

1.1 مقدمة

1.1.1 مقدمة تاريخية

برنامج المابل (maple) هو نظام الحساب الرمزي (Symbolic Computation System) أو نظام جبر الحاسوب (Computer Algebra System). حيث يعالج برنامج مابل البيانات المدخلة بطريقة رمزية (symbolic) أو جبرية (algebraic) ويعطي النتائج على هيئة رموز وتعابير رياضية بينما تعتمد برامج الرياضيات التقليدية الأخرى في معالجتها على القيم العددية لكل المتغيرات. يحتوي برنامج المابل على مجموعة كبيرة من البرامج الجاهزة (packages) التي تستخدم للحصول على الحل الدقيق أو الحل التقريبي للعديد من المسائل الرياضية. ويحتوي برنامج المابل على أوامر و إيعازات لكتابة البرامج المختلفة كبقية برامج الرياضيات المعروفة.

في مطلع عام 1980 تم تطوير أول نسخة من برنامج المابل في جامعة واترلو (Waterloo University) في كندا من قبل مجموعة الحساب الرمزي (Symbolic Computation Group). وفي عام 1988 أسست شركة كندية جديدة سميت واترلو مابل المحدودة (Waterloo Maple Inc.) لتسويق البرمج. بينما كان العمل مستمر على تطوير برنامج المابل في مختبرات جامعة واترلو (Waterloo University) وجامعة غرب أونتاريو (Western Ontario University) وبتعاون واسع مع مجموعات البحث العالمية في الجامعات الأخرى. وتكللت هذه الجهود عام 1990 بإصدار أول نسخة تعتمد على واجهات الرسوم للمستخدم والتي تدعم ويندوز (windows). واستمرت التطويرات على البرنامج حيث تم إدخال الجافا (Java) في تحسين واجهات المستخدم وذلك في عام 2003 في الإصدار التاسع من برنامج. وفي عام 2005 تم إطلاق الإصدار العاشر والذي يدعم وضع وثيقة (document mode) كجزء من واجهات المستخدم. وفي الإصدار الحادي عشر 2007 تم استخدام بيئة الوثيقة الذكية (smart document environment) لتسهيل تعلم واجهات المستخدم حيث يتضمن هذا الإصدار الكثير من الأدوات الرياضية لتحليل النماذج والمسائل الرياضية المختلفة. وحقق الإصدار الثاني عشر 2008 والإصدار الثالث عشر 2009 نقله نوعية على مستوى واجهات المستخدم وتحرير النتائج وطباعتها والربط بين المابل والبرامج الأخرى مثل الماتلاب (matlab) وقواعد البيانات و نظام (CAD) ولازال العمل مستمر على تحسين وتطوير هذا البرنامج.

1.1.2 مكونات برنامج المابل

يتكون برنامج المابل من ثلاثة أجزاء رئيسية وهي :

(أ) واجهة البرنامج (interface)

واجهة البرنامج يتم من خلالها معالجة المدخلات من التعبيرات الرياضية ، عرض الإخراجات (النتائج) .

(ب) النواة (kernel)

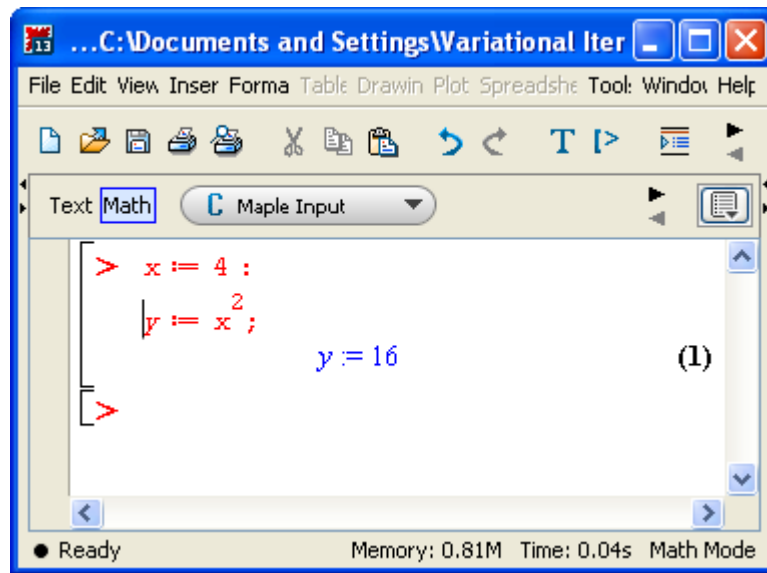
النواة وهي الجزء الذي يترجم المدخلات و يتعامل مع العمليات الجبرية وإدارة تخزين البيانات (ج) المكتبة

