Conversion Table between Maplesoft, Mathematica, C/C++ and Fortran

(March 2014)

Intrinsic functions

Functions	Maplesoft	Mathematica	C/C++	Fortran
Absolute value	abs(x)	Abs[x]	abs(x)	abs(x)
Square root	sqrt(x)	Sqrt[x]	sqrt(x)	sqrt(x)
Sine	sin(x)	Sin[x]	sin(x)	sin(x)
Cosine	cos(x)	Cos[x]	cos(x)	cos(x)
Tangent	tan(x)	Tan[x]	tan(x)	tan(x)
Cosecant	csc(x)	Csc[x]	1/sin(x)	1/sin(x)
Secant	sec(x)	Sec[x]	1/cos(x)	1/cos(x)
Cotangent	cot(x)	Cot[x]	1/tan(x)	1/tan(x)
Hyperbolic Sine	sinh(x)	Sinh[x]	sinh(x)	sinh(x)
Hyperbolic Cosine	cosh(x)	Cosh[x]	cosh(x)	cosh(x)
Hyperbolic Tangent	tanh(x)	Tanh[x]	tanh(x)	tanh(x)
Arcsine	arcsin(x)	ArcSin[x]	asin(x)	asin(x)
Arccosine	arccos(x)	ArcCos[x]	acos(x)	acos(x)
Arctangent	arctan(x)	ArcTan[x]	atan(x)	atan(x)
Arc hyperbolic sine	arcsinh(x)	ArcSinh[x]	asinh(x)	asinh(x)
Arc hyperbolic cosine	arccosh(x)	ArcCosh[x]	acosh(x)	acosh(x)
Arc hyperbolic tangent	arctanh(x)	ArcTanh[x]	atanh(x)	atanh(x)
Common logarithm	log10(x) or log[10](x)	Log10[x]	log10(x)	log10(x)
Natural logarithm	ln(x)	Log[x]	log(x)	log(x)
Power (e.g of x^2)	x^2	x^2	pow(x,2)	x**2
Real part	re(z)	Re[z]	creal(z)	real(z)
Imaginary part	im(z)	Im[z]	cimag(z)	aimag(z)
Complex conjugate	conjugate(z)	Conjugate[z]	conj(z)	conjg(z)
Floor (The largest integer not greater than x)	floor(x)	Floor[x]	floor(x)	floor(x)
Ceiling (The smallest integer not greater than x)	ceil(x)	Ceiling[x]	ceiling(x)	ceiling(x)
Maximum value	max(a,b,…)	Max[a,b,…]	max(a,b,…)	max(a,b,…)
Minimum value	min(a,b,…)	Min[a,b,…]	min(a,b,…)	min(a,b,…)

Sign	sign(x)	Sign[x]	?	sign(x,y)
Random number generator	rand(x)	RandomReal[] RandomInteger[] RandomComplex[]	rand(x)	rand(x) irand(x)* drand(x)**

*For integer between 0 to 2147483647 **For double precision between 0 to 1

Operators

	Maplesoft	Mathematica	C/C++	Fortran
Addition	+	+	+	+
Subtraction	-	-	-	-
Multiplication	*	*	*	*
Division	/	/	/	/
Exponentiation	^	^	pow(a,b)	**
Modulo (The remainder of m divided by n)	[m mod n]	Mod[m,n]	m%n	mod(m,n)
Concatenation	cat("a","b") or ||	<>	+	//
Less than	<	<	<	< or .lt.
Less than or equal to	<=	<=	<=	<= or .le.
Equal to	=	==	==	== or .eq.
Not equal to	<>	!=	!=	/= or .ne.
Greater than	>	>	>	> or .gt.
Greater than or equal to	>=	>=	>=	>= or .ge.
Logical negation	not	!	! (or ~)*	.NOT.
Logical conjunction	and	&&	&& (or &)*	.AND.
Logical inclusive disjunction	or	||	|| (or |)*	.OR.
Logical exclusive disjunction	xor	Xor[p,q,…]	^ *	.XOR.
Logical equivalence	equivalent(a,b)	⇔		.EQV.
Logical nonequivalence				.NEQV.

* These operators are for bitwise expressions.

Command usage

	Fortran	C/C++

Conditional statement 1	if(a>=0.0) a=5.0	if(a>=0.0) a=5.0;

Conditional statement 2	if(a>0.0) then
	  a=5.0
	  b=a+1.0
	else if(a<0.0) then
	  a=-5.0
	  b=a-1.0
	else
	  a=0.0
	  b=0.0
	endif
		if(a>0.0)
		  {a=5.0;
		  b=a+1.0;}
		else if
		  {a=-5.0;
		  b=a-1.0;}
		else
		  {a=0.0;
		  b=0.0;}

