Resources

Here's some links and documentation that might be helpful.

-Basic Training

MTA5 Basic Training.pptx

I've attached slides we use for our basic MapleTA training.

-Take a look at the Maple T.A. User Guide here

<http://www.maplesoft.com/documentation_center/>

They're not the best at updating documentation, but it's a starting point.

-Quick Reference for Students

Maple TA Quick Reference.pdf

Published for students so they know how to enter their answers. This will be updated shortly to add a few more things, but students will know in advance how to enter their answers.

-Algorithmic questions.

Algorithmic Questions.pdf

Take a look at the attached pdf (which is actually 4 documents put together) to start getting an idea of how to efficiently create variables.

-Maple Primes user forum

<http://www.mapleprimes.com/>

They're good at responding and your questions may already be answered there.

-GoogleDocs Area

<http://docs.google.com/leaf?id=0B1LscLsVFJIRMWFiNWJhMDUtYWRiNC00Y2YzLWI3NjItMTFiMjA5MjkxMDg0&hl=en>

We've gathered a few documents here to help us all out. There's a Wishlist of things we wish MTA5 could do or do better. Add your frustrations/suggestions here, and we'll pass them along to Maplesoft.

There's a linear algebra repository of functions we've created to help us make and grade questions involving matrices and vectors. Lots of useful stuff here.

Also take a peek at the Useful Commands document which is a list of all the Maple T.A. commands you can use in algorithms.

And of course, if you have any questions at any time, please ask!

Best Practices

Question Types

When making new questions, we tend to use the Question Designer type and then insert the kind of answer entry field we want, as opposed to making a question strictly/directly numeric or multiple choice or maple-graded or whatever. This gives us more flexibility in the question and makes it easier for us to change our mind with the response field.

Organizing Questions

Put any questions that you create in the group "2 My Questions", sorted by course. Once they are done and ready for testing, move them to the group "3 For Testing". And once they are officially done (I've checked and tagged them), I will move them to the group "4 DONE".

Algorithms

If calling Maple is necessary, make only one Maple call in the algorithmic section.

Creating Questions

When appropriate, clone the original question and modify the clone, and not the original, so that tags, corrections, etc. are copied and so that the original question isn’t broken.

Feedback

Include complete solutions/feedback (in the Feedback section) for every question. Only questions with good feedback will be added to the repositories.

E.g., include complete row reductions, the EEA, explanations, detailed steps, etc. wherever possible.

Equation Editor

Only use the equation editor to insert mathematics when necessary (when you can’t type the symbol you need). Otherwise, try to insert/format the equation directly inline.

Use the symbol for congruent, instead of spelling out the phrase "x is congruent to 1 (mod 4)". Use the equation editor and display this information mathematically in the question.

You can’t insert a MathML expression (that you generated in the algorithm or returned from Maple in the algorithm) into an equation editor box. It won’t display. And you already have the MathML version of what you want to display, so you don’t need the equation editor, just use the variable name directly.

Minimize the length of expressions in one equation editor box. Split the feedback up into separate expressions and lines and use separate equation editors boxes for each one. This makes initial and future editing easier.

Testing

Here are the things that you should be thinking about when testing your questions in Maple T.A. (besides checking that the right answer works and that a wrong answer is marked as incorrect and the appropriate marks are assigned).

1. Could your question be perceived as ambiguous to the student?

2. Is the question free of bugs and spelling errors?

3. Will the student know how to properly format their answer?

4. Does the question use the same format that is used in the students textbook?

5. Does the math in the question look nice? (Like it came from a textbook.)

6. If MC, are all possible answer choices unique?

7. If UE, does the question accept different notations for the correct answer?

8. Have the variables been generated efficiently to avoid unneccesary condition statements?

9. Have you minimized the amount of times Maple is called?

10. Has appropriate feedback been added to the question? Is it correct?

11. Have you tested the question SO much that you never, ever want to see it again? :)

Happy testing!

Watch for errors, typos, broken variables, incomplete sentences and incorrect feedback.

Please read the solutions in detail, work out the answers to the questions, and make sure the solutions are complete and correct. And when you are done, have the other double-check your work.

Examples

I've copied over some Demo questions that Maplesoft provides (see the group "1 Demo Class Questions") and a few examples of some more advanced questions that our developers have written that have complicated algorithms or that call Maple or use a Maple repository (see the group "2 My Questions").

I've copied 4 questions into each of your courses, so that you can use them as examples to help you write more complicated questions.

01. Gram-Schmidt on 3 vectors in R3

-Shows how to run a bunch of Maple commands with one call to the Maple kernel, and how to return and switch through a list of MathML and other objects/numbers to create the question and the feedback.

02. Dimension of Null space

-Shows how to make a MC question by making the answer and the alternates be variables in the algorithmic section. Make sure that none of your alternates are also a correct answer. Sometimes this requires you to narrow your range of choices for some of the variables.

03b. Find eigenvalues [0 1 1; 1 0 1; 1 1 0]

-Grades a list of answers. Code might only work for integers. Check that it also works for complex numbers, decimals, and maybe other expressions that use Pi, etc.

13. Ordering stuff

-Makes a pool of questions behind the scenes. Picks an index and the associated elements from a set of ordered lists. In this case the lists are strings of nouns, but the lists could be functions, sentences, etc. Sometimes this is the only way we can make questions algorithmic.

Algorithms

How to generate values for the algorithmic variables? Come up with a scheme of how to limit which numbers you choose and pass it by your ISC. Whatever you feel is a good way to make the question algorithmic yet still be able to do the same feedback for each version.

For MC questions, ensure through the use/calculation of your variables that none of your alternates are also a correct answer.

When checking questions that others wrote, take a look at their algorithms and see if they can be improved.

When checking over questions that others wrote, please see if you can improve the algorithms. Sometimes the question returns many variables that never get used, which slows down the generation of the question. We need to make our algorithms that call Maple as efficient as possible.

Calling Maple isn’t always necessary. See if you can write the algorithm without calling Maple.

Conditions

Here's a good example of how we can write better algorithms.

The original code:

$a=range(3,15,2);

$asq=$a\*$a;

$p=switch(rint(7),7,11,13,17,19,23,29);

condition:gt($p,$a);

Note that using a condition is slow/expensive. The condition isn't checked until the end. If the set of variables chosen doesn't satisfy the condition, then all of the values are re-generated. Slow.

Re-write the code this way:

$p=switch(rint(7),7,11,13,17,19,23,29);

$a=range(3,$p-1,2);

$asq=$a\*$a;

No condition necessary.

Keep this in mind when writing your algorithms. Avoid using the condition statement if you can, and If you must use it, put it as early as possible in the algorithm.

Technical Details

mod in Maple T.A.

For some reason MapleTA doesn't like when you use mod in the form "mod(p,q)",
but it will accept "p mod q".

Formatting

Variables should be italic and numbers should be in normal text. Fix the MathML by adding or removing mathvariant='normal' as appropriate.

Make the mod's not bold (remove tags from the MathML).