ="; m

70 END

№ 756

Даны действительные числа x,a, ( >0,x <1).Вычислить с точностью  значение:

1+ a(a-1).k..(a-k+1)  xk

10 CLS

20 INPUT "x...", x

30 INPUT "a...", a

40 INPUT "Eps...", Eps

50 sum = 0: k = 1: pred = 0: fact = 1

' pred - предыдущий член последовательности для каждого шага цикла

' нужен для учета точности. fact - факториал

60 mul = a

70 FOR i = 0 TO k: mul = mul \* (a - i + 1): NEXT ' произведение в числителе

80 fact = fact \* k ' факториал в знаменателе

90 cur = mul / fact \* (x ^ k) ' текущий член последовательности

100 sum = sum + cur: k = k + 1

110 IF cur - pred > Eps GOTO 140

120 pred = cur

130 GOTO 60

140 PRINT sum

150 END

№ 215-в

Дано натуральное число m ,действительные числа a1,…an, последовательность a1,…an. Определить число соседств двух чисел одного знака, причем модуль 1-го числа должен быть больше модуля 2-го числа.

10 CLS

20 INPUT "Введите m...", m

30 DIM a(m)

40 FOR i = 1 TO m

50 PRINT "a("; i; ")"; : INPUT a(i)

60 NEXT i

70 FOR i = 1 TO m - 1

' произведение двух чисел с одинаковым знаком больше нуля

80 IF a(i) \* a(i + 1) > 0 THEN IF ABS(a(i)) > ABS(a(i + 1)) THEN k = k + 1

90 NEXT i

100 PRINT k

№ 339-г

Даны целые числа а1…an, (в этой последовательности могут быть повторяющиеся члены). Вычислить сколько чисел входит в последовательность по одному разу.

10 CLS

20 INPUT "Введите n...", n

30 DIM a! (n)

40 FOR i = 1 TO n

50 PRINT "a("; i; ")"; : INPUT a!(i)

60 NEXT

70 nepovt = 0

80 FOR i = 1 TO n

90 found = 0

100 FOR j = 1 TO n

110 IF i <> j AND a! (j) = a!(i) GOTO 140

120 NEXT j

130 nepovt = nepovt + 1

140 NEXT i

150 PRINT "Неповторяющихся чисел:"; nepovt

160 END

№ 145-а

Последовательность x1,x2,… образовано по закону x1=0, x2= 5/8

Xi=Xi-1/2+3/4\*Xi-2, i=3,4…

10 CLS

20 DIM x (20)

30 LPPRINT “ Последовательность x(20) “

40 LPRINT

50 x(1)=0: x(2)=5/8

60 FOR I=1 TO 20

70 x(i)=x(i-1)/2+3/4\*x(i-2)

80 NEXT i

90 FOR i =1TO 20

100 LPPRINT USING “###.######”: x(i):

110 IF i MOD 5=0 THEN LPPRINT

120 NEXT i

130 LPRINT

140 END

Задача N 145A Последовательность х (20)

0.000000 0.625000 0.312500 0.625000 0.546875

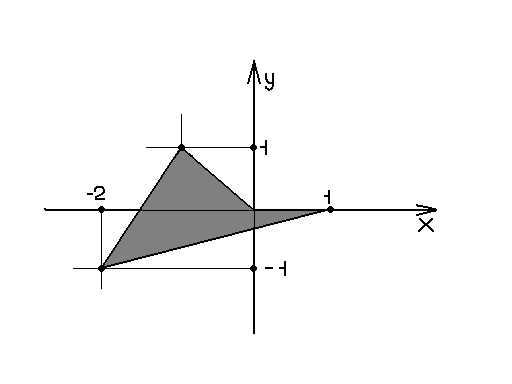
0.742118 0.781250 0.947266 1.059570 1.240234

1.414795 1.637573 1.879883 2.168121 2.493973

2.873077 3.307018 3.808317 4.384422 5.048449

№ 59-и

Даны действительные числа x, y. Определить принадлежит ли точка с координатами (x,y) заштрихованной части плоскости.