وتظم المكتبة جزئيين رئيسيين وهما المكتبة الدوال القياسية (standard library functions) ومكتبة البرامج الجاهزة (packages library) . تتضمن المكتبة الأساسية صيغ أغلب الدوال الرياضية المعرفة على شكل شفرات مابل (maple codes) .

1.2 المفاهيم الأساسية

1.2.1 الخطوات الأولى

سنقوم بكتابة عبارات أو دوال المابل على يمين الرمز < وفي نهاية الجملة نضع نقطتين شارحة (:) أو فارزة منقوطة (;) ، ثم اضغط مفتاح الإدخال (Enter) لتنفيذ الحسابات أو (Shift+ Enter) للانتقال للسطر الثاني السطر التالي). وكما موضح بالشكل



شكل (1.2.1)

من خلال الشكل (1.2.1) يمكن ملاحظ التالي: عند وضع نقطتين شارحة (:) عن نهاية العبارة فان البرنامج ينفذ لكن لا تظهر النتيجة على الشاشة بينما عند وضع فارزه منقوطة (;) فان البرنامج ينفذ ويظهر النتائج على الشاشة .

1.2.2 نظام مساعدة (Help System)

يحتوي برنامج المابل على نظام مساعدة يمكن استخدامه من قبل المستخدم لإيجاد معلومات حول موضوع معين أو لتوضيح عمل دالة معينة وهناك الكثير من الأوامر نستطيع الوصول من خلالها إلى نظام المساعدة وندرج منها الأوامر التالية:

> ?function;

> ?help;

> usage(function);

> example(function);

كما يمكن الحصول النظام المساعدة من شريط المهام لبرنامج المابل وأيضا يمكن الحصول على نظام المساعدة لدالة معينة بعد تظليلها والضغط على مفتاح F2 أو الضغط على F1 + Ctrl.

يمكن للقارئ إن يلاحظ الفرق بين هذه الأوامر من خلال تطبيقها على دالة معينة.

