Maple T.A. Tips and Techniques - Algorithmic Variables (Part I)

Introduction

You can use algorithmic variables to randomize your questions in Maple T.A. These algorithmic variables can use the built-in algorithmic generator or can use any of Maple's randomization routines. This article discusses various functions built into Maple T.A. that can be used to create random variables and provides some examples. A later article will discuss routines and packages in Maple that can be used in Maple T.A. to create random variables.

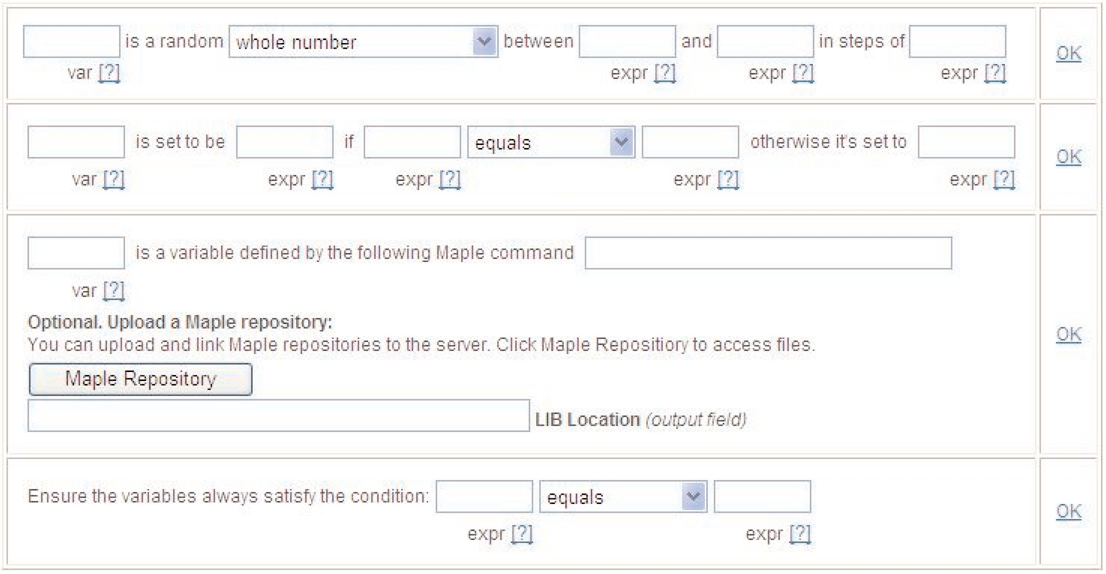
Algorithmic Variables in Maple T.A.

To create an algorithmic variable in Maple T.A., use the algorithm designer in the Question Bank Editor. Algorithmic variables can be used in any question type and can be used in the question text, the answer region, as well as in the hints and feedback sections. Algorithmic variables are denoted by a dollar sign, for example $var.

To create an algorithmic variable:

1. Upload an existing Question Bank or create a new one.
2. Open an existing Topic or add a new one.
3. Click **Add a question**.
4. Choose the question type from the drop-down menu.
5. In the Algorithm field, click **Add**.
6. If you know the format of the command, you can type it in the text box and proceed to Step 8. Otherwise, click **Show Designer** to load the algorithm designer tool.
7. Type the required information in the algorithm designer template. Note that you do not need to use the $ to denote variables in the template. Click **OK** to add the command to the algorithm text box and displays a possible value for the variable.
8. Click **Refresh** to have Maple T.A. generate another possible value for the variable(s).
9. Click **Save** to return to the main page for the question.

You can now continue creating your question and can use the variables that you created.



var [?] is a random whole number between expr [?] and expr [?] in steps of expr [?] OK

var [?] is set to be expr [?] if expr [?] equals expr [?] otherwise it's set to expr [?] OK

var [?] is a variable defined by the following Maple command

Optional. Upload a Maple repository:
You can upload and link Maple repositories to the server. Click Maple Repository to access files.

Maple Repository

LIB Location *(output field)* OK

Ensure the variables always satisfy the condition: expr [?] equals expr [?] OK

You can use the algorithm designer to create simple random integers, set conditions on the variables, or enter a Maple command that will generate a random object.

Randomization Functions in Maple T.A.

Maple T.A. contains several functions for creating random variables. You can create random integers or real numbers, perform operations on lists of items, or set conditions on variables.

It is important to note that the algorithm generator in Maple T.A. works linearly. This means you must be careful to define variables before referencing them in other variable definitions.

Random Numbers

You can define random numbers or display given numbers in a specific format using the following functions.

rand(m,n) - enerates a random real number between m and n (inclusive), rounded to 6 decimal places
rand(m,n,k) - as above, but expressed to k significant digits
range(n) - generates a random integer between 1 and n (inclusive)
range(m, n) - generates a random integer between m and n (inclusive)
rint(n) - generates a random integer between 0 and n-1 (inclusive)
rint(m, n) - generates a random integer between m and n-1 (inclusive)
decimal(n,x) - returns x expressed as a floating-point number rounded to n decimal places

List Operations

There are several functions in Maple T.A. that operate on lists. These allow you to choose an item out of a list based on a property or a value. They are very useful if you want to create a question with the same stem or base information, but have answers with different properties.

min(a,b,c,d,…) - returns the smallest element from a list
max(a,b,c,d,…) - returns the largest element from a list
indexof(k,a,b,c,d,…) - returns the index of item k within a list. The position numbering in the list starts at 0. If the item k is not present in the list, the function returns a value of -1.
rank(n,a,b,c,d,…) - returns the nth element from an ordered list. The position numbering in the list starts at 1. If n is less than one or larger than the number of elements in the list, the function returns an error message.
switch(n, a, b, c, …) - returns the nth item from a list. The position numbering in the list starts at 0. If n is larger than one less than the number of elements in the list, the function returns the last element of the list. If n is less than zero, the function returns the first element.

