Приложение В. Стандартни Процедури и Функции

В това приложение е дадено описанието на стандартните процедури и функции, подредени по азбучен ред.

function ABS(x)

Предназначение: Пресмятане на абсолютна стойност.

Резултат: От същия тип, от който е параметъра.

Действие: x е аритметичен израз от целочислен или реален тип. Резултатът е абсолютната стойност на израза, т.е. ако X е с положителен знак, то резултата е стойността на X, а ако е с отрицателен знак, то резултата е -X.

function ADDR(var x): pointer;

Предназначение: Получаване на адреса на променлива.

Резултат: От тип pointer.

Действие: x е променлива. Резултатът е указател, който сочи към x. Както и константата nil, резултатът на ADDR е съвместим с всички указатели.

procedure APPEND(var F [, FilName: string]);

Предназначение: Отваряне на файл за запис в края му.

Действие: Отваряне на съществуващ файл за запис с автоматично преминаване в края на файла. Ако файл с такова име не съществува, то той се създава. Ако не е указано име на файл, то файловата променлива трябва да бъде вече свързана с външен файл с помощта на REWRITE или OPEN. Ако файловата променлива е свързана с външен файл с помощта на RESET, получава се входно/изходна грешка 143 по време на изпълнение на програмата.

function ARCTAN(x): real;

Предназначение: Пресмятане на аркустангенс.

Действие: Пресмята аркустангенс от дадения в радиани параметър.

procedure BLOCKREAD(f; var Buffer; count: word [; var Result: word]);

Предназначение: Прочитане на определен брой байтове от файл.

Действие: BlockRead прочита count (или по-малко) байта от файла f в паметта на компютъра, започвайки от първия байт заеман от Buffer. Броят на действително прочетените байтове се връща в параметъра Result (ако е зададен). Ако не е зададен параметърът Result и са били прочетени по-малко байтове, от колкото е зададено в Count, ще бъде индицирана входно/изходна грешка.

procedure BLOCKWRITE(f; var Buffer; count: word [;var Result: word]);

Предназначение: Записване на определен брой байтове във файл.

Действие: BlockWrite записва count (или по-малко) байта от паметта на компютъра, започвайки от първия байт заеман от Buffer във файла f. Броят на действително записаните байтове се връща в параметъра Result (ако е зададен). Ако не е зададен параметърът Result и са били записани по-малко байтове, от колкото е зададено в Count, ще бъде индицирана входно/изходна грешка.

function CHR(x: ShortCard): char;

Предназначение: Получаване на символ от зададен ASCII код.

Действие: x е целочислен аритметичен израз в интервала [0,255]. Резултатът е символ с ASCII код x.

procedure CLOSE(var f [, TRUNCATE])

Предназначение: Затваряне на файл.

Действие: f е файлова променлива на типизиран или нетипизиран файл. Ако този файл е бил отворен с APPEND, RESET, REWRITE или OPEN, се извършват необходимите действия за затваряне на външния файл, с който е свързана файловата променлива, след което връзката между външния файл и файловата променлива се прекъсва. Ако е зададен TRUNCATE, то преди затваряне на файла неговата дължина се намалява до текущата позиция на файла.

function COS(x: real): real;

Предназначение: Пресмятане на косинус.

Действие: Пресмята косинус от дадения в радиани параметър.

procedure DEC(var x [; n]);

Предназначение: Намаляване стойността на променлива.

Действие: **x** е променлива от дискретен (ordinal) тип, а **n** е необезателен целочислен аритметичен израз. **x** се намалява с 1 или с **n**, ако **n** е зададено.

procedure DELETE(var s: string; ndx, count: ShortCard);

Предназначение: Премахване на подниз от низ.

Действие: Премахва (изтрива) count на брой символа от низа **s** започвайки от позиция **ndx**. Ако **ndx** е по-голямо от текущата дължината на низа, не се извършва нищо. Ако **count** + **ndx** е по-голямо от дължината на низа **s**, той просто се отрязва до позиция **ndx**.

procedure DISPOSE(p {, CaseValue});

Предназначение: Унищожаване на динамична променлива.

Действие: **p** е променлива от тип указател, сочеща към динамична променлива. DISPOSE унищожава тази динамична променлива, като връща паметта, заемана от нея за повторно използване.

Ограничение: Вариантните константи трябва да бъдат същите, каквито са били, когато е била заемана паметта чрез процедурата NEW.

function EOF[(var f)]: boolean;

Предназначение: Проверява ситуацията "край на файл".