مثال: استخدم نظام المساعدة لبرنامج المابل حول دالة $\sin(x)$ ؟

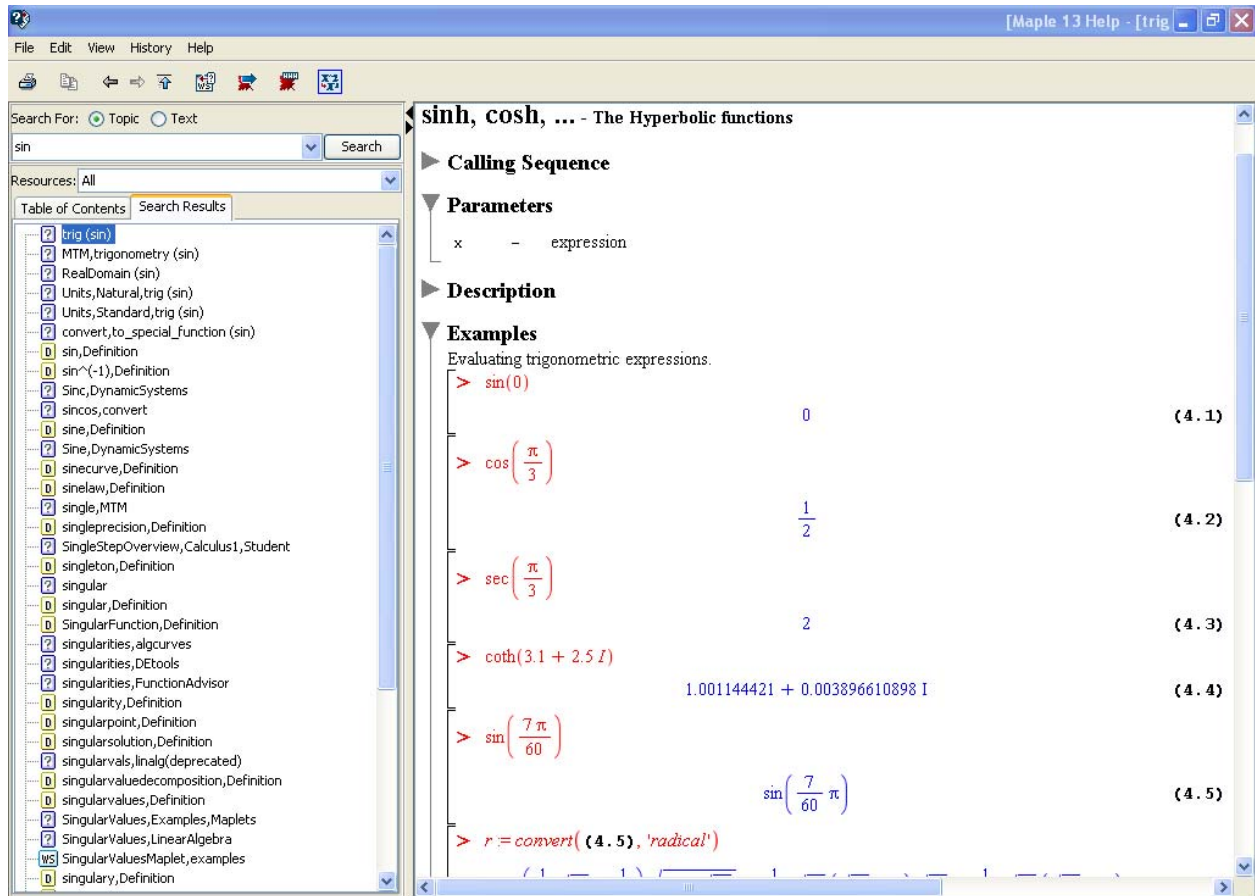
نستخدم احد الأوامر التالية:

> ?sin;

> usage(sin);

> example(sin);

ويمكن أيضا كتابة \sin وظللها اضغط على F2 أو الضغط على F1 + Ctrl. الشكل التالي يوضح نافذة تنفيذ الأمر > ?sin;



شكل (1.2.2)