Conditions

You can place conditions on your variables. For example, you can specify that two variables are not to be equal, or define a value for a specific variable based on the values of other variables.

condition:x - where x is the definition of the condition
if(a,b,c) -if a is nonzero, return b. Otherwise, return c
eq(a,b) - returns 1.0 if a is equal to b. Otherwise, it returns 0.0.
ne(a,b) - returns 1.0 if a is not equal to b. Otherwise, it returns 0.0.
gt(a,b) - returns 1.0 if a is greater than b. Otherwise, it returns 0.0.
ge(a,b) - returns 1.0 if a is greater than or equal to b. Otherwise, it returns 0.0.
lt(a,b) - returns 1.0 if a is less than b. Otherwise, it returns 0.0.
le(a,b) - returns 1.0 if a is less than or equal to b. Otherwise, it returns 0.0.
not(a) - returns 1.0 if a is equal to 0.0. Otherwise it returns 0.0.

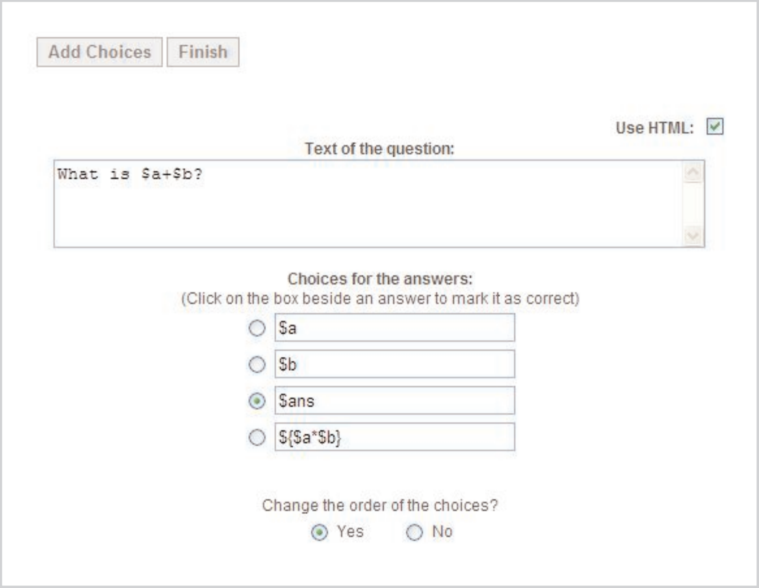
Examples

These examples include the Maple T.A. source code required to generate the question. If you want to easily load the questions into Maple T.A., download the April05Tips.qu file from the Web site and upload it into the Question Bank Editor.

1. Basic Arithmetic

In this example we use a multiple choice question and ask the student to add two numbers. In the algorithm designer, we define two integer variables, $a and $b, and then define a third variable, $ans, which is the sum of $a and $b. In the choices to be displayed, we create an inline algorithmic variable that is the product of $a and $b by using the format ${expression}.

```
mode=Multiple Choice@
name=arithmetic@
editing=useHTML@
algorithm=$a=range(2,10);
$b=range(3,15);
condition:not(eq($a,$b));
$ans=$a+$b;@
question=What is $a+$b?@
answer=3@
choice.1=$a@
choice.2=$b@
choice.3=$ans@
choice.4=${$a*$b}@
```



Use HTML: ☑

Text of the question:

```
What is $a+$b?
```

Choices for the answers:
(Click on the box beside an answer to mark it as correct)

○ $a
○ $b
● $ans
○ ${$a*$b}

Change the order of the choices?
● Yes ○ No

One of the features of Maple T.A. is the ability to define inline algorithmic expressions directly in questions by using the format ${expression}.

2. Calculating Area

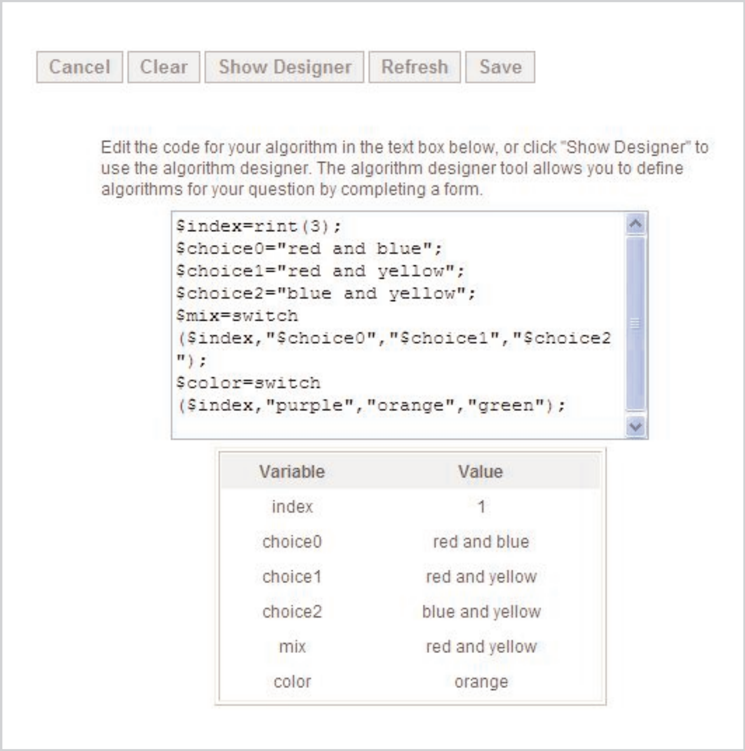
In this example we use a numeric question and ask the student to calculate the area of a rectangle given two algorithmically-generated values for the width and length. This example shows that you can combine commands, such as decimal(x,n) and rand(m,n). We've also added a condition on the variables $width and $length so that they will not be equal.

```
question=Given a rectangle with width of $width meters and length of $length meters, calculate its area.@
answer.num=$area@
answer.units=m^2@
showUnits=true@
grading=exact_value@
negStyle=both@
numStyle=thousands   @
mode=Numeric@
name=area@
algorithm=$width=decimal(1,rand(2.5,6.5));
$length=decimal(1,rand(4.5,10.0));
condition:not(eq($width,$length));
$area=$width*$length;@
```