Действие: **f** е файлова променлива. Ако не е указан параметър се подразбира стандартният входен файл. За текстов файл резултатът е TRUE, ако е прочетен последният символ от последния ред на текста или ако е достигнат символа край на файл (ASCII код 26) и FALSE в противен случай. За типизиран файл резултатът е TRUE, ако позиционирането във файла е след последния запис и FALSE в противен случай. За нетипизиран файл резултатът е TRUE, ако позиционирането във файла е след последния байт и FALSE в противен случай.

function EOLN[(var f: text)]: boolean;

Предназначение: Проверява ситуацията "край на ред".

Действие: **f** е файлова променлива от тип TEXT. Ако не е указан параметър, подразбира се стандартният входен файл. Резултатът е TRUE, ако е прочетен последният

символ от реда (позиционирането във файла е върху EOLN маркер) или ако EOF(f) е TRUE, а FALSE в противен случай.

procedure EXCL(var s: set; x)

Предназначение: Изключване на елемент от множество.

Действие: **s** е променлива от тип множество, а **x** е елемент от базовия тип на множеството. Тази процедура изключва зададения елемент **x** от множеството **s**, т.е. **EXCL(s, x)** е еквивалентно на **s:= s - [x]**;

procedure EXIT(ProcedureName);

Предназначение: Форсира излизане от процедура (функция).

Действие: Предизвиква принудително излизане от процедурата (функцията), чието име е зададено като параметър. При изпълнение на тази процедура се изпълняват всички завършващи части (операторите след етикета EXIT до края на съответната процедура) на процедурите (функциите), от които трябва да се излезе. Ако указаната чрез параметъра процедура (функция) не е активна, в момента на изпълнението на EXIT, индицира се грешка по време на изпълнение.

function EXP(x: real): real;

Предназначение: Пресмятане на експонента.

Действие: Връща експонента от x, т.е. EXP(x) = e^x.

function FILEPOS(var f): LongInt;

Предназначение: Получаване на текущата позиция във файл.

Действие: **f** е файлова променлива. Резултатът е номера на текущия запис. За нетипизирани файлове се дава броят на байтовете, намиращи се пред текущата позиция на файла. Ако текущата позиция е в началото на файла, то FilePos е равно на 0, а ако е в края на файла, т.е. EOF(f) = TRUE, то FILEPOS(f) = FILE-SIZE(f).

Ограничения: Не може да бъде използвана с текстов файл. Файла трябва да бъде отворен.

function FILESIZE(var f): LongInt;

Предназначение: Получаване на размера (дължината) на файл.

Действие: f е файлова променлива. Резултатът е от тип LongInt. Връща броя на компонентите (записите за типизираните и байтовете за нетипизираните файлове) на файла.

Ограничения: Не може да бъде използвана с текстов файл. Файлът трябва да бъде отворен.

procedure FILLCHAR(var Src; count: Natural; ch: char);

Предназначение: Запълване на непрекъсната област от паметта.

Действие: FILLCHAR запълва със символа ch count на брой байта от непрекъсната област на паметта, с начало Src.

procedure FILLWORD(var Src; count: Natural; W: word);

Предназначение: Запълване на непрекъсната област от паметта.

Действие: FILLWORD запълва с думата W count на брой думи от непрекъсната област на паметта, с начало Src.

function FRAC(x: real): real;

Предназначение: Получаване на дробната част на реално число.

Действие: x е аритметичен израз от реален тип. Резултатът е също от реален тип и представлява дробната част на реалния израз, даден като параметър.

procedure FREEMEMWORDS(p: pointer; WrdSz: Natural);

Предназначение: Освобождаване на заета памет.

Действие: p е променлива от тип указател, сочеща към динамична променлива с размер WrdSz думи. FREEMEMWORDS унищожава тази динамична променлива, като връща паметта, заемана от нея, за повторно използване. Стойността на WrdSz трябва да бъде точно толкова, колкото е била при използването на процедурата GETMEMWORDS, т.е. трябва да се освобождават толкова думи, колкото са били заети.

procedure GETDATE(var Year, Month, Day: word);

Предназначение: Получаване на текущата дата.

