Сравнение языков Pascal и C/C++, быстрый переход с одного на другой

Автор: Третьяков Андрей

Константы и выражения, выделенные *курсивом*, взяты для примера; вместо них могут быть любые другие значения
Синим выделены расширения Delphi
Зелёным выделены необязательные части
Красным выделены места (источники ошибок), на которые надо обратить внимание

	Pascal	**C**	**C++**
Алфавит	Не чувствительный к регистру	Чувствительный к регистру	
Открывающая скобка	begin	{	
	begin не ставится в определении структур, классов, case, ассемблерных вставок		
Закрывающая скобка	end;	}	
; перед закрывающей скобкой	Необязательна	Обязательна	
Точка входа	begin *код ;* end.	int main(void) { *код;* return 0; }	
Однострочные комментарии	//	//	
Многострочные комментарии	{ *коммент.* }	/* *коммент.* */	

Переменные

Определение	*имя1*, *имяN*: *тип*;	*тип имя1*, *имяN*;	
Инициализация	только в Delphi; только глобал.; по одной; только константами: *имя*: *тип* = *иниц*;	*тип имя1* = *иниц1*, *имяK*, *имяN* = *иницN*;	
Расположение	В блоке var (до begin)	В нач. блока { } до первой строчки кода	До любого оператора; в заголовке for
	формальные параметры в заголовке процедур и функций		
Примеры ввода-вывода:	var a: Integer; f: Real; s: String;	int a; double d; char s[256];	
Ввод	Read(a, f, s);	scanf("%d %lf %s", &a, &d, s);	cin >> a >> d >> s;
Вывод	WriteLn(a, ' ', f:5:3, s);	printf("%d %5.3lf %s\n", a, d, s);	cout << a << " " << d << " " << s << endl;

Типы данных

Целочисленные		
1 байт, со знаком	Shortint	char
1 байт, без знака	Byte	unsigned char
2 байта, со знаком	Smallint	short int
2 байта, без знака	Word	unsigned short int
4 или 2 байта, со знаком	Integer	int
4 байта, без знака	Cardinal	unsigned int
4 или 8 байт, со знаком	Longint	long int
4 или 8 байт, без знака	Longword	unsigned long int
8 байт, со знаком	Int64	long long int
Вещественные		
4 байта, короткий, одинарная то	Single	float
6 байт	Real	-

8 байт, длинный, двойная точно	Double	double	
10 байт, расширенный	Extended	long double	
Другие			
Булев (логический)	Boolean	int	bool
Символьный	Char	char	
Строковый	String	*полного аналога нет*	
	String в Delphi - это либо ShortString, либо AnsiString, в зависимости от настроек и ключей компиляции		
Статическая строка	String, ShortString	char [256]	
Динамическая строка	String, AnsiString	char *	std::string
Указатель на символ/строку (C)	PChar	char *	
Перечислимый	type *RGB* = (*red*, *green*, *blue*);	enum *RGB* {*red*, *green*, *blue*};	
	Значения типа используются как именованные константы		
Диапазон	*10..18*	-	
Множество	set of *тип*	-	
Примеры множеств Размер типа - не более 256 В	set of *Char*;	-	
	set of *0..255*;	-	
	set of '*0*'..'*9*';	-	
Массив	*A*: array[*1..100*] of *Integer*;	*int A*[*100*];	
Первый элемент массива	*A*[***1***]	*A*[***0***]	
Последний элемент массива	*A*[***100***]	*A*[***99***]	
Адрес массива	*@A*	***A***	
Инициализация массива	*A*: array[*1..3*] of *Integer* = (*2*, *4*, *6*);	*int A*[*3*] = {*2*, *4*, *6*};	
Многомерный массив	*A*: array[*1..10*, *1..20*, *1..30*] of *Integer*;	*int A*[*10*][*20*][*30*];	
Доступ к элементу мн.массива	*A*[*i*, *j*, *h*]	*A*[*i*][*j*][*h*]	
Инициализация мн.массива	*A*: array[*1..2*, *1..3*] of *Integer* = ((*1*, *3*, *5*), (*2*, *4*, *6*));	*int A*[*2*][*3*] = {{*1*, *3*, *5*}, {*2*, *4*, *6*}}; *int A*[][*3*] = {{*1*}, {*2*, *4*, *6*}};	
Типизированный указатель	^*тип*	*тип* *	
Нетипизированный указатель	Pointer	void *	
Взятие адреса, устан.указателя	var *a*: *Integer*; *p*: ^*Integer*; *p* := *@a*;	*int* **p*, *a*; *p* = *&a*;	
Разыменование указателя (тип.)	*p*^ := *10*; // *a* = *10*	**p* = *10*; // *a* == *10*	
Пользовательский тип	type *новый_тип* = *сущ_выр*; *новый_типN* = *сущ_вырN*;	typedef *сущ_выр новый_тип*;	
Запись/структура	*имя* = record *поле1, полеK*: *тип1*; ... *полеN*: *типM*; end;	struct *имя* { *тип1 поле1, полеK*; ... *типM полеN*; };	
Инициализация записи/ структуры	type *Point* = record *x*: *Integer*; *y*: *Double*; end; var *p*: *Point* = (*x*: *3*; *y*: *6.7*);	struct *Point* { *int x*; *double y*; }; struct *Point p* = {*3*, *6.7*};	