3. Color Combinations

This fill-in-the-blanks example illustrates the use of the switch command to customize the question. Given two lists of information, one of three pairs of colors and one of the colors when the pairs of colors are mixed, we can define a random index value and then base the question on one of these three pieces of matching information.

```
mode=Blanks@
name=color@
algorithm=$index=rint(3);
$choice0="red and blue";
$choice1="red and yellow";
$choice2="blue and yellow";
$mix=switch($index,"$choice0","$choice1","$choice2");
$color=switch($index,"purple","orange","green");@
question=When you mix $mix, you get  <1> @
blank.1=%24color@
extra=@
format.input=text@
```



Edit the code for your algorithm in the text box below, or click "Show Designer" to use the algorithm designer. The algorithm designer tool allows you to define algorithms for your question by completing a form.

```
$index=rint(3);
$choice0="red and blue";
$choice1="red and yellow";
$choice2="blue and yellow";
$mix=switch
($index,"$choice0","$choice1","$choice2
");
$color=switch
($index,"purple","orange","green");
```

Variable	Value
index	1
choice0	red and blue
choice1	red and yellow
choice2	blue and yellow
mix	red and yellow
color	orange

The Algorithmic Editor shows both the algorithmic commands and a sample value for the variables.

Summary

There are a wide variety of routines that can be used to create algorithmic variables within Maple T.A. This article summarized these functions and provided a few examples. If you'd like more information on the randomization functions inside Maple T.A., see the Advanced Authoring section of the online Instructor Help.

Maplesoft™
command the brilliance

Corporate Headquarters
Maplesoft, Waterloo, Canada
t. 519.747.2373 | f. 519.747.5284
800.267.6583 (US & Canada)
info@maplesoft.com

European Office
Maplesoft Europe GmbH, Zug, Switzerland
t. +41 (0)41 763 33 11
f. +41 (0)41 763 33 15
info-europe@maplesoft.com

www.maplesoft.com | www.mapleapps.com

© Maplesoft, a division of Waterloo Maple Inc., 2005. Maplesoft, Maple, and Maple T.A. are trademarks of Waterloo Maple Inc. All other trademarks are property of their respective owners.

Maple T.A. Tips and Techniques – Algorithmic Variables (Part II)

Introduction

You can use algorithmic variables to randomize your questions in Maple T.A. These algorithmic variables can use the built-in algorithmic generator or can use any of Maple's randomization routines. In a previous article, Algorithmic Variables (Part I), we covered various functions inside Maple T.A. that can be used to create random variables. This article discusses several of the routines and packages in Maple that can be used in Maple T.A. to create random variables.

Algorithmic Variables in Maple T.A.

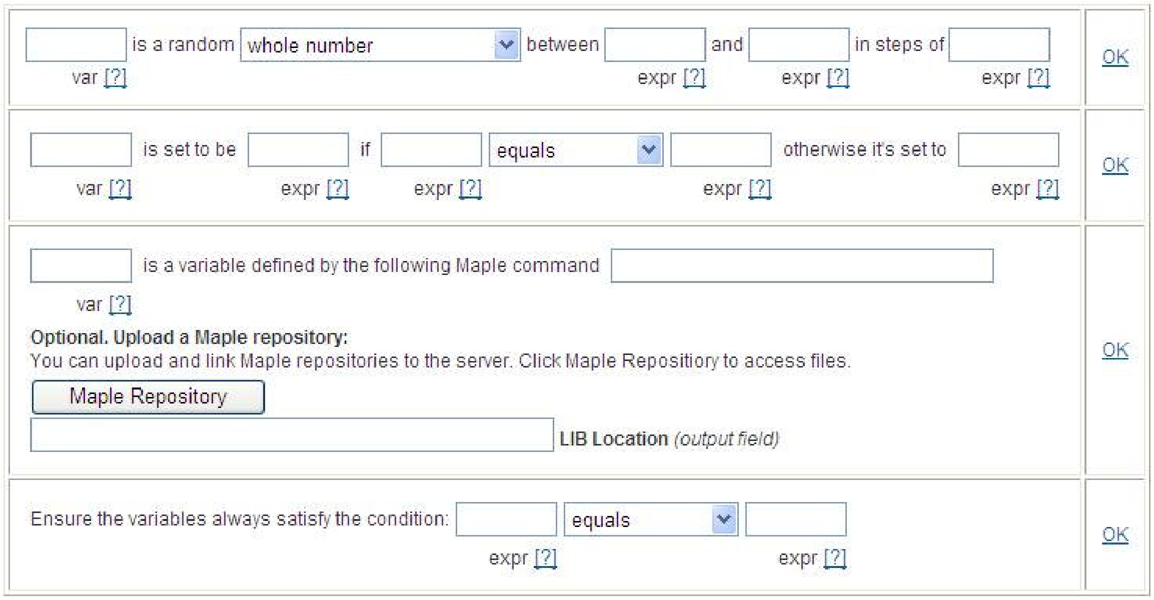
To create an algorithmic variable in Maple T.A., you can use the algorithm designer in the Question Bank Editor. Algorithmic variables can be used in any question type and can be used in the question text, the answer region, as well as in the hints and feedback sections. Algorithmic variables are denoted by a dollar sign, for example $var.