Действие: В променливите Year, Month, Day се получават съответно текущите година, месец и ден, като Year може да бъде в граници от 1980 до 2099, Month - от 1 до 12, а Day - от 0 до 31. Ако в променливата Day получите стойност 0, значи не е установена текущата дата в операционната система.

procedure GETMEMWORDS(var p: pointer; WrdSz: Natural);

Предназначение: Получаване на памет.

Действие: **p** е променлива от тип указател. WrdSz е обемът на областта от паметта (в думи), която ще се отдели. Адресът на началото й ще бъде записано в **p**.

Забележка: В никакъв случай да не се разчита, че областите от паметта, заети с две последователни извиквания на GETMEMWORDS, ще образуват непрекъснат блок.

procedure GETTIME(var Hour, Min, Sec, s100: word);

Предназначение: Получаване на текущото време.

Действие: В променливите Hour, Min, Sec и s100 се получават съответно текущите час, минута, секунда и стотна от секундата. Hour е в граници от 0 до 23, Min - от 0 до 59, Sec - от 0 до 59, а s100 - от 0 до 99.

procedure HALT[(ExitCode: Natural)];

Предназначение: Прекратяване на изпълнението на програмата.

Действие: ExitCode е необезателен параметър и, ако не е указан, се подразбира нула (0). HALT прекратява изпълнението на програмата, като връща на операционната система зададения си параметър. Тъй като в UniDOS не се поддържа ErrorLevel (както в MS-DOS), ефектът е еднакъв при всички параметри.

function HI(x: word): byte;

Предназначение: Получаване на старшия байт на дума.

Действие: x е параметър с големина една дума. Резултатът е от тип byte и представлява старшият байт на параметъра. Ако параметърът е с големина двойна дума, то той се отрязва (като се изхвърля старшата дума) и тогава се обработва.

function HIWORD(x: LongWord): word;

Предназначение: Получаване на старшата дума на двойна дума.

Действие: x е параметър с големина двойна дума. Резултатът е от тип word и представлява старшата дума на параметъра. Ако параметъра е с големина дума, то резултата на тази функция е 0.

procedure INC(var x [, n]);

Предназначение: Увеличаване на стойността на променлива.

Действие: **x** е променлива от дискретен (ordinal) тип, а **n** е необезателен целочислен аритметичен израз. **x** се увеличава с 1 или с **n**, ако **n** е зададено.

procedure INCL(var s: set; x);

Предназначение: Включване на елемент в множество.

Действие: s е променлива от тип множество, а x е елемент от базовия тип на множеството. Тази процедура включва зададения елемент в множеството s. Т.е. Incl(s, x) е еквивалентно на s:= s + [x];

procedure INSERT(const Src: string; var s: string; ndx: ShortCard);

Предназначение: Вмъкване на низ в друг низ.

Действие: Insert вмъква низа **Src** в низа **s** от позиция **ndx**.

function INT(x: real): real;

Предназначение: Получаване на цялата част на реално число.

Действие: x е аритметичен израз от реален тип. Резултатът е също от реален тип и представлява цялата част на реалния израз, даден като параметър.

function IORESULT: integer;

Предназначение: Получаване на кода на грешката на последната входно/изходна операция.

Действие: Автоматичната проверка на входно/изходните операции трябва да е изключена ({$I-}), за да може да се проверява кодът на грешката. В противен случай изпълнението на програмата ще бъде прекратено веднага след настъпване на входно/изходната грешка. IOresult връща 0, ако последната входно/изходна операция е успешно завършена.

function LENGTH(const s: string): ShortCard;

Предназначение: Получаване на дължината на низ (string).

Действие: Резултатът на функцията е дължината на дадения като параметър низ.

function LN(x: real): real;

Предназначение: Пресмятане на натурален логаритъм.

Действие: x е аритметичен израз от реален тип. Резултатът е натурален логаритъм от x, т.е. това е логаритъм от x при основа неперовото число. $\ln(x) = \log_e X$

function LO(x: word): byte;

Предназначение: Получаване на младшия байт на дума.

Действие: **x** е параметър с големина една дума. Резултатът е от тип byte и представлява младшият байт на параметъра. Ако параметърът е с големина двойна дума, то той се отрязва (като се изхвърля старшата дума) и тогава се обработва.

function LONGSWAP(w: LongWord): LongWord;

Предназначение: Размяна на байтовете в двойна дума.

Действие: Разменя байтовете в двойна дума. Местата си разменят първият и последният байт и вторият и предпоследният. Превръща двойна дума от вътрешно представяне младши ... старши в представяне старши ... младши байт и обратно.

function LOWORD(x: LongWord): word;

Предназначение: Получаване на младшата дума на двойна дума.