Константы

Неименованные		
Целочисленные		
Десятичные	*123*	*123* //int *123*u, *123*U //unsigned int *123*l, *123*L //long *123*UL, *123*lu //unsigned long
Восьмеричные	-	***0****11* //*9* +суфф. u,l
Шестнадцатеричные	$*3B*, $*3b* //*59*	0x*3B*, 0X*3b* //*59* +суфф. u,l

Вещественные			
Обычная форма	*3.5*	*3.5* //double *3.5*f, *3.5*F //float *3.5*l, *3.5*L //long double *.5 == 0.5*; *5. == 5.0*	
Экспоненциальная форма	*3.5*E*-2*, *3.5*e*-2* //*0.035* *3.5*E*2*, *3.5*e*2* //*350* *-1.6*e*-19* //*-1.6* * 10^(*-19*)		
Другие			
Булевы (логические)	True False	*1* 0	true false
Символьные	'*A*' *#10* *#32* // ' '	'*A*' '**0***12*' // '*n*' '**x***20*' // ' '	
		sizeof '*A*' == 4	sizeof '*A*' == 1
Строковые	'Qwerty'	"Qwerty"	
Множество	[*3*, *5*, *8*]	-	
Пустой указатель	nil	NULL	NULL // == 0
Именованные			
Нетипизированные	const *имя1* = *знач1*; ... *имяN* = *значN*;	#define *имя знач* *точка с запятой* ***НЕ*** *ставится!*	
Типизированные	const *имя1* : *тип1* = *знач1*; *имя2* : *тип1* = *знач2*; ...	const int *имя1* = *знач1*; const *double имя2* = *знач2*;	

Операции (operators)

Вызов функции (процедуры)	()	() *ранг 1*
Индексация эл-тов массива	[]	[] *1*
Доступ к полю структуры	.	. *1*
--||-- через указатель	.	-> *1*
Логическое отрицание	not *ранг 1*	! *2*
Побитовое инвертирование	not *1*	~ *2*
Унарный +	+ *1*	+ *2*
Унарный -	- *1*	- *2*
Инкремент	Inc(·) *функция*	++ *2*
Декремент	Dec(·) *функция*	-- *2*
Взятие адреса	@ *1*	& *2*
Разыменование	^	* *2*
Приведение типов	*тип*(*выраж*)	(*тип*)*выраж* *2* (*тип*)*выраж* *тип*(*выраж*)
Размер переменной/типа	SizeOf(*пер*) SizeOf(*тип*)	sizeof *пер*, sizeof(*пер*) *2* sizeof(*тип*)
Умножение	* *2*	* *3*
Деление вещественное	/ *2*	/ *3*
Деление целочисленное	**div** *2*	/ *3* (если **оба** операнда целые)
Остаток от деления	mod *2*	% *3*
Сложение	+ *3*	+ *4*
Вычитание	- *3*	- *4*
Побитовый сдвиг влево	shl *2*	<< *5*
Побитовый сдвиг вправо	shr *2*	>> *5*
Меньше/больше(или равно)	< <= >= > *4*	< <= >= > *6*
Равно	= *4*	== *7*
Не равно	<> *4*	!= *7*
Побитовое И	and *2*	& *8*
Побитовое исключ. ИЛИ	xor *3*	^ *9*
Побитовое ИЛИ	or *3*	| *10*

