Convertion et instuctions de Turbo Pascal vers le langage C

Pour des programmes, du code, des cours en Turbo Pascal : http://turbo-pascal.developpez.com/
Pour des programmes, du code, des cours en C et C++ : http://www.developpez.com/c/

Sommaire :

Langage	Tests	Mémoire	Accès aux fichiers
Types	Fonctions mathématiques	Chaîne de caractères	

Langage (retour au Sommaire) :

Turbo Pascal	C++	Explication
uses NomUnite;	**#include <**NomUnite**>**	Utilise une unité
{ Commentaire intégré **}** **(*** Long commentaire ***)**	**/*** Commentaire intégré ***/**	Permet d'ajouter un commentaire intégré au code
	// Commentaire jusqu'à la fin de la ligne	Permet d'ajouter un commentaire en fin de ligne, ou de désactiver un ligne de code

caseVAR**of**
1: ... ;
2: ...;
else ...;
end;
	switch (VAR) {
	case 1: ...; **break;**
	case 2: ...; **break;**
	default: ...; **break;**
	}
		Tableau des différents cas pour une variable :
		1: Si VAR égal 1 alors ...
		2: Si VAR égal 2 alors ...
		Sinon: ...

for i := 1 to 9 do ...	for (i=1; i<= 9; i++) ...	Boucle de 1 jusqu'à 9 par ajout de 1 à chaque étape
for i := 9 downto 1 do ...	for (i=9; 1 <=i; i--) ...	Boucle de 9 à 1 par soustraction de 1 à chaque étape
i := 1; *repeat* *inc (i,2);* *until 9<=i;*	for (i=1; i<= 9; i += 2)	Boucle de 1 à 9 par ajout de 2 à chaque étape
sizeof (a);	sizeof (a);	Renvoie la taille d'une variable

if COND then
begin
instructionA1;
instructionA2;
end else begin
instructionB1;
instructionB2;
end;
	COND **?** instructionA1; instructionA2; : instructionB1; instructionB2;	Si COND est vrai alors exécute les instructions A1, A2, ...; sinon exécute les instructions B1, B2, ...

Types (retour au Sommaire) :

Turbo Pascal	C++	Explication
var char a;	**unsigned char** a;	Déclare le caractère "a" (valeur entre 0 et 255)
var byte a;	**unsigned char** a;	Déclare l'octet "a" (valeur entre 0 et 255)
var shortint a;	**char** a;	Déclare l'octet signé "a" (valeur entre -127 et 128)
var word a;	**unsigned int** a;	Déclare le mot "a" (valeur entre 0 et 65535)
var integer a;	**int** a;	Déclare le mot signé "a" (valeur entre -32 768 et 32 767)
X	**unsigned long** a;	Déclare le double-mot "a" (valeur entre 0 et "4 294 967 295") Remarque: N'existe pas en Turbo Pascal!
var long a;	**long** a;	Déclare le double-mot signé "a" (valeur entre "-2 147 483 648" et "2 147 483 647")
var single a;	**float** a;	Déclare la variable "a" de type virgule flotante sur 32 bits (valeur entre "-3,4 x10^38" et "3,4 x10^-38", précision sur 7 chiffres)
var double a;	**double** a;	Déclare la variable "a" de type virgule flotante sur 64 bits (valeur entre "1,7 x10^308" et "3,4 x10^-308", précision sur 15 chiffres)
var extended a;	**long double** a;	Déclare la variable "a" de type virgule flotante sur 80 bits (valeur entre "3,4 x10^4932" et "1,1 x10^-4932", précision sur 19 chiffres)
const a = 10;	**#DEFINE** a 10	Déclare la constante "a" stockant le nombre 10
const a = "abc";	**#DEFINE** a "abc"	Déclare la constante "a" stockant la chaîne "abc"
a = **ord(** b);	a = **(char)** b;	Copie la valeur ASCII du caractère "b" dans "a"
a = **word(** b);	a = **(unsigned int)** b;	Copie la valeur "b" dans le format mot non-signé dans "a"
p := **addr(** a);	p = **&** a;	Copie l'adresse de "a" dans le pointeur "p"
a = **ptr(**p); ???	a = * p ;	Copie l'adresse du pointeur "p" dans "a"
a := **$** F54B;	a = **0x** F54B;	Stocke le nombre hexadécimal F545B h dans "a"