Действие: **x** е параметър с големина двойна дума. Резултатът е от тип word и представлява младшата дума на параметъра.

procedure MARK(var p: pointer);

Предназначение: Маркира динамичната памет.

Действие: Mark записва (маркира) състоянието на динамичната памет в променливата p, която по-късно може да бъде използвана като параметър за Release.

function MAX(type): type;

Предназначение: Получаване на максималната стойност на дискретен (ordinal) тип.

Действие: Резултатът на функцията е най-голямата (максималната) стойност от множеството от стойности, принадлежащи на дадения като параметър дискретен (ordinal) тип.

function MAXAVAIL: natural;

Предназначение: Получаване на размера на най-големия непрекъснат блок от свободната памет.

Действие: Резултатът на функцията е размера на най-големия непрекъснат блок свободна памет в думи.

function MEMAVAIL: natural;

Предназначение: Получаване на обема на свободна памет.

Действие: Връща обема на свободната памет в думи, т.е. това е сумата от обемите на всички непрекъснати блокове свободна памет.

function MIN(type): type;

Предназначение: Получаване на минималната стойност на дискретен (ordinal) тип.

Действие: Резултата на функцията е най-малката (минималната) стойност от множеството от стойности принадлежащи на дадения като параметър дискретен (ordinal) тип.

procedure MOVE(const Src; var Dest; Nbytes: Cardinal);

Предназначение: Премества непрекъсната област от паметта.

Действие: Премества Nbytes на брой байта от оперативната памет започвайки от област с начало Src в област с начало Dest.

procedure MOVEWORDS(const Src; var Dest; Nwords: Cardinal);

Предназначение: Премества непрекъсната област от паметта.

Действие: Премества Nwords на брой думи от оперативната памет от област с начало Src в област с начало Dest.

function MSGLIMIT: natural;

Действие: Връща широчината (в символи) на устройството, към което е насочен файлът MESSAGE, т.е. броя на символите на един ред от екрана.

procedure NEW(var p {, CaseValue});

Предназначение: Създаване на нова динамична променлива.

Действие: Създава нова динамична променлива и записва нейния адрес в **p**. Вариантните константи трябва да следват в ред по нарастване вложеността на вариантните части.

function ODD(x: LongInt): boolean;

Предназначение: Проверка за нечетност.

Действие: Връща TRUE, ако параметърът й е нечетно число.

procedure OPEN(var f [; Fname: string [, TEMPORARY]]);

Предназначение: Отваряне на файл за четене и писане.

Действие: Отваря външен файл с име в променливата Fname за четене и запис. Ако файл с такова име не съществува, то той се създава. Ако не е указано име на външния файл, то файловата променлива f трябва да е вече свързана с външен файл с помощта на друго използване на OPEN, в такъв случай се преминава в началото на файла. Ако е зададен параметър TEMPORARY, отваря се файл с уникално име. В такъв случай fname трябва да бъде променлива от тип string (с максимален брой елементи не по-малко от 80, т.е. поне string[80]), в която при извикване на OPEN трябва да се съдържа пътят, където ще се създаде файла, а след изпълнението на процедурата във fname се намира името на създадения файл.

function ORD(x): integer;

Предназначение: Получаване на поредния номер в множеството от стойности, принадлежащи на даден тип.

Действие: **x** е променлива от дискретен (ordinal) тип. Функцията ORD връща номера на **x** в множеството от стойности, дефинирани от типа, на който тя принадлежи.

function PARAMCOUNT: natural;

Предназначение: Получаване на броя на параметрите зададени на командната линия.

Действие: Връща броя на параметрите, зададени на командната линия, на операционната система при стартирането на програмата. Ако програмата е стартирана без параметри, функцията връща 0.

procedure PARAMSTR(n: natural; var s: string);

Предназначение: Получаване на стойността на параметър, зададен на командната линия.

Действие: Получаване на стойността на параметър с номер **n** от командната линия в низа **s**. Ако **n** = 0, получава се името на програмата (евентуално заедно с пътя, от където е стартирана).

function POS(const SubStr, Src: string): ShortCard;

Предназначение: Търсене на подниз в низ.