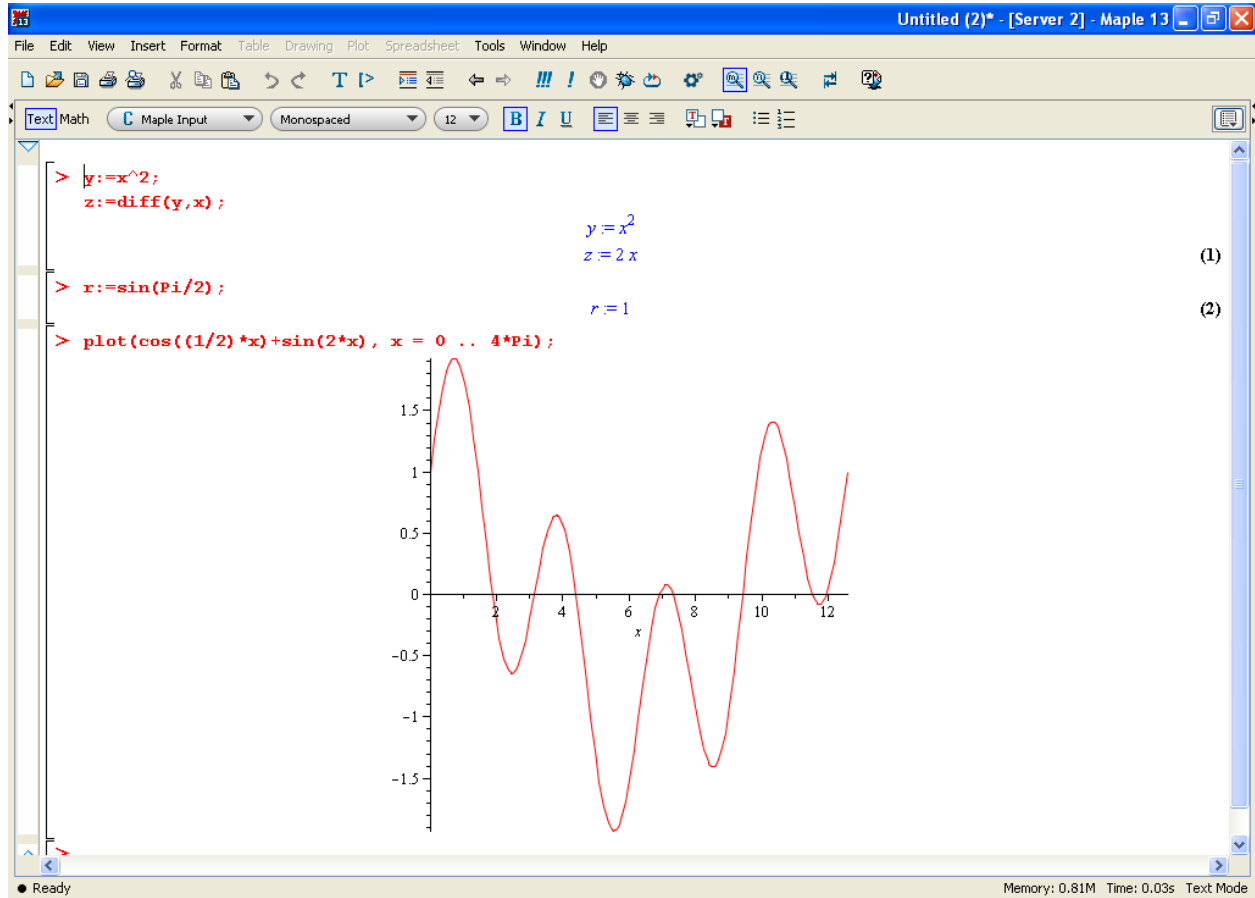
1.2.3 الواجهات و أوراق العمل Worksheets and Interface

أوراق عمل مابل هي ملفات نصية تشرح وتوضح كيفية حل المسائل الرياضية في ميادين الرياضيات والعلوم ، والهندسة. يقدم برنامج مابل نوعين من أوراق العمل وهي التفاعلية (interactive) و التي يكن إعادة استخدامها (reusable). ويمكن إن تستخدم أوراق العمل لتحل محل الآلات الحاسبة ، برامج تطبيق الجداول الحسابية ، وبرامج في لغات مثل فورتران (FORTRAN). ويمكن لأوراق العمل القيام بالمهام التالية:

- (1) تنفيذ العمليات الحسابية
- (2) معالجة العبارات الرياضية
- (3) وصف حل المشاكل العملية

ورقة عمل المابل يتم تنظيمها في سلسلة من مجموعات التنفيذ. كل مجموعة تنفيذ تحتوي على واحد أو أكثر من المناطق. هناك أربعة أنواع من المناطق : المدخلات (input)، والنص (text)، والإخراج (output) والرسومات (graphics). ويمكن للمستخدم تغيير المدخلات (input)، والنص (text) فقط ، بينما

الإخراج (output) والرسومات (graphics) يتم إنشاؤها أثناء تنفيذ برنامج المابل لذلك لا يستطيع المستخدم التلاعب بها. كل مجموعة تنفيذ يتم التعرف عليها بواسطة مربع قوس على طول الحافة اليسرى من ورقة عمل المابل وكما موضح بالشكل (1.2.3)



الشكل (1.2.3)

منطقة الإدخال تحتوي على أوامر المابل وتكون بعد العلامة > وتظهر باللون الأحمر بينما منطقة النص فتحتوي على معلومات غير تنفيذية عادة تتضمن تفسيرات و اشتقاقات رياضية متعلقة بالأوامر في المدخلات المتعلقة بها. ويمكن استخدام منطقة النص لأداء العديد من عمليات التجهيز على النص. بالإضافة إلى تغيير الخطوط والأحجام ، والمحاذاة وغيرها. يمكن أن يكون هناك العديد من الإدخال والمناطق النص ، ولكن لا يوجد أكثر من منطقة إخراج واحدة ، وذلك في مجموعة التنفيذ. وتظهر النتائج باللون الأزرق وتكون مطبوعة بشكل رياضي . وعند اكتشاف خطأ يتم إرسال رسائل خطأ مناسبة من قبل برنامج المابل وتظهر باللون الوردي في منطقة الإخراج وذلك عند تنفيذ البرنامج.