Tests (retour au Sommaire) :

Turbo Pascal	C++	Explication
if CAS1**and** CAS2 then	if (CAS1**&&** CAS2)	si le CAS1 est vrai ET le CAS2 est vrai
if CAS1**or** CAS2 then	if (CAS1 **||** CAS2)	si le CAS1 est vrai OU le CAS2 est

	(double-barre verticale, touche ALT+6)	vrai
if CAS1=**FALSE** then	if (**!**CAS1)	si le CAS2 est faux
if a**=** b then	if (a**==** b) *(deux "égal")*	si a égal b
if a**<** b then	if (a**<** b)	si a est inférieur à b
if a**<=** b then	if (a**<=** b)	si a est inférieur ou égal à b

Fonctions mathématiques (retour au Sommaire) :

Turbo Pascal	C++	Explication
inc (a);	a**++**;	Ajoute un à la valeur "a"
inc (a**,** **b**);	a**+= b** ;	Ajoute la valeur "b" à "a"
dec (a);	a**--**;	Soustrait un à "a"
dec (a**,** **b**);	a**-= b** ;	Soustrait la valeur "b" à "a"
a := b ***c**;	a = b ***c**;	Multiplication
a := a ***b**;	a***=**b ;	Ecrit "a" multiplié par "b" dans "a"
a := b **div**c;	a = b **/**c;	Division entière
a := a **div** b;	a**/=**b ;	Divise "a" par "b"
a := b **mod**c;	a = b **%**c;	Ecrit le reste de la division de "b" par "c" dans "a"
a := a **mod** b;	a**%=**b ;	Ecrit le reste de la division de "a" par "b" dans "a"
a := b **and**c;	a = b **&**c;	"et"
a := a **and**b;	a **&=** b;	Ecrit "a" AND "b" dans "a"
a := b **or**c;	a = b **|** c; *(trait vertical, touche ALT+6)*	"ou"
a := a **or** b;	a**|=**b ;	Ecrit "a" OR "b" dans "a"
a := b **xor** c;	a := b ^ c;	"ou exclusif"
a := a **xor** b;	a**^=**b ;	Ecrit "a" XOR "b" dans "a"
a := b **shr** n;	a = b >> n;	Décalage binaire vers la gauche (division par 2^ n) de "b"
a := a **shr** n;	a**>>=** n;	Ecrit le décalage binaire de "n" bits vers la gauche de "a" dans "a"
a := b **shl** n;	a = b << n;	Décalage binaire vers la droite (multiplication par 2^ n) de "b"
a := a **shl** n;	a**<<=** n;	Ecrit le décalage binaire de "n" bits vers la droite "a" dans "a"
a := **not** (b);	a := **~** b; *(touche ALT+2)*	Ecrit le complément de "b" dans "a" (inverse tous les bits).

Mémoire (retour au Sommaire) :

En C : Pour afficher du texte dans un application console, il faut utiliser l'unité stdlib.h.
Ajoutez la ligne "**#include <stdlib.h>**" au début de votre programme.

Turbo Pascal	C++	Explication
getmem (p, taille);	p = malloc (taille);	Alloue une espace mémoire de "taille" octets et stocke le pointeur dans "p"
dispose (p);	free(p);	Libère la mémoire allouée qui était pointée par "p"
move (src,dst ,taille);	memcpy (dst, src,taille);	Copie les "taille" octets de " src" dans "dst"

Chaînes de caractères (retour au Sommaire) :

En C : Pour afficher du texte dans un application console, il faut utiliser l'unité stdio.h.
Ajoutez la ligne "**#include <stdio.h>**" au début de votre programme.