To create an algorithmic variable:

1. Upload an existing Question Bank or create a new one.
2. Open an existing Topic or add a new one.
3. Click **Add a question**.
4. Choose the question type from the drop-down menu.
5. In the Algorithm field, click **Add**.
6. If you know the format of the command, you can type it in the text box and proceed to Step 8. Otherwise, click **Show Designer** to load the algorithm designer tool.
7. Type the required information in the algorithm designer template. Note that you do not need to use the $ to denote variables in the template. Click **OK** to add the command to the algorithm text box and display a possible value for the variable.
8. Click **Refresh** to have Maple T.A. generate another possible value for the variable(s).
9. Click **Save** to return to the main page for the question.

You can now continue creating your question and can use the variables that you created.

Algorithm Design Tool



is a random whole number between and in steps of

is set to be if equals otherwise it's set to

is a variable defined by the following Maple command

Optional. Upload a Maple repository:
You can upload and link Maple repositories to the server. Click Maple Repository to access files.

LIB Location *(output field)*

Ensure the variables always satisfy the condition: equals

You can use the algorithm designer to create simple random integers, set conditions on the variables, or enter a Maple command that will generate a random object.

Randomization Routines in Maple

Maple has many randomization routines that can be used in Maple T.A. This section discusses a subset of the available functions. The function rand will generate a random integer, while randpoly will generate a random polynomial with specific properties. The LinearAlgebra package contains the RandomMatrix and RandomVector functions. The RandomTools package contains a variety of functions for creating random Maple objects, from simple random integers or polynomials to lists of values that follow specific probability distributions. In addition, since Maple is a programming language, you could write your own customized program in Maple to generate any type of variable or object that you need.

All random number generators use the same underlying random number sequence, so when using Maple's randomization routines, you must include randomize(): as the first part of the call. This uses a number based on the system clock as the initial state instead of the default seed that is used in Maple. If you don't include randomize(), each call will return the same sequence of values since each call to Maple from within Maple T.A. starts a new Maple kernel and that will reset the default randomization seed.

Notes

1. When entering multiple commands, ensure that all (except the last) have a trailing colon. The last command should have a trailing semi-colon.

2. When referencing negative (or possibly negative) random variables in a maple variable definition, be sure to place the negative variable in parentheses, otherwise you may receive an error message.

Random Integers
The rand( ) function is a simple way to create random integers. You can also you the RandomTools[Generate] command as shown in the RandomTools section below.

rand( )	Generates a 12-digit nonnegative integer
rand(n)( )	Generates a random integer between 0 and n-1 (inclusive)
rand(a..b)( )	Generates a random integer between a and b (inclusive)

Random Polynomials
The randpoly function is the most straightforward way to create random polynomials. You can also you the RandomTools[Generate] command as shown in the RandomTools section below.

randpoly(*variables*, *options*)	Generates a random polynomial in terms of *variables* using the *options* specified. The most common options are used to specify the type of coefficients and exponents, the degree of the polynomial, and the number of terms in the polynomial.

These options are included as follows:

coeffs=rand(a..b) – the default is rand(-99..99)
degree=n – the default is 5
expons=n – the default is rand(6)
terms=m – the default is 6, but this value is overridden by the degree option if there is a conflict

LinearAlgebra Functions
The two most common random objects that can be created using the LinearAlgebra package are Matrices and Vectors. You can create both row and column vectors.

LinearAlgebra[RandomMatrix](m, n, density, generator)	Generates a random m x n matrix using the options specified.
LinearAlgebra[RandomVector][o](dimension, density, generator)	Generates a random vector using the options specified. [o] can be used to specify a row vector instead of a column vector (which is the default).

Some examples:

$M=maple("randomize(): LinearAlgebra[RandomMatrix](3,3)"); returns a random 3x3 matrix

$V=maple("randomize(): LinearAlgebra[RandomVector][row](6, generator=rand(1..5)/10))"); returns a row vector of 6 elements whose entries are rational numbers with a denominator of 10 and a numerator between 1 and 5. The elements are returned in simplified form.