10 CLS

20 LPRINT

30 LPRINT “Задача №59i”

40 LPRINT

50 INPUT “ введите x , y=: x , y

60 LPRINT

70 IF x>=0 THEN

80 IF (y<=0) AND (3\*y-x>= -1) THEN

90 GOTO 220

100 ELSE

110 GOTO 200

120 END IF

130 ELSE

140 IF (y+x<=0) AND (y-2\*x<=3) AND (3\*y-x>= -1) THEN

150 GOTO 220

160 ELSE

170 GOTO 200

180 END IF

190 END IF

200 LPRINT “точка А не принадлежит заштрих. области"

210 GOTO 230

220 LPRINT “точка А принадлежит заштрих. области"

230 LPRINT

240 LPRINT USING “\ \#####.##”: “x=”: x:

250 LPRINT USING “\ \#####.##”: “y=”: y:

260 LPRINT

270 END

Задача N 59i

x = 0.50 y = 0.10

точка А не принадлежит заштр. Плоскости

x = - 0.40 y = - 0.20

точка А принадлежит заштр. плоскости

**Windows-приложения**

Windows — *интегрированная программа.* Под управлением оболочки Windows могут работать не только специальные программы, разрабо­танные для эксплуатации **в с**реде Windows (Windows-приложения), но и "обычные" программы, работающие в среде DOS, т.н. DOS-приложения (DOS-прикладные программы). Оболочка Window обеспечивает эффективный и комфортабельный обмен информацией между отдельными программами, выполняемыми под ее управлением. Здесь речь в первую очередь идет о Windows-пpилoжeнияx. С понятием интегрированности связывают обычно также возможность совместного использования ресурсов компьютера различными прог­раммами. Так, к примеру, принтер, подключенный к компьютеру, может с одинаковым успехом использоваться всеми программами на конкурентной основе. Причем все операции, связанные с необходимостью перекодировок, смен драйверов (например, при переходе от печати текстов к выводу иллюстраций) берет на себя оболочка.

Большинство пользователей привлекает в среде Windows не только и не столько комфортабельность самой оболочки, сколько специфика ре­ализованных в этой среде приложений. Особенности реализации в среде Windows даже знакомых пользователям по работе в DOS прикладных программ *(приложений)* практически позволяют рассматривать Windows-версии этих программ как совершенно новые продукты.

Работа в o6oлочке Windows и в Windows-приложениях предполагает своеобразную перестройку "образа жизни". "Жизнь" пользователя в среде 'Windows сопряжена с "мышиным" управлением, обменами данными между отдельными программами и параллельным выполнением. Стандартизация интерфейсов отдельных Windows-приложений позволяет легко переходить от одного приложения к другому, не начиная каждый раз с нуля (хотя бы в плане способов и средств управления).

В фирменной поставке пакета Windows находится несколько при­ложений. Все они объединены в группу *Accessories* (аксессуары, инстру­менты). Это небольшие по размеру и возможностям прикладные програм­мы, составляющие "джентльменский набор" пользователя. Им далеко до профессиональных специализированных пакетов. Но они прекрасно ил­люстрируют возможности оболочки и обеспечивают некоторый мини­мальный сервис. Более того, весьма полезно начинать знакомство с серь­езными пакетами именно с соответствующих средств этой группы. Так, например, поработав некоторое время с текстовым редактором *Write,* в дальнейшем можно легко перейти к использованию таких профессиональ-ных пакетов обработки текстов, как *Word для Windows, Lotus Ami Professional, WordPerfect для Windows'* т.п. Кроме того, в приложениях из группы *Accessories* реализованы многие новинки, характерные именно для последней версии 3.1 оболочки Windows (работа с объектами, новые шрифты...).