Действие: В низа Src се търси поднизът SubStr и, ако се намери, връща се номерът на позицията, от която се среща. Ако не се намери, връща се 0.

function PRED(x)

Предназначение: Получаване на предишната стойност от множеството стойности на даден дискретен (ordinal) тип.

Действие: **x** е израз от дискретен (ordinal) тип. Резултатът на функцията е от същия тип и е предната стойност от множеството стойности, принадлежащи на дискретния (ordinal) тип. Ако такава стойност няма, резултатът е непредсказуем.

function RANDOM[(x: Cardinal)]

Предназначение: Получаване на псевдо-случайно число.

Действие: Ако не е даден параметър, резултатът е реално число и 0 <= RANDOM < 1. Ако е зададен параметър, то резултатът е цяло число и 0 <= Random(**x**) < **x**.

procedure RANDOMIZE[(x)];

Предназначение: Инициализация на генератора на случайни числа.

Действие: Ако тази процедура не бъде изпълнена преди използване на функцията RANDOM, програмата ще получава (от RANDOM) при всяко стартиране една и съща псевдо-случайна редица от числа. В някои случаи това е необходимо. За да инициализирате псевдо-случайната последователност със случайно число, не трябва да се задава параметър на процедурата Randomize. Ако се зададе параметър (от цял или реален тип) то псевдо-случайната последователност от числа ще започва от това число и ще бъде една и съща при еднакви стойности на параметъра. Параметър 0 е еквивалентен на липса на параметър, т.е. randomize(0) = randomize.

procedure READ([f,] v1 [, v2, ... vn]);

Предназначение: Четене от типизиран файл.

Действие: f е файлова променлива и, ако не бъде зададена, се използва стандартният входен файл OUTPUT.

Ако f е текстов файл (или не е зададен), v1, ..., vn са променливи от тип integer, cardinal, longint, char или техни диапазони, както и real или string. От файла се четат символи и се преобразуват във вътрешно представяне на съответния тип.

Ако f е типизиран файл, то v1, ..., vn трябва да бъдат от типа на компонентите на този файл. За всяко v се прочитат необходимото количество байтове от файла и се записват в поредното v.

procedure READLN[([f ,] [v1, ..., vn])];

Предназначение: Прочитане с последващо пропускане на всички символи до края на реда.

Действие: Процедурата се прилага само за текстов файл и е еквивалентна на извикване на процедурата Read със същите параметри и след това пропускане на всички символи до началото на следващия ред.

Забележка: Като маркер за край на ред се възприема символа <cr> (ASCII код 13) или последователността от <cr> и <lf> (ASCII код 10). Самостоятелното срещане на <lf> в текстов файл не се обработва като край на ред.

procedure RELEASE(p: pointer);

Предназначение: Връщане на динамичната памет в положението в което е била при изпълнението на съответната процедура Mark.

Действие: Освобождава паметта, заета след изпълнението на процедурата Mark(p), с която е била маркирана динамичната памет.

procedure RESET(f [; Fname: string]);

Предназначение: Отваряне на файл за четене.

Действие: Отваря външен файл с име в променливата Fname за четене. Опитът за запис в този файл предизвиква входно/изходна грешка. Ако не е зададено име на файл, файловата променлива f трябва вече да е свързана с външен файл с помощта на OPEN или RESET. В този случай се предприемат необходимите действия, така че следващото четене да бъде от началото на файла.

procedure RETURN[([x])];

Предназначение: Прекратяване работата на текущата процедура.

Действие: Прекратява се работата на текущата процедура, като управлението се връща на извикалата я такава. Ако текущата процедура има завършваща част (EXIT:), RETURN е еквивалентно на GOTO EXIT, ако RETURN предшества етикета EXIT. Ако RETURN се използва вътре във функция, като параметър на RETURN трябва да се укаже резултатът, който се връща от функцията. Ако се укаже само (), то резултатът е такъв, какъвто е указан чрез предишно присвояване на стойност на идентификатора на функцията.

procedure REWRITE(var f [; Fname: string [; NEW | TEMPORARY]]);

Предназначение: Създаване на нов външен файл.