Логическое И	and *2*	&& *11*
Логическое ИЛИ	or *3*	|| *12*
Условная		? : *13*
Присваивание	*:= не операция*	= *14*
Составное присваивание		*= /= %= += -= *14* &= ^= |= <<= >>=
Операция запятая		, *15*
Приоритет сложных условий	if (a > b) **or** (x > y) then …	if (a > b || x > y) …

Операторы (statements)

Пустой	;	;
Составной	begin *оп1* ; … *опN*; end;	{ *оп1* ; … *опN*; }
Условный	if *условие* then *оператор* ;	if (*условие*) *оператор* ;
Условный с else	if *условие* then *оператор1* else *оператор2* ;	if (*условие*) *оператор1* ; //*кроме: после* } else *оператор2* ;
Переключатель (оп.выбора)	case *выр* of //*целое,Char,Bool.* *конст1* : *опер1* ; *констK*, *констN* : *операN*; else *операD*; end;	switch (*выр*) { //*целое, char* case *конст1* : *опер1* ; ... *операM*; break; case *констK* : case *констN* : *операN*; break; default: *операD* ; }
Цикл с предусловием	while *условие* do *оператор* ;	while (*условие*) *оператор* ;
Цикл с постусловием	repeat *опратор1* ; ... *операторN*; until *условие_**остановки*** ;	do *оператор* ; while (*условие_**работы***)**;**
Цикл for	for *i* := *выр1* downto *выр2* do *оператор* ;	for (*иниц* **;** *условие* **;** *инкрем*) *оператор* ;
Примеры for	for *i* := *1* to *10* do *оператор* ;	for (*i* = 1; *i* <= *10*; *i*++) *оператор* ;
	for *i* := *10* downto *1* do *оператор* ;	for (*i* = *10*; *i* >= 1; *i*--) *оператор* ;
условие приводится к типу…	булеву (логическому)	целому
		скобки *в условии обязательны*
Выход из цикла	Break;	break;
Переход на след. итерацию	Continue;	continue;
Безусловный переход	Label *lb1* ; … goto *lb1* ; ... *lb1* : *оператор* ;	goto *lb1* ; ... *lb1* : *оператор* ;
Возвращение из функции	**Exit**;	return *выр* ;
Возвращение из процедуры	**Exit**;	return;
Досрочное завершение прогр.	Halt;	**exit**(*код_возврата*);

Подпрограммы

Функции	function ***имя***(*пар1*, *парK*: *тип1*; *парN*: *типN*): *тип_возвр*; begin *код*; ***имя*** := *выр*; *или* Result := *выр*; end;	*тип_возвр* ***имя***(*тип1 пар1*, ***тип1*** *парK*, *типN парN*) { *код*; return *выр*; }	
Процедуры	procedure ***имя***(*пар1*, *парK*: *тип1*; *парN*: *типN*); begin *код*; end;	void ***имя***(*тип1 пар1*, ***тип1*** *парK*, *типN парN*) { *код*; }	
Функции/проц. без параметров	function ***имя***: *тип*; begin ... end;	*тип* ***имя***(void) { ... }	*тип* ***имя***() { ... }
"*тип* ***имя***()" эквивалентно…		*тип* ***имя***(...)	*тип* ***имя***(void)
Вложенные подпрограммы	Допускаются	Не допускаются	
Вызов функции	*res* := ***имя***(*a, b + c, 3, 4*);	*res* = ***имя***(*a, b + c, 3, 4*);	
Вызов процедуры	***имя***(*a, b + c, 3, 4*);	***имя***(*a, b + c, 3, 4*);	
Вызов функции без параметров	*res* := ***имя***;	*res* = ***имя***();	
Прототип функции/процедуры	function ***имя***(*пар1*: *тип1*; *парN*: *типN*): *тип_возвр*; **forward;**	*тип_возвр* ***имя***(*тип1 пар1*, *типN парN*);	
		скобки *после* ***имени*** *обязательны*	
Адрес функции	@***имя***	***имя***	
		res = ***имя***; //вернёт адрес функции, а не её результат; //функция не будет вызвана	