RandomTools Functions

There are many functions in the RandomTools package, including sub-packages that implement different pseudo-random number generators. There are several RandomTools[Generate] functions can be used to create random objects, a selection of which are included in the table below. Each of the functions in the table below should follow RandomTools[Generate], as in RandomTools[Generate](choose({a,b,c,d,e,f}));

choose(collection)	Select one of the entries in a non-empty collection with equal probability
complex(*flav*)	A random complex number with real and imaginary parts described by the given random flavor *flav*
exprseq(*flav*,n)	An expression sequence with n entries where each entry is described by the given random flavor *flav*
float(opts)	A random floating-point number in a particular range. The options can include a range (e.g., range=2.532..7.723) and the number of digits (e.g., digits=4).
identical(*expr*)	Describes the object *expr* itself

integer(opts)
negint(opts)
nonnegint(opts)
nonposint(opts)
nonzeroint(opts)
posint(opts)
	A random integer in a particular range. The options can include a range (e.g., range=0..10) and a statistical distribution from where the integer is chosen (e.g., distribution=poisson[5])

list(*flav*, n)	A list with n entries where each entry is described by the given random flavor *flav*
listlist(*flav*, m, n)	A list of m lists, each with n entries, where each entry is described by the given random flavor *flav*. In the case where only m is given, n is assumed equal to m.

rational(opts)
negative(opts)
nonnegative(opts)
nonpositive(opts)
nonzero(opts)
positive(opts)
	A random rational number in a particular range. The options can include a range, a statistical distribution from where the rational number is chosen, or a denominator.

polynom(coeffs, x, opts)	A random polynomial in a given number of variables x with coefficients coeffs of a given random flavor. The default degree of the polynomial is 5, but you can specify the degree of the polynomial by using the degree option (e.g., degree=3).
set(*flav*,n)	A set containing n entries where the entries of the set are described by the given random flavor *flav*. The final set can contain fewer

	than n entries if the same object is generated more than once.
truefalse(opt)	Describes the values true or false. By default, the values true and false will be chosen with equal probability, but you can modify this by using the probability option (e.g., probability=p, where p is a numeric value between 0 and 1 that specifies the probability that the object will be true).

Some examples:

$a=maple("randomize(): RandomTools[Generate](integer(range=2..9))"); returns a random integer between 2 and 9 (inclusive)

$b= maple("randomize(): RandomTools[Generate](list(rational(denominator=30), 10))"); returns a list of 10 rational numbers whose denominator is 30. They are returned in simplified form.

$c=maple("randomize(): RandomTools[Generate]([integer(range=3..10),rational(range=3..10, denominator=13)])"); returns a list of two elements where the first element is an integer in the range 3 to 10 and the second element is a rational number between 3 and 10 whose denominator is 13

You can also include other Maple commands in the algorithmic variable definition. $d=maple("randomize(): seq(RandomTools[Generate](integer(distribution=normald[3.5, 1.5])), i=1..10)"); creates a 10-item sequence of integers that follow a normal distribution.

Examples

The following examples include the Maple T.A. source code required to generate the question. If you want to easily load the questions into Maple T.A., download the Aug05Tips.qu file from the Web site and upload it into the Question Bank Editor.

1. Degree of polynomial
This numeric question uses Maple to generate a random polynomial and Maple's MathML[ExportPresentation] function to generate a nicely formatted version of it. We then ask the student to determine the degree of the polynomial.

```
question=What is the degree of $displaypoly?@
answer.num=$b@
answer.units=@
showUnits=false@
grading=exact_value@
```

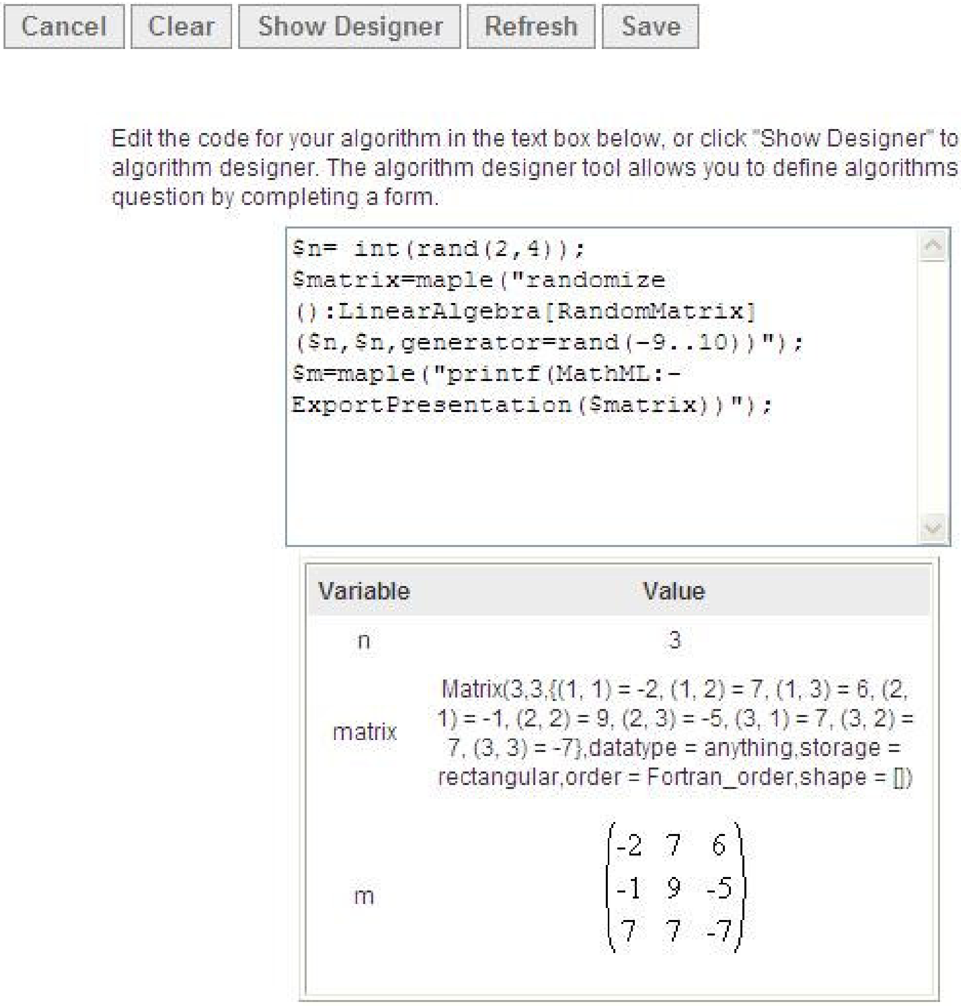
```
negStyle=minus@
numStyle=thousands scientific dollars arithmetic@
mode=Numeric@
name=Polynomial@
algorithm=$a=range(0,2);
$b=range(2,5);
$poly=maple("randomize(): randpoly(x,degree=$b)");
$displaypoly=maple("printf(MathML[ExportPresentation]($poly))");@
```