Conditional statement 4
Note: For C, you may have to use “printf” instead of “cout.”
	integer n
	select case(n)
	  case(1)
	  print *,”first”
	  case(2)
	  print *,”second”
	  case(3)
	  print *,”third”
	  case default
	  print *, “Come again!”
	end select
		int n
		switch(n){
		  case 1:
		  cout <<”first” << endl;
		  break;
		  case 2:
		  cout <<”second” << endl;
		  break;
		  case 3:
		  cout <<”third” << endl;
		  break;
		  default:
		  cout <<”Come again!” << endl;
		  break;
		}

Ternary operator
Note that this is equivalent with the following algorithm:
If(n<5)
  a=b
else
  a=c
	a = merge(b,c,n<5)	a = (n<5) ? b : c

Looping structure 1	do i=1,100
	  a = a + 2.0
	  print *,a
	end do
		for(i=1;i<=100;i++)
		  {a += 2.0;
		  cout << a << endl;
		  }

Looping structure 2
Note: While(1) indicates infinite loop until it breaks. When while(0), it will go out of the loop.
	n = 10
	do while(1)
	print *,n
	n = n – 1
	if(n==0) exit
	end do
		int n = 10;
		while (1) {
		cout << n << endl;
		n--;
		if (n == 0)
		break;
		}

Looping structure 3 This executes at last once then it is checked by the condition after that.		int n; cin >> n; do {cout << n << endl; n += 2; }while(n<20);

Quick reference of files I/O

	Fortran	C	C++

Reading a file	open(unit=7,file="in.d")
	read(7,*) var
	close(7)
		FILE *fi;
		fi = fopen("in.d","r");
		fscanf(fi,"%f",&var);
		fclose(fi);
			#include<fstream>
			ifstream fin("in.d");
			fin >> var;

Writing a file	open(unit=8,file="out.d")
	write(8,*) var
	close(8)
		FILE *fp;
		fp = fopen("out.d","w");
		fprintf(fp,"%f",var);
		fclose(fp);
			#include<fstream>
			ofstream fout("out.d");
			fout << var << endl;

Note that the file formats for C can be used for C++.

Useful commands for higher-level calculations

	Maplesoft	Mathematica
Partial derivative of $x^{1/3}y^4$ with respect to x Note: It can also do a normal derivative	diff(x^(1/3)*y^4,x)	D[x^(1/3) y^4, x]
Multiple derivative $\frac{\partial^n f}{\partial x^n}$	diff(f,[x,n])	D[f,{x,n}]
Total derivative of x^3y^5		Dt[x^3 y^5]

Indefinite Integration: $\int \frac{x}{(1+x^4)^{1/2}}\,dx$	int(x/(1+x^4)^2,x)	Integrate[x/(1+x^4)^(1/2),x]
Definite integration: $\int_0^{\pi} \sin x \exp x^2 dx$	eval(int(sin(x)*exp(x^2)),x=0..Pi)	Integrate[Sin[x] Exp[x^2],{x,0,Pi}]
Numerical integration $\int_0^{\pi/2} \frac{1}{\sin(\tan x)}\,dx$	eval(int(1/sin(tan(x)),x=0..Pi/2)	NIntegrate[1/Sin[Tan[x]],{x,0,Pi/2}]
Multiple integral	int(x^3*y^2,[x=0..a,y=b..c])	Integrate[x^3 y^2,{x,0,a},{y,b,c}] or NIntegrate[Sin[x y],{x,0,Pi},{y,-Pi,Pi}]
Differential equations: $y''(x) - 3y(x) = \exp x$	dsolve(y''(x)-3*y(x)=exp(x))	DSolve[y''[x]-3y[x]==Exp[x],y[x],x]
Numerical solution of differential equations: $y'^2(x) - \cos 3x = 0$ $y(0) = 0,\ x = 0 \sim 10$	eval(dsolve({(y'(x))^2-3*cos(x)=0, y(0)=0},x0..10)	NDSolve[y'[x]^2-Cos[3x]==0,y[0]==0, y[x],{x,0,10}]
Summation: $\sum_{n=1}^{\infty} \frac{1}{n^2}$	Sum(1/n^2,n=1..infinity)	Sum[1/n^2,{n,1,Infinity}]
Product: $\prod_{n=1}^{\infty} \left(1 + \frac{1}{n^2}\right)$	product(1+1/n^2,n=1..infinity)	Product[1+1/n^2,{n,1,Infinity}]
Limits: $\lim_{x \to 0} \frac{\sin x}{x}$	limit(sin(x)/x,x=0)	Limit[Sin[x]/x, x→0]
Root finding of normal nonlinear equations: $x^3 - 4x^2 + 5x - 3 = 0$	solve(x^3-4*x^2+5*x-3=0,x)	Solve[x^3-4x^2+5x-3==0,x]
Root finding for a transcendental equation: $x = \tan x$ near x=2		FindRoot[x==Tan[x],{x,2}]
Solving for a partial differential equation $\frac{\partial^2 f}{\partial t^2} = c^2 \frac{\partial^2 f}{\partial x^2}$	pdsolve(*expression*,f(x,t))	Eq=D[f[x,t], {t,2}]==c^2 D[f[x,t], {x,2}] ; DSolve[eq,f[x,t],{x,t}]