***Теория!***

* Арифметические функции

Турбо-Паскаль содержит множество стандартных функций и процедур, сосредоточенных в модуле system.tpu (который по умолчанию находится в библиотеке turbo.tpl, но может быть выделен оттуда и использован отдельно).

В этом параграфе мы рассмотрим общие функции и процедуры, не имеющие привязки к какому-либо разделу нашего пособия. Остальные будут проанализированы в соответствующих параграфах (например функции и процедуры работы со строками - в параграфе "Строковые типы", функции и процедуры работы с динамической памятью - в параграфе "Указатели" и т.д.)

|  |  |
| --- | --- |
| Функция или процедура | Действие |
| *Тригонометрические функции* | |
| f. ArcTan(X: Real): Real; | Возвращает арктангенс аргумента |
| f. Cos(X: Real): Real; | Возвращает косинус аргумента |
| f. Sin(X: Real): Real; | Возвращает синус аргумента |
| f. Pi: Real; | Возвращает число Пи:  3.1415926535897932385 |
| *Функции для выделения целой и дробной части числа* | |
| f. Frac(X: Real): Real; | Возвращает десятичную часть аргумента |
| f. Int(X: Real): Real; | Возвращает целую часть аргумента. |
| *Логарифмическая и экспоненциальная функции* | |
| f. Ln(X: Real): Real; | Возвращает натуральный логарифм аргумента. |
| f. Exp(X: Real): Real; | Возвращает экспоненту аргумента |
| *Степенные функции* | |
| f. Sqr(X: тип):тип; | Возвращает квадрат аргумента. Тип результата соответствует типу аргумента. |
| f. Sqrt(X: Real): Real; | Возвращает квадратный корень аргумента |
| *Абсолютное значение* | |
| f. Abs(X:тип):тип; | Возвращает модуль аргумента. Тип результата соответствует типу аргумента. |

В отличии, например, от языка программирования Бейсик, ТП содержит очень небольшой объем арифметических функций. С помощью несложных операций он может быть расширен.

ab= exp(b\*ln(a)); {а в степени b}

logab=ln(b)/ln(a); {логарифм по произвольному основанию}

Tan(x) = sin(x)/cos(x); {тангенс x}

сTan(x) = 1/tan(x); {арктангенс x}

Arcsin(x)=arctg(x/sqrt(1-sqr(x))); {арксинус x}

Arccos(x)=Pi/2-arcsin(x); {арккосинус x}

Arcctan(x)= Pi/2-arctan(x); {арккотангенс x}

Ch(x)=(exp(x)+exp(-x))/2; {гиперболический косинус x}

Sh(x)=(exp(x)-exp(-x))/2; {гиперболический синус x}

* Процедуры выхода

ТП содержит в модуле **system** три процедуры выхода, позволяющие аварийно завершит программу или блок.

|  |
| --- |
| procedure **Exit** |
| Процедура **Exit** немедленно завершает работу текущего блока. Если текущий блок - программа, то происходит выход из нее. |
| procedure **Halt**[ ( Exitcode: Word ) ]; |
| Процедура **Halt**производит выход из программы с кодом завершения **ExitCode.** |
| procedure **RunError**[(Errorcode: Byte)]; |
| Процедура **RunError**производит выход из программы, эмулируя ошибку времени выполнения с номером **Errorcode.** |

Процедура **Exit** чаще всего используется для выхода из процедур и функций.Процедура **Halt** - для остановки программы с целью передачи операционной системе кода завершения, который затем может быть проанализирован, например, с помощью оператора BAT-файла:  
**IF ErrorLevel код завершения ...**

Процедура **RunError** применяется обычно для отладки программы. Ее использование позволяет наиболее простым методом определить место ошибки, а также проверить реакцию программы на критические ситуации.

* Различные процедуры и функции

|  |
| --- |
| *Обработка больших объемов данных (до 64К)* |
| procedure **FillChar**(var X; Count: Word; value) |
| Процедура **FillChar** заполняет переменную или произвольную область памяти **X** значением **Value** любого типа (размером в 1 байт), в количестве **Count**байт |
| procedure **Move**(var Source, Dest; Count: Word); |
| Процедура**Move**копирует область памяти **Source** в **Dest** размером в **Count** байт. |
| *Работа с байтами одного слова* |
| function **Hi**(X): Byte; |
| Функция **Hi** возвращает старший байт аргумента. X должна быть переменной типа **integer** или **word** (см. параграф 1.7.1.1). |
| function **Lo**(X): Byte; |
| Функция **Lo** возвращает младший байт аргумента. X должна быть переменной типа **integer** или **word** (см. параграф 1.7.1.1). |
| function **Swap**(X:тип): тип; |
| Функция **Swap** меняет в аргументе младший и старший байт местами и возвращает полученный результат. Переменная X должна быть любого порядкового двухбайтного типа (см. параграф 1.7.1.1) |
| *Генератор случайных чисел* |
| function **Random** [( Range: Word)]: тип |
| Функция **Random** возвращает:  1) Случайное вещественное число из промежутка (0,1) (В случае, если Range отсутствует. При этом **тип** результата – **Real**)  2) Случайное натуральное число (или 0) из промежутка [0, Range) (В случае, если Range передается в качестве аргумента. При этом **тип** результата – **Word**) |
| procedure**Randomize**; |
| Процедура **Randomize** инициализирует генератор случайных чисел. После вызова этой процедуры **Random** будет возвращать другую последовательность случайных чисел. |
| *Границы и размеры типов данных* |
| function **High**(X):тип |
| Функция **High** возвращает максимальное значение аргумента. Если переданный параметр является строкой или массивом, то возвращаемый результат - максимальная длина строки или массива. Если же переданное значение имеет порядковый тип, то возвращаемое является верхней границей диапазона данного типа. |
| function **Low**(X):тип |
| Функция **Low** возвращает минимальное значение аргумента. Если переданный параметр является массивом, то возвращаемый результат - минимальный индекс массива. В случае передачи строки возвращаемое значение = 0. Если же переданное значение имеет порядковый тип, то возвращаемое является нижней границей диапазона данного типа. |
| function **SizeOf**(Аргумент):integer; |
| Функция **SizeOf** возвращает размер аргумента в байтах. Передаваемый в функцию параметр может быть как переменной, так и типом данных |
| *Параметры командной строки* |
| function **ParamCount**: Word; |
| Функция **ParamCount** возвращает количество аргументов, переданных программе в командной строке DOS при ее запуске. |
| function **ParamStr(Index:byte)**: String; |
| Функция **ParamStr** возвращает параметр командной строки с порядковым номером **Index**. Параметр с номером 0 — это полное имя самой программы. |