2. Intersection of Sets
Here we use the RandomTools[Generate](set) function, to generate two unique sets and ask the student to determine the intersection. This question is written as a Maple-Maple question because we want the student to enter in a set as the answer.

```
question=What is the intersection of $set1 and $set2? <p><br></p>
Enclose your answer in braces.@
maple=is($RESPONSE=$ANSWER);@
maple_answer=$set1 intersect $set2@
type=maple@
mode=Maple@
name=Intersection@
editing=useHTML@
algorithm=$set1=maple("randomize():
RandomTools[Generate](set(posint(range=8),5))");
$set2=maple("randomize(): RandomTools[Generate](set(posint(range=10),4))");@
```

3. Matrix Determinant
In this example, we use Maple's LinearAlgebra[RandomMatrix] function to create a square matrix with entries between -9 and 10. Again, we use MathML[ExportPresentation] to generate a nicely formatted version for display.

```
question=Calculate the determinant of the following matrix.<br>$m.@
maple=evalb($ANSWER-$RESPONSE=0);@
maple_answer=LinearAlgebra[Determinant]($matrix);@
type=formula@
mode=Maple@
name=Matrices - Determinants@
editing=useHTML@
algorithm=$n= int(rand(2,4));
$matrix=maple("randomize():LinearAlgebra[RandomMatrix]($n,$n,generator=rand(-
9..10))");
$m=maple("printf(MathML:-ExportPresentation($matrix))");@
```



Edit the code for your algorithm in the text box below, or click "Show Designer" to use the algorithm designer. The algorithm designer tool allows you to define algorithms for your question by completing a form.

```
$n= int(rand(2,4));
$matrix=maple("randomize
():LinearAlgebra[RandomMatrix]
($n,$n,generator=rand(-9..10))");
$m=maple("printf(MathML:-
ExportPresentation($matrix))");
```

Variable	Value
n	3
matrix	Matrix(3,3,{(1, 1) = -2, (1, 2) = 7, (1, 3) = 6, (2, 1) = -1, (2, 2) = 9, (2, 3) = -5, (3, 1) = 7, (3, 2) = 7, (3, 3) = -7},datatype = anything,storage = rectangular,order = Fortran_order,shape = [])
m	$\begin{pmatrix} -2 & 7 & 6 \\ -1 & 9 & -5 \\ 7 & 7 & -7 \end{pmatrix}$

Two Maple commands are being used. One to create the Matrix construct, and one to generate a nicely formatted version for display.

Summary

There are a wide variety of routines and packages in Maple that can be used to create algorithmic variables in Maple T.A. You can use straight functions or combinations of functions to achieve the result that you want. This article summarized a subset of such functions. If you’d like more information on the available randomization routines in Maple, refer to the Maple documentation.

Application Demonstration

Maplesoft
www.maplesoft.com

Techniques for Creating Algorithmically Generated Questions

© Maplesoft, a division of Waterloo Maple Inc., 2007

Maple T.A. supports algorithmic questions, which allows you to generate many different questions that have the same basic structure. Algorithmic questions are very useful for generating hundreds of practice questions for students, and can also be used in assignments to ensure each student gets a different version of the question. This document explores techniques for creating good algorithmically generated questions. The focus is not so much on the technical aspects of creating these questions (see the Instructor help for the technical details), but rather on ways to think about your problem which lead to better algorithms. It includes examples of ways to generate values with different properties, and introduces you to some Maple commands that may be useful for generating question values.

Start with the Answer

Example 1: Consider the following problem:

What is the square root of *n*?

Approach 1:

1) Select a number, *n*, between 1 and 100
2) Take the square root of *n*
3) Check if the square root of *n* is an integer
4) If it isn't, go to back to step 1) and try again

Approach 2:

1) Select a number, *k*, between 1 and 10
2) Set *n* = *k*^2

Both approaches lead to acceptable values for *n*, but the first approach will take much longer to generate an acceptable question. It uses a "pick a possible value, test to see if it has the necessary characteristics, if not reject, and pick again" approach. It is more complicated to understand, and therefore will probably take longer to write and require more testing. Since it is more complicated, it will also be harder to reuse or modify later.

The second approach is to decide what characteristics you need in your variables, and then write the generation steps to ensure that only values that have those characteristics get generated in the first place.

The fundamental difference between the two approaches is that the first one starts with the question, and the second starts with the solution. *When creating algorithmic questions it is almost always better to start with the solution.*

Example 2: Factor a quadratic, $x^2 + b\,x + c$.