ولمزيد من المعلومات حول أوراق العمل يمكن إن نستخدم الأمر ; worksheet > أو ; shortcut >

1.2.4 البرامج الجاهزة (Packages)

بالإضافة إلى مكتبة الدوال القياسية (standard library functions) ، هنالك العديد من الدوال الخاصة والمتوفرة على شكل برنامج جاهز (Packages) أو جزء من برنامج جاهز (Subpackages) وللمزيد عن المعلومات حول أنواع البرامج الجاهزة (Packages) يمكن استخدام الأمر `> ?index[package];` . البرامج الجاهزة (Packages) أو جزء من برنامج جاهز (Subpackages) يمكن استدعائها عن طريق احد الأوامر التالية

```
> with (Packages);  
  
> with(package[subpackage]);  
  
> package[func](args);  
  
> package[subpackage][func](args);  
  
> func(args);
```

مثال: عند تنفيذ الأمر `> with(algcurves);` ستكون النتيجة

[AbelMap, Siegel, Weierstrass form, algfun_series_sol, differentials, genus, homogeneous, homology, implicitize, integral_basis, is_hyperelliptic, j_invariant, monodromy, parametrization, periodmatrix, plot_knot, plot_real_curve, puiseux, singularities]

مثال: عند تنفيذ الأمر `> with(Algebraic);` ستكون النتيجة

[Content, ConvertRootOf, Degree, Divide, Expand, Extended Euclidean Algorithm, Gcd, Gcdex, GetAlgebraics, GreatestCommonDivisor, MakeMonic, Normal, PrimitivePart, PseudoDivision, Quotient, Reduce, Remainder, Resultant, Squarefree]

مثال: عند تنفيذ الأمر with(LinearAlgebra); ستكون النتيجة

[&x, Add, Adjoint, BackwardSubstitute, BandMatrix, Basis, BezoutMatrix, BidiagonalForm, BilinearForm, CharacteristicMatrix, CharacteristicPolynomial, Column, ColumnDimension, ColumnOperation, ColumnSpace, CompanionMatrix, ConditionNumber, ConstantMatrix, ConstantVector, Copy, CreatePermutation, CrossProduct, DeleteColumn, DeleteRow, Determinant, Diagonal, DiagonalMatrix, Dimension, Dimensions, DotProduct, EigenConditionNumbers, Eigenvalues, Eigenvectors, Equal, ForwardSubstitute, FrobeniusForm, GaussianElimination, GenerateEquations, GenerateMatrix, Generic, GetResultDataType, GetResultShape, GivensRotationMatrix, GramSchmidt, HankelMatrix, HermiteForm, HermitianTranspose, HessenbergForm, HilbertMatrix, HouseholderMatrix, IdentityMatrix, IntersectionBasis, IsDefinite, IsOrthogonal, IsSimilar, IsUnitary, JordanBlockMatrix, JordanForm, KroneckerProduct, LA_Main, LUdecomposition, LeastSquares, LinearSolve, LyapunovSolve, Map, Map2, MatrixAdd, MatrixExponential, MatrixFunction, MatrixInverse, MatrixMatrixMultiply, MatrixNorm, MatrixPower, MatrixScalarMultiply, MatrixVectorMultiply, MinimalPolynomial, Minor, Modular, Multiply, NoUserValue, Norm, Normalize, NullSpace, OuterProductMatrix, Permanent, Pivot, PopovForm, QRdecomposition, RandomMatrix, RandomVector, Rank, RationalCanonicalForm, ReducedRowEchelonForm, Row, RowDimension, RowOperation, RowSpace, ScalarMatrix, ScalarMultiply, ScalarVector, SchurForm, SingularValues, SmithForm, StronglyConnectedBlocks, SubMatrix, SubVector, SumBasis, SylvesterMatrix, SylvesterSolve, ToeplitzMatrix, Trace, Transpose, TridiagonalForm, UnitVector, VandermondeMatrix, VectorAdd, VectorAngle, VectorMatrixMultiply, VectorNorm, VectorScalarMultiply, ZeroMatrix, ZeroVector, Zip]

نلاحظ إن تنفيذ الأمر `with (Packages);` في الأمثلة المتقدمة يعطي تفاصيل البرامج الجاهز (Packages) وما يقوم به من مهام متعددة ومختلفة .