* Процедуры и функции модуля Crt

Модуль **Crt** содержит множество процедур и функций, поддерживающих работу с экраном и клавиатурой в обход стандартных потоков ввода-вывода.

|  |
| --- |
| *Обработка клавиатуры* |
| function **KeyPressed**: Boolean; |
| Функция **KeyPressed** возвращает **True**, если произошло нажатие клавиши. |
| function **ReadKey**: Char; |
| Функция **ReadKey** возвращает символ с клавиатуры, ожидая нажатия. |
| *Окна* |
| procedure **Window**(X1, Y1, X2, Y2: Byte); |
| Процедура **Window** устанавливает окно на экране. Весь дальнейший вывод и ввод информации происходит в его пределах. В качестве параметров передаются координаты левого верхнего и правого нижнего углов |
| procedure **ClrScr**; |
| Процедура **ClrScr** очищает текущее окно. |
| *Цвета* |
| ТП определяет в модуле Crt 15 констант для цветов текстовых режимов, плюс атрибут мигания, который для требуемого эффекта суммируется с цветом тона.  Константы тона и фона:  **Black=0, Blue=1, Green=2, Cyan=3, Red=4, Magenta=5, Brown=6, LightGray=7;**  Константы тона:  **DarkGray=8, LightBlue=9, LightGreen=10, LightCyan=11, LightRed=12, LightMagenta=13, Yellow=14, White=15;**  Атрибут мигания: **Blink=128** |
| procedure **TextBackground**(Color: Byte); |
| Процедура **TextBackground** устанавливает цвет фона |
| procedure **TextColor**(Color: Byte); |
| Процедура **TextColor** устанавливает цвет тона |
| procedure **NormVideo**; |
| Процедура **NormVideo** восстанавливает цвета тона и фона, определенные до запуска программы. |
| procedure **HighVideo**; |
| Процедура **HighVideo** устанавливает повышенную яркость символов. |
| procedure **LowVideo**; |
| Процедура **LowVideo** устанавливает пониженную яркость символов. |
| *Позиционирование курсора* |
| procedure **GotoXY**(X, Y: Byte); |
| Процедура **GotoXY** устанавливает курсор в позицию X,Y |
| function **WhereX**: Byte; |
| Функция **WhereX** возвращает текущую координату X курсора. |
| function **WhereY**: Byte; |
| Функция **WhereY** возвращает текущую координату Y курсора. |
| *Экранные строки* |
| procedure **ClrEol**; |
| Процедура **ClrEol** очищает все символы, начиная с позиции курсора до конца строки. |
| procedure **DelLine**; |
| Процедура **DelLine** удаляет строку, на которой находится курсор, все нижележащие поднимаются выше. |
| procedure **InsLine**; |
| Процедура **InsLine** вставляет пустую строку по позиции курсора, все нижележащие опускаются ниже. |
| *Звуки* |
| procedure **Sound**(Hz: Word); |
| Процедура **Sound** включает внутренний динамик с частотой звука **Hz** герц. |
| procedure **NoSound**; |
| Процедура **NoSound** выключает внутренний динамик |
| *Различные процедуры* |
| procedure **TextMode**(Mode: Integer); |
| Процедура **TextMode** устанавливает текстовый режим экрана.  В модуле Crt определен следующий набор констант режимов экрана:  **BW40=0**, черно-белый режим 40x25  **CO40=1**, цветной режим 40x25  **BW80=2**, черно-белый режим 80x25  **CO80=3**, цветной режим 80x25  **Mono=7**, монохромный режим 80x25 |
| procedure **AssignCrt**(var f: Text); |
| Процедура **AssignCrt** сопоставляет текстовый файл с экраном. Таким образом весь дальнейший ввод или вывод в текстовый файл будет перенаправляться на экран. |
| procedure **Delay**(MS: Word); |
| Процедура **Delay** приостанавливает работу программы на MS тиков. |

***Практическая работа!***

***Задание 1.****Ввести****X****и****Y****координаты курсора. Перевести курсор в позицию****X****и****Y****и вывести на экран желтым цветом (****Yellow****) слово "ЗДЕСЬ!".*

***Задание 2.****Всеми цветами вывести на экран фразу "Паскаль - ЧЕМПИОН!".*

***Задание 3.****Вывести на экран ваше любимое четверостишие. С промежутком примерно в секунду, стереть его построчно с экрана, начиная с первой строки.*

***Задание 4.****В центре экрана создать окно размером 6 строк на 50 столбцов. Затем решить предыдущую задачу.*

***Задание 5.****Вывести на экран фразу "Жизнь — это движение!". Передвинуть ее на 10 строк вниз с задержкой в 0.25 секунды на каждой строке. Затем аналогичным образом привести ее в первоначальное положение.*

***Задание 6.****Попытайтесь сыграть музыкальную гамму. Время звучания каждой ноты - полсекунды.*