We need to create the quadratic for use in the question, and we need to ensure that the roots are both real and both integers. One approach is to find values for *b* and *c* using the quadratic formula, $x = \frac{-b \pm \sqrt{b^2 - 4\,a\,c}}{2\,a}$, where *a*=1.

To ensure real roots, we must pick values such that $b^2 - 4\,a\,c \geq 0$. To ensure integer roots, we also need to make sure that it is a perfect square and that the numerator is a multiple of $2\,a$. All of this is possible but it would take some thought and care to get it right.

However, if we start with the solution, the problem becomes much simpler. If the factorization is $(x + m)\cdot(x + n)$, then we just need to pick nice, integer values for *m*, and *n*. Then we expand the factors, and the question is created.

Picking Values with Specific Characteristics

Starting from the solution is a key component to creating a good algorithmic question. Often, you will need to select values with particular characteristics. Here, too, the best approach is to find a way to build in the characteristics you need, rather than use a pick-test-reject-pick again method.

Problem: Pick a number from 10, 20, 30, …, 100.
Solution: Pick a number from between 1 and 10, and then multiply it by 10
Implementation: $a = `rint(1,11)*10`

In Maple T.A., use the **rint** function to generate a random integer. The command `rint(m,n)` will generate a random integer between *m* and *n*-1 inclusive. The command `rint(n)` will return a number between 0 and *n*-1.

Note that with this implementation, if you wish to change the upper bound, for example to 200, you only need to change a single number.

Problem: You need to make three variables, all different, and they all have to have a common factor.
Solution: Fix a list of possible common factors, for example {2, 3, 5}. Pick each variable from a different range, say *a* comes from 1-3, *b* comes from 4-6, and *c* comes from 7-9. Randomly pick a common factor, *k*, from the common factor list. Your values are then $k\cdot a, k\cdot b, k\cdot c$
Implementation:

```
$a = rint(1,4)
$b = rint(4,7)
$c = rint(7,9)
$k = switch(rint(3), 2, 3, 5)
```

In Maple T.A., `switch(n, a, b, c,…)` returns the *n*th item from the list *a*, *b*, *c*, …, where the position numbering starts at 0.

Problem: You need your variable to be one of: $0, \frac{\pi}{2}, \frac{\pi}{3}, \pi, 2\,\pi$.

Solution: Sometimes the simplest solution is best. If you need to pick from a fixed set of items, put all the possible values in a list, and then randomly pick one of them.

Implementation:

```
$v = switch(rint(5), Pi/2, Pi/3, Pi, 2*Pi, 0)
$v_Display =
maple("MathML[ExportPresentation](identify($v))")
```

Or in one step

```
$v = maple("randomize();
[Pi/2, Pi/3, Pi, 2*Pi, 0][(rand(1 .. 5))()]");
```

Using Maple's Random Generation Tools

When using Maple to generate your values you have access to many built-in tools for creating random objects with particular properties. These tools should be used selectively. Calls to built-in Maple T.A. commands are the most efficient option. If you just need to pick random integer values, you may find the Maple T.A. functions **range()** or **rint()** sufficient. However, for the cases where it does make sense to use these random generation tools, you will find them very powerful and very convenient.

Here are a few examples of the sorts of tools that are available. For more information, consult the Maple help system.

RandomTools[Generate]

This command can be used to generate a wide variety of objects, including fractions, integers, non-zero integers, polynomials, complex numbers, and lists. You can further specify the characteristics of each type of object.

You can ask for a list of 5 integers between -10 and 10, and excluding zero:

$RandomTools[Generate](list(nonzeroint(range=-10..10), 5));$

$$[-3, 10, 5, -9, -2]$$

You can ask for a fifth degree polynomial whose coefficients are fractions with 12 in the denominator (Maple will simply fractions like 9/12 before displaying the result).

```
RandomTools[Generate](polynom(positive(denominator = 12), x,
    degree = 5));
```

$$\frac{5}{12}+\frac{3}{4}\,x+\frac{7}{12}\,x^{2}+\frac{1}{3}\,x^{3}+x^{4}+\frac{1}{4}\,x^{5}$$

For more information, see the **?RandomTools[Generate]** help page inside Maple.

LinearAlgebra[RandomMatrix]

This command can be used to create a random matrix of a specified size, with entries selected using the criteria given by the user.

To create a 3x3 matrix with entries from 1 to 9:

```
LinearAlgebra[RandomMatrix](3, 3, generator = 1 ..9);
```

$$\begin{bmatrix} 8 & 3 & 4 \\ 6 & 8 & 9 \\ 5 & 5 & 9 \end{bmatrix}$$

To create a lower-diagonal matrix with the same type of entries:

```
LinearAlgebra[RandomMatrix](3, 3, generator = 1 ..9, outputoptions
    = [shape = triangular[lower]]);
```

$$\begin{bmatrix} 9 & 0 & 0 \\ 3 & 7 & 0 \\ 9 & 8 & 9 \end{bmatrix}$$

For more information, see the **?LinearAlgebra[RandomMatrix]** help page.

Tip – To call Maple from within Maple T.A. the maple command must be enclosed in maple(“ “).

For example,

```
$poly =
maple(“RandomTools[Generate](polynom(positive(denominator=1
2),x,degree=5))”)
```

Legal Notice: The copyright for this application is owned by Maplesoft. The application is intended to demonstrate the use of Maplesoft Products to solve a particular problem. It has been made available for product evaluation purposes only and may not be used in any other context without the express permission of Maplesoft.