1.2.5 التقييم العددي (Numerical Evaluation)

واحدة من مزايا برنامج المابل هي القدرة على الحسابات المضبوطة أو التقريبية لدقة معينة يحددها المستخدم. حيث يعطي برنامج المابل إجابة محددة لتعابير حسابية عد التعامل مع الأعداد صحيحة ويقلل من ظهور الكسور. وعندما تكون النتيجة عبارة عن عدد غير نسبي (irrational number) فإن الإخراج سيقى من دون أن يحسب أو يقرب لاحظ الأمثلة التالية:

`> -5^(1/3);`

`> -3^22;`

`> (3+8*4+9)/2;`

يتعامل برنامج المابل مع الأرقام بالنظام العشري وبدقة يحددها المستخدم وحسب الأمر `Digits := n`; حيث إن `n` عدد صحيح يمثل الدقة التي يرغب بها المستخدم.

مثال: احسب 1.002^{10} لدقة `n` تساوي 10 , 15 , 30 , 40 .
الشكل (1.2.4) يوضح حل هذه المسألة .

في حالة عدم استخدام الأمر `Digits := n` ; فإن برنامج المابل سيتعامل مع `n=10` . و يستخدم الأمر `Digits := n` ; مرة واحد في البرنامج عادتاً . لمزيد من المعلومات حول الدقة استخدم الأمر `Digits` . ذلك يمكن حساب تعبير معين بدقة محددة تختلف عن دقة باقي المتغيرات وذلك باستخدام الأمر `evalf(expr,n);` حيث أن `expr` يمثل أي متغير أو تعبير حسابي و `n` عدد صحيح يمثل الدقة الشكل (1.2.5) يوضح هذا الأمر

1.3 اللغة المابل (Maple Language).

المابل لغة برمجة رفيعة المستوى ، حسنة التنظيم . أنها تدعم مجموعة كبيرة من هياكل البيانات أو مكونات مابل (الدوال ، المتسلسلات ، المجموعات ، والقوائم ، المصفوفات والجداول وهكذا) ، والعمليات فيما بين هذه المكونات (نوع الاختبار والانتقاء والتكوين ، وهكذا).

1.3.1 المبادئ الأساسية

◀ العمليات الحسابية : + , - , * , / , ^ , mod .

◀ العمليات المنطقية : and , or , xor , implies , not .

◀ عمليات العلاقات : < , > , = , <= , >= , <> .

◀ اسم متغير : هو مزيج من الحروف والأرقام ، الفراغات ، الشارحة (_) وسندرج بعض المتغيرات على سبيل المثال > Aa10:=2; a2:=3; a3_w:=5; برنامج المابل حساس للأحرف الصغيرة والأحرف الكبيرة لذلك يجب إن نكون حذرين عن تعريف متغيرات جديدة وعلى سبيل المثال هناك فرق بين > evalf(Pi); و > evalf(pi) . هنالك كلمات لا يمكن استخدامها كأسماء متغيرات لأنها كلمات محجوزة لكي تستخدم في هيكل بيانات برنامج المابل . حيث هنالك احد عشر مصطلح وهي (and, or, xor,) (not, implies, union, intersect, subset, minus, mod, assuming) ، بالإضافة إلى 36 مصطلح يدخل ضمن لغة مابل ليصبح المجموع 47 الكلمات المحجوزة وتعطى بالجدول التالي

and	assuming	break	by	catch
description	do	done	elif	else
end	error	export	fi	finally
for	from	global	if	implies
in	intersect	local	minus	mod
module	next	not	od	option
options	or	proc	quit	read
return	save	stop	subset	then
to	try	union	use	uses
while	xor			