Действие: Създава нов външен файл с име, зададено в променливата Fname. В този файл е разрешен само запис. Опит за четене ще предизвика входно/изходна грешка. Ако външен файл с такова име вече съществува, то той се унищожава. Ако не е зададено име, то файловата променлива f трябва вече да е свързана с външен файл с помощта на OPEN, APPEND или REWRITE, в този случай се предприемат необходимите действия, така че следващият запис да бъде от началото на файла. Ако е необходимо файлът да бъде празен, след това изпълнение използвайте процедурата TRUNCATE. Ако е зададен параметър NEW и съществува стар файл, то той не се унищожава, а се връща грешка. Ако е зададен параметър TEMPORARY, файлът се отваря аналогично на начина, по който се отваря файл при процедурата OPEN с параметър TEMPORARY.

function ROUND(x: real): LongInt;

Предназначение: Превръщане на реално число в целочислено чрез закръгляване.

Действие: Превръща аритметичния израз **x**, който е от реален тип, в целочислен чрез закръгляване, т.е. резултатът е това цяло число, което е най-близко до **x**.

procedure SEEK(var f; NewPos: LongInt);

Предназначение: Позициониране във файл.

Действие: Файла се позиционира на позиция NewPos. Първият елемент на файла има номер 0. Ако стойността на NewPos е не по-малка от FileSize(f) и файлът е отворен чрез RESET, т.е. само за четене, възниква входно/изходна грешка. А ако е разрешен запис във файла, т.е. той е бил отворен чрез REWRITE, APPEND или OPEN, то файлът се увеличава (допълва), така че FileSize(f) = NewPos. Стойността на елементите с които се допълва файлът е непредсказуема.

Ограничение: Не може да се използва с текстови файлове.

function SIN(x: real): real;

Предназначение: Пресмятане на синус.

Действие: Пресмята синус от дадения в радиани параметър.

function SIZEOF(x {, CaseValue}): Cardinal;

Предназначение: Пресмята размера в байтове на параметъра си.

Действие: **x** може да бъде както променлива, така и тип. Ако е тип с вариантни записи (или променлива от такъв тип), то е възможно да се изброят вариантни константи по начина, по който се изброяват в NEW и DISPOSE. Като резултат се връща броя на байтове необходими за вътрешното представяне на параметъра.

function SQR(x)

Предназначение: Повдигане на квадрат.

Действие: **x** е аритметичен израз от цял или реален тип. Резултат на функцията е стойността на този израз повдигнат на квадрат. Типът на резултата съвпада с този на параметъра.

function SQRT(x: real): real;

Предназначение: Извличане на квадратен корен.

Действие: Резултатът на функцията е реален и е квадратен корен от стойността на параметъра.

function SUCC(x)

Предназначение: Получаване на следващата стойност от множеството стойности на даден дискретен (ordinal) тип.

Действие: **x** е от израз дискретен (ordinal) тип. Резултатът на функцията е от същия тип и е следващата стойност от множеството стойности, принадлежащи на дискретния (ordinal) тип. Ако такава стойност не съществува, резултатът е непредсказуем.

function SWAP(w: word): word;

Предназначение: Размяна на младшия и старшия байт в една дума.

Действие: Разменя младшия и старшия байт в една дума. Ако параметъра е с големина двойна дума той се отрязва (като се изхвърля старшата дума) и тогава се обработва.

function TRUNC(x: real): LongInt;

Предназначение: Превръщане на реално в цяло число чрез отрязване.

Действие: **x** е аритметичен израз от реален тип. Функцията превръща **r** в целочислен тип чрез отрязване на дробната част, т.е. trunc(r) = round(int(r));

procedure TRUNCATE(var f);

Предназначение: Отрязване на файл.

Действие: Процедурата намалява дължината на файла, отрязвайки го до текущата позиция.

function UPCASE(ch: char): char;

Предназначение: Преобразуване на буква в главна буква.

Действие: Ако стойността на параметъра на функцията е малка буква от латинската азбука ('a'..'z'), резултатът на функцията е съответната главна буква. В противен случай резултатът на функцията е същия символ, който е даден като параметър.

function WORDSIZEOF(x {,CaseValue}): Cardinal;

Предназначение: Пресмята размера в думи на параметъра си.

Действие: **x** може да бъде както променлива така и тип. Ако е тип с вариантни записи (или променлива от такъв тип), то е възможно да се изброят вариантни константи по начина, по който се изброяват в NEW и DISPOSE. Като резултат се връща броя на думите необходими за вътрешното представяне на параметъра.

procedure WRITE([f,] v1 [, v2, ... vn]);

Предназначение: Записване в типизиран файл.

Действие: f е файлова променлива и, ако такъв параметър не бъде зададен, използва се стандартният изходен файл.