Thank you for evaluating this Maple application sample www.maplesoft.com

Application Demonstration

Maplesoft
www.maplesoft.com

Three Steps to More Effective Algorithms in Maple T.A.

© Maplesoft, a division of Waterloo Maple Inc., 2007

Every question type in Maple T.A. can take advantage of algorithmically-generated variables in question statements, answers, hints, or feedback. By using algorithmic variables in questions you can generate numerous variations on a single template question. This style of question generation offers many benefits, including that any changes or corrections that need to be done only have to be made to the template – not to each variation of the question. However, if algorithms are not optimally created, you may not take advantage of all the benefits they offer. This Tips and Techniques article will demonstrate how to write clean, efficient algorithms while avoiding some common pitfalls.

1. **Use Conditions Sparingly**

The **condition** statement provides a convenient way of setting conditions that must be met when the system generates variable values, although, this means that multiple instantiations of the variables may have to be generated before the conditions are finally met. To ensure optimal efficiency, reduce the number of conditions in your algorithm, where possible, by adjusting the variable definitions appropriately.

For example, suppose you wanted two random integers $a and $b such that
$1 \leq \$a < \$b \leq 4$.

Instead of:	*Try:*

`$a = range(4);`
`$b = range(4);`
`condition: lt($a,$b);`
	`$a = range(3);`
	`$b = range($a+1,4);`

For further examples of how to optimize your variable definitions, see the previous Tips and Techniques entitled "Techniques for Creating Algorithmically Generated Questions".

2. **Delete Any Unused Variables**

Suppose you had defined a variable $y, for example `$y = maple("sin($x)");`, and after subsequent changes to your question, $y no longer contributes to any other variable, nor is it used anywhere else in the question. To improve performance, you should remove this line of code from your algorithm.

3. **Invoke Maple Selectively**

Algorithmic variables can be generated by Maple T.A., Maple or both. To define a variable via Maple use the following syntax:

```
   $<variable_name> = maple("<Maple_command>");
```

Using Maple to define variables puts the entire suite of Maple commands at your disposal. Keep in mind, each call to Maple from within Maple T.A. starts a new Maple kernel. To maximize the performance of your question, avoid unnecessary calls to Maple.

By taking advantage of Maple T.A.'s built-in functions for basic range- and list-based variables, and reserving Maple commands for more advanced functionality, your questions will run much faster than they would by relying on Maple calls alone. The examples below will help you decide whether to use Maple or Maple T.A. to create your algorithm.

Use Maple T.A. functions to…	Call the Maple engine to…

Generate a random integer or floating point number in a range.
`$a = rand(0,1);`
`$b = range(-5,5,2);`
`$c = rint(8);`
	Generate a random sequence of numbers that follows a statistical distribution.
	`$L = maple("RandomTools[Generate](list(distribution(Normal(5,3)),10))");`

Define a string or a literal formula (also a string).
`$s = "This is a string.";`
`$a = "sqrt(3)/2";`
`$f = "(x+1)/(x-1)";`
	Define functions and other symbolic objects.
	`$f = maple("proc(x) x^2*exp(x) end proc");`
	`$S = maple("{seq($f(i),i=1..10)}");`

Choose an item from a list, be it the n^{th} indexed item, the n^{th} largest, or the max or min.
`$a = rint(6);`
`$b = rint(6);`
`$m = min($a,$b);`
`$z = rint(4);`
`$x = switch($z,1,-4,-7,10);`
`$ans =`
`switch($z,"odd","even","odd","`
`even");`
	Create a random list, vector or matrix.
	`$n = range(10);`
	`$M = maple("randomize():`
	`LinearAlgebra[RandomMatrix]($n,$n)");`

Randomize specific parameters (e.g., coefficients or exponents) of an expression.
`$n = range(2,10);`
`$f = "x^$n-x";`
`$g = "cos($n*x)";`
	Create random expressions.
	`$f = maple("`
	`RandomTools[Generate](polynom(integer(`
	`range=-5..5),x))”);`

*Render simple expressions in MathML***.**
`$n = range(2,10);`
`$F = mathml("x^$n");`
`$G = mathml("sin($n*x)");`
	Render more complex or randomly generated expressions in MathML.
	`$p = maple("randomize():`
	`randpoly(x,degree=5)");`
	`$P =`
	`maple("MathML[ExportPresentation]($p)"`
	`);`

Perform integer arithmetic.
`$a = rint(5);`
`$b = range(-3,3);`
`$c = ($a+3*$b)^2;`
Reduce a fraction.
`$a = range(8);`
`$b = range(8);`
`$c = frac($a,$b);`
	Perform arithmetic and general symbolic computation on expressions.
	`$f = "x^2-x";`
	`$g = maple("randomize():`
	`randpoly(x,degree=2)");`
	`$h = maple("factor(($f)+($g))");`
	`$a = switch(rint(4),2,3,4,6);`
	`$b = maple("sin(Pi/$a) ");`

Perform basic floating point calculations.
`$a = range(10);`
`$b = decimal(4,sin(pi/$a));`
	Do more sophisticated floating point calculations.
	`$a = range(10);`
	`$b = maple("fsolve(x^3-x+$a,x)");`

Legal Notice: The copyright for this application is owned by Maplesoft. The application is intended to demonstrate the use of Maple to solve a particular problem. It has been made available for product evaluation purposes only and may not be used in any other context without the express permission of Maplesoft.

Thank you for evaluating this Maple application sample www.maplesoft.com