◀ لاختصر أسماء دوال مابل الطويلة أو أي تعبيرات طويلة نستخدم اسم مستعار جديد اقصر وذلك باستعمال الأمر `> alias(C = expr1);` ولحذف هذا الاختصار للمتغير C نستخدم الأمر `> alias(C =` ؛ C);

◀ عبارة التخصيص لمتغير : لاعطاء أي متغير مثل x قيمة معينة في برنامج المابل فاننا نستخدم : = بالشكل

`< x:=23;` أو `< x:=y^2-cos(y);` . وهناك فرق عندما نكتب الأوامر السابقة بالشكل `< x=23;` أو `< x=y^2-cos(y);` والفرق هو في عدم وضع فارزة منقوطة قبل المساواة وللحصول على ليد من المعلومات استخدم الأمر `> ?` .

◀ يمكن استخدام احد إطراف أي معادلة الأيمن أو الأيسر باستخدام `> lhs(expr);` او `> rhs(expr);`

مثال : جد الطرف الأيمن والطرف الأيسر لكل من العبارات التالية $y = a \cdot x^2 + b$, $r = [2, 5]$, $w + z < 6$.

الشكل (1.3.1) يوضح بشكل تفصيلي حل المثال أعلاه

```
> restart;
e := y = a*x^2+b;
lhs(e);
rhs(e);

> r := 2 .. 5;
lhs(r);
rhs(r);

> h := w+z < 6;
lhs(h);
rhs(h);
```

$e := y = a x^2 + b$
 y
 $a x^2 + b$

$r = 2..5$
 2
 5

$h := w + z < 6$
 $w + z$
 6

الشكل (1.3.1)

◀ يمكن كتابة المعادلات أو المتراجحات باستخدام

> Name of Equation:=lift side=right side;

> Name of inequality :=lift side =< right side;

> Name of inequality :=lift side >= right side;

الشكل (1.3.1) يوضح طريقة كتابة المعادلات والمتراجحات.

◀ الفترات : يستخدم برنامج المابل a..b للتعبير عن الفترة [a,b] وعادتا ما تظهر في التكامل المحدد ورسم الدوال وغيرها . الشكل (1.3.1) وضح طريقة كتابة الفترة [2,5] .

◀ العبارات : و هي تعليمات الإدخال التي يتم إدخالها من لوحة المفاتيح (keyboard) والتي سيتم تنفيذها من قبل البرنامج وندرج بعضا منها (, restart, if, function, for, end, do, by, break, while, not, save, return).

◀ restart : إن من المهم جدا عن كتابة برنامج جديد بلغة مابل إن تكون العبارة الأولى فيه هي

restart: > لكي تفرغ الذاكرة من التنفيذيات السابقة للبرنامج ويعاد تصفير كل المتغيرات المستخدمة في البرنامج.

◀ الفارزة المنقوطة : و النقطتين الشارحة : تستخدمان لفصل عبارات برنامج المابل والفرق بينهما هو عند تنفيذ البرنامج فان نتيجة الجملة المنتهية بفارزة منقوطة ستظهر بينما لا تظهر نتيجة العبارة المنتهية بالنقطتين الشارحة .

وهما فواصل البيان (منقوطة وسرطان القولون) وتعادل في حالة البيانات المتداخلة واحد أو أكثر المستويات (في الداخل إذا التصريحات ، لا التصريحات ، أو الإجراءات).

◀ الأقواس : إن برنامج المابل حساس لنوعية الأقواس المستخدمة فيه . ويمكن حصر انواع الاقواس بالاتي :

() يستخدم في العمليات الجبرية و تعيين متغيرات الدوال ; $f=(x+y)^3-z$ او $y:=\sin(2*t)$ >

[] يستخدم في كتابة قائمة ولكتابة المتجهات والمصفوفات.

{ } يستخدم لكتابة المجموعات {a,b,c}.