Ако f е типизиран файл, то v1, ..., vn трябва да бъдат от типа на компонентите на този файл. Като всяко v се записва във вътрешно представяне във файла.

Ако f е текстов файл v1, ..., vn са променливи или изрази от тип INTEGER, CARDINAL, LONGINT, WORD, LONGWORD, CHAR или техни диапазони, както и от тип REAL, PACKED ARRAY OF CHAR или STRING. Всички параметри се преобразуват от вътрешно представяне в символно и тогава се записват в текстовия файл. Всеки параметър v1, ..., vn се записва в една от следните три форми: v, v:w, v:w:f. Тук с v е обозначен параметъра, с w и f се обозначават аритметични изрази от тип INTEGER, като третата форма е допустима само за реални параметри. Когато параметъра w е зададен, той указва дължина на полето (брой позиции), в което е необходимо да се запише дадения параметър. Ако размера на полето не е указан, подразбира се 0, което означава, че се използват толкова позиции, колкото е необходимо. Ако w е зададен и неговата абсолютна стойност е по-малка от големината на v в символно представяне, то стойността му се игнорира (т.е. използват се толкова позиции, колкото са необходими). Ако абсолютната стойност на w е по-голяма от броя на позициите, необходими за представяне на v, то: ако стойността на w е положителна, то пред символното представяне на стойността на v се записват толкова интервали, че заедно с представянето на v да заемат w позиции (т.е. прави се изравняване в дясно); а ако стойността на w е отрицателна, интервалите се записват след стойността на v (т.е. прави се изравняване вляво).

LONGINT и всички негови диапазони (INTEGER, CARDINAL, ... са също диапазони на LONGINT) се превръщат в символно десетично представяне (без водещи нули).

CHAR отпечатва се символът (евентуално и интервали, ако w < -1 или 1 < w).

CHAR ARRAY Едномерен пакетиран масив от символи (PACKED ARRAY [xx..xx] OF CHAR) се отпечатва така, както би се отпечатал низ, който съдържа всичките му елементи.

STRING отпечатват се последователно всички символи от низа.

LONGWORD и неговите диапазони BYTE и WORD се превръщат в символно шестнадесетично представяне (със водещи нули), т.е. числото се записва в шестнадесетична бройна система със съответно 8, 2 и 4 символа.

REAL Ако v е реален параметър, w е общата дължина на полето, в което трябва да бъде отпечатана стойността. Ако не е зададено f, то реалното число се отпечатва в експоненциален вид: **sd.ddddE±ee**, където **s** е знак минус при отрицателно число или интервал при положително; **d** е цифра от мантисата на реалното число, броят на цифрите е w-6; ± е знакът '+' или '-' в зависимост от знака на порядъка (експонентата); **ee** е десетично представяне на порядъка. Ако е зададен параметър f, то числото се отпечатва с фиксирана точка - f цифри за дробната част (след десетичната точка), останалото (w-f) за цялата част, точката и знака. Незначещите водещи нули пред цялата част и незначещите следващи нули след дробната част не се отпечатват (с изключение на първата 0 от дробната част). Ако зададения брой цифри за дробната част е по-малък от необходимия, числото се закръглява при отпечатването. Минималната стойност на w при отпечатване в експоненциален вид е 8 (символният вид е **sd.dE±ee**). Ако се зададе стойност, по-малка от 8, използва се 8. Максималната стойност за w е 15 (символният вид е **sd.ddddddddE ±ee**).

Примери:

```
writeln(124, ',', byte(124):4, ',', word(124): -6);
```

ще отпечата следното (със символа '·' ще отбелязваме интервал):

```
124,··7c,007c··
```

```
writeln('UniPascal': -16, '-', 'v1.60': 14);
UniPascal·····-·····v1.60
```

```
writeln(12.34:12,',',1.234:10:5,',',1.0:8:3,',',
1.512:8:0, ',', 1.455:8:2,';');
·1.23400E+01,····1.234·,···1.0··,·····2,····1.46;
```

procedure WRITELN[([f ,] [v1, ..., vn])];

Предназначение: Записване с последващо преминаване на нов ред.

Действие: Процедурата се прилага само за текстов файл. Тя е еквивалентна на извикване на процедурата Write със същите параметри и след това преминаване на следващия ред, т.е. записване на маркера за край на ред.

Забележка: Като маркер за край на ред се записват символите <cr> (ASCII код 13) и <lf> (ASCII код 10).