*зафиксировать курсор* кнопкой мыши — и операция выполняется. С помощью того же манипулятора можно перемещать пиктограммы и окна по экрану, менять их размер, открывать и закрывать их — и все это при минимальном использовании клавиатуры для ввода каких бы то ни было директив. Кроме того, для любителей традиционного интерфейса DOS реализована возможность выхода на этот уровень. При разработке графического интерфейса Windows не последнюю роль играли и эргономические соображения: учтены требования к цветовой гамме, сочетаниям цветов, шрифтам, формам и размерам пиктограмм и окон. По сравнению с некоторыми другими пакетами внешнее оформление оболочки Windows может быть признано "спар­танским" вследствие отсутствия излишеств и за деловой стиль. Понятие "графически-ориентированный" включает в себя для Win­dows также и соответствие изображения на экране последующему изображению на твердой копии (распечатке). В этом плане можно считать, что в оболочке Windows реализован принцип WYSIWYG ( *What Yon See Is What You Get* = To, что Вы видите, то и получаете), до сих нор бывший привилегией относительно небольшого числа программ- С помощью TrueType-шрифтов этот принцип нашел в рамках Windows 3.1 свое дальнейшее развитие.

Windows обеспечивает *независимый запуск* и *параллельное выпол­нение нескольких программ.* Большинство других оболочек и операционных

**Windows 95**

Объектно-ориентированный подход

При создании Windows 95 фирма Microsoft в полной мере реализо­вала объектно-ориентированный подход. Поскольку именно он лег в основу новой операционной системы, вначале скажем несколько слов о том, что такое ориентация на объекты.

Понятие «объектно-ориентированный» возникло в программиро­вании сравнительно недавно. Когда вычислительная мощность ма­шин была невысока, о создании объектно-ориентированных сис­тем не могло быть и речи. Основой всего был программный код. Программисты записывали последовательности команд для выпол­нения тех или иных действий над данными, которые оформлялись в модули и процедуры. Для работы с каждым объектом создавалась своя процедура.

*Объекты, их свойства и* методы

Постепенно с увеличением производительности вычислительных систем процедурный подход начал заменяться объектным. На пер­вое место выдвинулся объект, а не код, который его обрабатывает. На уровне пользователя объектный подход выражается в том, что интерфейс представляет собой подобие реального мира, а работа с машиной сводится к действиям с привычными объектами. Так, пап­ки можно открыть, убрать в портфель, документы — просмотреть, исправить, переложить с одного места на другое, выбросить в корзину, факс или письмо — отправить адресату и т. д. Понятие объекта оказалось настолько широким, что до сих пор не получило строгого определения.

Объект, как и в реальном мире, обладает различными свойствами. Программист или пользователь может изменять не все свойства объектов, а только некоторые из них. Можно изменить имя объек­та, но нельзя изменить объем свободного места на диске, который также является его свойством. Свойства первого типа в языках про­граммирования носят название read/write (для чтения и записи), а свойства второго — read only (только для чтения).

Метод — это способ воздействия на объект. Методы позволяют со­здавать и удалять объекты, а также изменять их свойства. Напри­мер, для того чтобы нарисовать на экране точку, линию или плоскую фигуру, составляются разные последовательности кодов или програм­мы. Пользователь, однако, применяет для отображения этих объек­тов один метод Draw( ), который содержит коды для отображения всех объектов, с которыми он работает. За такое удобство приходится пла­тить тем, что объектно-ориентированные системы могут работать только на достаточно мощных вычислительных установках.

Процедурный подход в ранних ОС

До настоящего времени во всех операционных системах преобла­дал процедурный подход. Для того чтобы произвести в системе ка­кое-либо действие, пользователь должен был вызвать соответству­ющую программу (процедуру) и передать ей определенные пара­метры, например, имя обрабатываемого файла. Программа выпол­няла над файлом указанные действия и заканчивала работу. При этом пользователь в первую очередь имел дело с задачей обработки документа, а затем уже с самим документом. В давние времена, когда ЭВМ не были персональными, пользователь описывал дейст­вия, которые должна была выполнить задача, на некоем странном языке, называемом языком управления заданиями (JCL—Job Con­trol Language).