Turbo Pascal	**C++**	**Explication**
var a: **string** [10];	**char** a [10+1];	Déclare la chaîne "a" ayant une longueur maximale de 10 caractères. Remarque: En C, il faut faire +1 pour pouvoir écrire le zéro terminal.
a := length(txt);	a = strlen (txt);	Ecrit la longueur du texte "txt" dans le nombre "a"
a[1] := ' g';	a[0] = ' g';	Ecrit le caractère 'g' dans le premier octet de la chaîne 'a'
a := ' Salut';	strcpy (a, "Salut");	Ecrit la chaîne "Salut" sans la chaîne "a".
a := b ;	strcpy (a, b);	Ecrit la chaîne "b" dans la chaîne "a"
Write ("Allo ?");	**printf (**"Allo ?");	Ecrit le texte "Allo ?"
Writeln ("Bonjour ! ");	**printf (**"Bonjour !**\n** ");	Ecrit le texte "Bonjour !", puis revient à la ligne
Writeln ("Nombre = **",**);	**printf (**"Nombre = **%d\n** ,**nbr**);	Ecrit le texte "Nombre = nbr" incluant le nombre "nbr", puis revient à la ligne
Writeln ("a=**",** **A**," b=", **B** , " et c=", **C**);	**printf (**"a=**%d b=%d** c=**%d\n** ", **A, B,C**);	Ecrit le texte "a=A b=B c=C" incluant les trois nombre "A", "B" et "C", puis revient à la ligne
Writeln ("Texte = **", txt**);	**printf** ("Texte = **%s\n** ",**txt**);	Ecrit le texte "Texte = txt" incluant le texte "txt", puis revient à la ligne

Caractères spéciaux en C++ :

\a	Emet un beep sonore
\b	Retour en arrière d'un caractère
\f	Saut de page
\n	Retour à la ligne
\r	Revient au début de la ligne
\t	Tabulation
\v	Descend d'une ligne
****	Ecrit le caractère>\<
\"	Ecrit le caractère >"<
\'	Ecrit le caractère >'<
\?	Ecrit le caractère >?<

/000	Ecrit le caractère ASCII en code octale (sur base 8)
/x000	Ecrit le caractère ASCII en code hexadécimal (sur base 16)

Code pour afficher des nombres avec printf en C :

%c	Un caractère
%s	Chaîne de caractère
%d	Entier
%ld	Entier long
%f	Nombre réel **float**
%lf	Nombre réel **double**
%Lf	Nombre réel **long double**
%p	Pointeur

Accès aux fichiers (retour au Sommaire) :

En C++ : Pour afficher du texte dans un application console, il faut utiliser l'unité **stdio.h**. Ajoutez la ligne "**#include <stdio.h>**" au début de votre programme. En bonus : "**#include <errno.h>**" permet d'accéder à la dernière erreur.

Turbo Pascal	C++	Explication
const nomfich = "test.txt"; Var F: File; Var Erreur: Integer; {$I-} Assign (F, "test.txt"); FileMode := **0**; Reset (F, 1); {$I+} Erreur := **IOResult**; If Erreur<>0 then Writeln ('Erreur d'ouverture du fichier "',nom,'" : erreur ',Erreur);	const char *nomfich = "test.txt"; FILE *F; F = fopen (nomfich,**"r"**); if (F == NULL) printf ("Erreur d'ouverture du fichier '%s' : erreur %i\n",nom,**errno**);	Ouvre le fichier "test.txt" en lecture seule avec gestion des erreurs. Note : Changer "FileMode := 0;" en "FileMode := 1;" (Pascal) ou "r" en "w" pour l'ouvrir en écriture seule. Ou bien "FileMode := 2;" (Pascal) et "rw" (C) pour la lecture+écriture. Note: **IOResult** (Pascal) / **errno** (C) = code de la dernière erreur d'entrée/sortie.
Append (F);	append (F);	Se positionne à la fin d'un fichier ouvert pour pouvoir rajouter des octets.
a := Read (F);	a = fgetc (F);	Lit un caractère
Write (F, a);	fputc (F, a);	Ecrit le caractère "a"
var Ligne: String; Ligne := ReadLn(F);	#define LongMax 80 char Ligne[LongMax+1]; fgets (F, LongMax+1, Ligne);	Lit une ligne de texte. Le C ajoute une limite : LongMax caractères au maximum (+1 pour le zéro terminal)
const txt = "test"; Writeln (F,txt);	const char *txt = "test"; fputs (F, txt);	Ecrit une ligne de texte.
BlockRead (F, Tampon, taille);	fread (Tampon, taille,1,F);	Lit taille octets.

BlockWrite (F, Tampon, taille);	fwrite (Tampon, taille,1,F);	Ecrit taille octets.
Seek (F, pos);	fseek (F, pos);	Positionne le curseur du fichier de handle "F" à la position "pos"
a := FilePos(F);	a := ftell (F);	Lit la position du fichier de handle "F" et la stocke dans "a"
Close (F);	fclose (F);	Ferme le fichier de handle "F"

Retour au Sommaire