◀ أنواع الاقتباس : هناك ثلاثة انواع من علامات الاقتباس وبرنامج المابل يستخدم كل منها لغرض معين وسندرج الأنواع والغرض منها

'expr' تستخدم لحذف حساب التعبير فمثلا لحذف قيمة المتغير x نستخدم >x:='x';

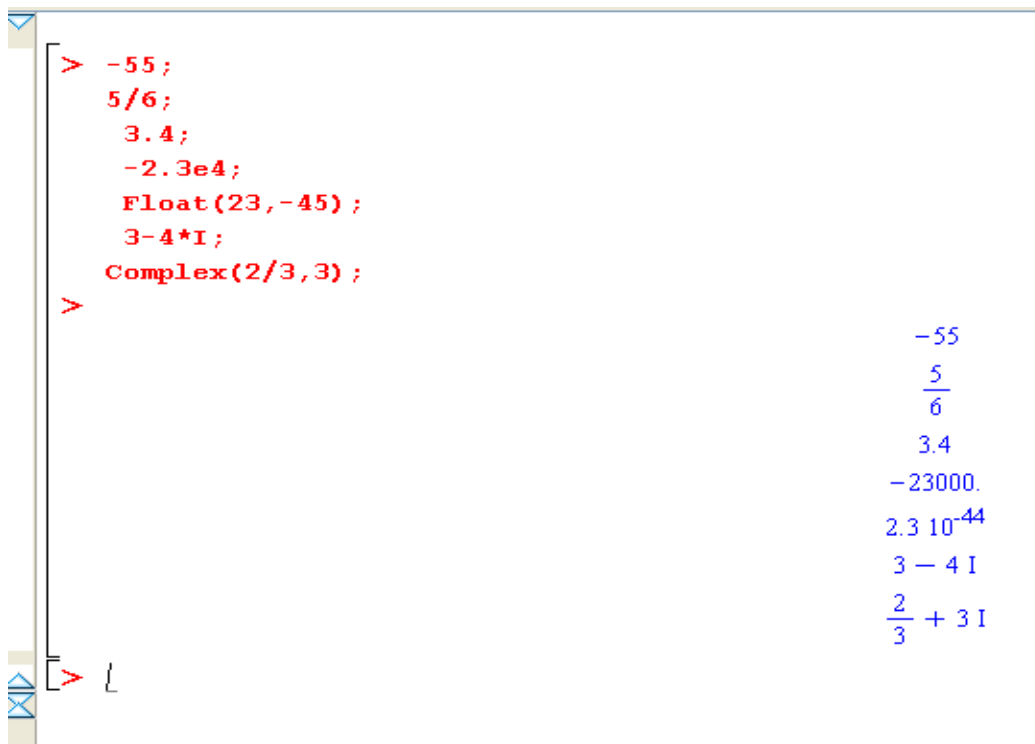
'expr' تستخدم لتشكيل رمز أو اسم ;'the name:=7' > كما تستخدم في الطباعة

k:=5; print('the value of k is',k)

◀ لكتابة تعليق ضمن البرنامج فان التعليق يكتب مسبوق ب # حيث ان أي عبارة مسبوق ب # ه عبارة غير تنفيذية.

1.3.2 الثوابت Constants

نعلم انه هنالك العديد من الثوابت منها الصحيح ,النسبي ,الحقيقي , المعقدة , والجذرية ، وعلى سبيل المثال لاحظ الشكل (1.3.2) لمعرفة طريقة كتابة كل منها



```
> -55;  
5/6;  
3.4;  
-2.3e4;  
Float(23,-45);  
3-4*I;  
Complex(2/3,3);  
>
```

-55
 $\frac{5}{6}$
3.4
-23000.
 $2.3 \cdot 10^{-44}$
3 - 4 I
 $\frac{2}{3} + 3 I$

شكل (1.3.2)

◀ الثوابت المعروفة في برنامج المابل هي `false, gamma, infinity, true, Catalan, FAIL, Pi`

الشكل (1.3.3) يوضح استخدام هذه الثوابت

```
> constants;  
type(f(exp(gamma)+3)+1/4, 'constant');  
type(f(exp(gamma)+x)+1/4, 'constant');  
constants := constants, x;  
type(f(exp(gamma)+x)+1/4, 'constant');  
false,  $\gamma$ ,  $\infty$ , true, Catalan, FAIL,  $\pi$   
true  
false  
constants := false,  $\gamma$ ,  $\infty$ , true, Catalan, FAIL,  $\pi$ , x  
true  
> [
```

الشكل (1.3.3)

◀ الزوايا