С появлением терминала язык управления заданиями упростился и постепенно превратился в командную строку, однако на первом месте все равно находилась процедура обработки документа, а сам документ играл вспомогательную роль.

Следующим этапом упрощения работы с машиной стал создание различного рода операционных оболочек (сначала текстовых),которые «спрятали» от пользователя командную строку DOS. Ввод последовательности символов, из которой состоит команда опера­ционной системы, свелся к нажатию одной функциональной кла­виши или щелчку мыши. Самой распространенной из таких «надстро­ек» над операционной системой стала оболочка Norton Commander,

Однако основным «инструментом» пользователя все еще оставалась клавиатура. Качественный переход произошел после того, как поя­вились графические оболочки. Теперь пользователь в основном ра­ботает с устройством указания, таким как мышь, трекбол или план­шет, а не с клавиатурой (разумеется, это не относится к работе внут­ри самих приложений, например, в текстовых редакторах). Ему не нужно помнить почти никаких команд операционной системы. Для того чтобы запустить приложение, достаточно щелкнуть мышью на его изображении или на «значке» (автор предпочитает называть его пиктограммой).

От процедурного подхода к объектно-ориентированному

В начале 90-х гг. процедурный подход все еще преобладает, однако намечаются и некоторые признаки объектно-ориентированного. Например, уже в Windows 3+ можно поставить в соответствие кон­кретному документу приложение для его обработки. Тогда же поя­вился метод объектного связывания и встраивания (OLE), позволя­ющий щелчком на изображении объекта неявно запустить прило­жение, которое его обрабатывает, а после окончания обработки вернуться в предыдущее приложение.

С OLE тесно связан так называемый метод редактирования доку­ментов «на месте» (in-place). Если в документ встроен объект, ко­торый должен обрабатываться конкретным приложением, то при щелчке на этом объекте нужное приложение неявным образом за­пускается, причем в рабочем поле не изменяется ничего, кроме па­нелей инструментов. Например, если в тексте, который обрабаты­вается в редакторе Microsoft Word, есть таблица, созданная в редакторе Microsoft Excel, то при щелчке на ней произойдет замена *nанелей инстр*ументов Excel. Пользователь может обрабатывать документ совсем другим приложе­нием, даже не подозревая об этом,

Еще один механизм, который упростил работу и приблизил эру объ­ектно-ориентированного подхода, называется «Drag & Drop», что в буквальном переводе означает «перетащить-и-оставить». Работая этим методом, вы щелкаете кнопкой мыши (как правило, левой) на изображении объекта, перемещаете его по экрану при нажатой кнопке и отпускаете кнопку, когда указатель окажется в нужном месте экрана. Таким образом, процедуры копирования, перемеще­ния и удаления стали объектно-ориентированными.

Что делал пользователь, когда ему нужно было удалить файлы в опе­рационной системе MS-DOS? Он запускал процедуру удаления фай­лов, передавая их имена в качестве параметров:

del FILEI.TXT FILE2TXT

Это действие ничем не напоминает реальный мир, в котором вы просто выбрасываете ненужные

Бумаги в мусорную корзину. На первом месте для пас стоит объект (бумага), над которым выполня­ется процедуры (переноса в мусорную корзину), R операционных оболочках, которые работают под управлением Windows 3.1, такое действие уже реализовано как объектное-ориентированное — с по­мощью механизма «Draw & Drop». Например, в оболочке Norton Desktop можно схватить мышью файл и перенести его на изобра­жение мусорной корзины. Этого достаточно для удаления файла. Так работа на персональном компьютере все больше напоминает манипуляции с объектами в реальном мире.