Authoring Numerical Problems in LON-CAPA: Step-by-Step

5/22/14 M. Lucas (lucasm@ohio.edu or LON-CAPA Username lucas:ohiou)

Version 0.2

Overview

The goal of this document is to discuss the coding of numerical problems in LON-CAPA. In particular this tutorial will walk through the creation of a numerical response problem in steps.

I will presume that you already know or will learn somewhere else how to get into construction space and get into the EditXML mode. I will also presume that you will work primarily in the EditXML mode (basically a text window) rather than using the 'colorful editor' (selected by the Edit button).

A Numerical Problem

Let's start with a standard numerical problem. A little story and calculation are given to the student, and they have to calculate an answer. When they enter their answer the computer grades it, returning either the 'green box of elation' or the 'pink box of doom' (or perhaps the 'yellow box of confusion').

First we'll define the problem, and then we'll add bit by bit to make it a little more sophisticated.

Defining the Problem

We start with a standard introductory physics problem involving Newton's Second Law. We want to give the student a block of a given mass and ask for the acceleration given the force. Not bad. Let's see what we can do with it.

The following is a pretty good outline to consider when thinking about this type of problem:

- We have to pick values for the variables and calculate any answers.
- We have to lay out the problem text for the student.
- We need to tell the computer what kind of answer to expect and how to grade it.
- We can also tell the computer what kind of hints we'd like to give.

Creating a Problem

Problems are coded with tags. These tags tell the computer how to handle the information in the problem file. A tag is bracketed by the less-than and greater-than signs. This is the same way that HTML (HyperText Markup Language) works for web pages. In fact many HTML tags can be used to provide extra formatting for a problem. They just get passed right through to the browser.

LON-CAPA uses XML, eXtensible Markup Language, which in essence just means that LON-CAPA makes up it's own tags as well.

One last rule: any begin tag needs an associated end tag. For example, the `<problem>` tag that tells LON-CAPA this is the beginning of the problem needs to be paired with an end tag `</problem>` that tells LON-CAPA it is at the end of the problem. Some tags don't have any text in between their begin and end tags, so instead of saying `<startouttext></startouttext>` you can use the shorthand `<startouttext/>`.

So here is what our problem might look like:

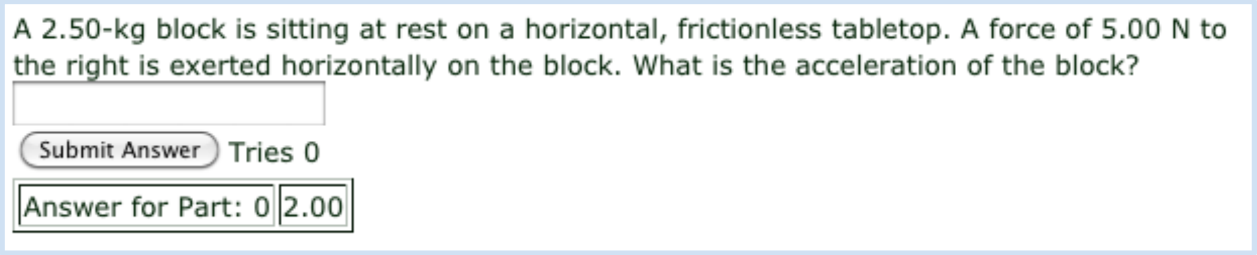
```
<problem>
<startouttext/>
A 2.50-kg block is sitting at rest on a horizontal,
frictionless tabletop. A force of 5.00 N to the right
is exerted horizontally on the block.
What is the acceleration of the block?
<endouttext/>
<numericalresponse answer=”2.00” id=”11”>
   <textline/>
</numericalresponse>
</problem>
```

The `<startouttext/>` tag tells LON-CAPA this is a block of text to be displayed. For technical reasons, the block of text ends with `<endouttext/>` (Note: these are two different tags, not the begin and end of a single tag.)

Once the problem is defined, LON-CAPA needs to know what kind of answer to expect. In this case we use a `<numericalresponse>` tag. Inside the tag we tell it that the answer to expect is 2.00 and that the id for this particular response is 11. The first part should be pretty self-explanatory, but the second may need some explaining. Any submission that a problem takes in needs a unique id. As well any part also needs a unique identifier. These really can be any string. If the id is not provided, LON-CAPA will provide an id when the problem is published.

Another detail is the `<textline/>` tag which tells the computer the kind of box to put up to accept the answer.

There you have it, one complete LON-CAPA problem! The problem in construction space looks like the following:



A 2.50-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 5.00 N to the right is exerted horizontally on the block. What is the acceleration of the block?

Answer for Part: 0 | 2.00

Adding Randomization

… but, as you can guess, there are some issues here. Sure, this would work, but the student needs to put in 2.00 (2 or 2.0 or 2.00000 would also work, but not 2.0001). Secondly, the news spreads like wildfire! “Just put in 2.00 and let’s go do something that’s more fun!”

So how does one make the problem a little more robust with some randomization? Choosing random variables and calculating answers is done using a scripting language called Perl. One does not need to be a Perl expert to code LON-CAPA problems. A few simple rules and a couple functions get you started. Any variable begins with a dollar sign, $; a function begins with an ampersand, &; and don’t forget the pesky semi-colons at the end of each line.

For this specific problem we need to do the following: pick a mass, pick a force and calculate the acceleration. We can then just stick the Perl variables in the text (though we’ll be more careful about that in a second).

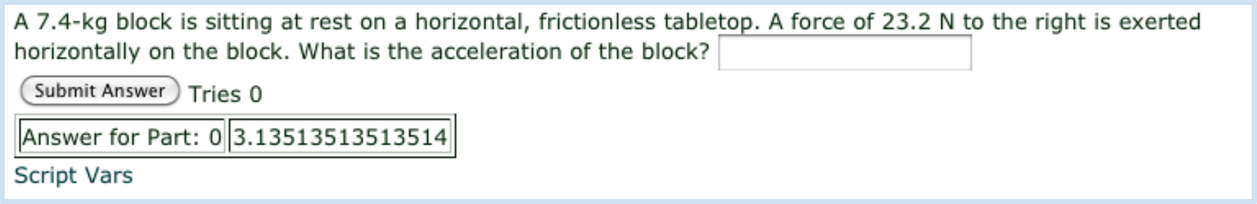
```
<problem>
<script type=”loncapa/perl”>
# Pick mass and force, calculate the acceleration
$mass = &random(2.00,8.00,0.1);
$force = &random(11,25,0.1);
$acceleration = $force/$mass;
</script>
<startouttext/>
A $mass-kg block is sitting at rest on a horizontal,
frictionless tabletop. A force of $force N to the right
is exerted horizontally on the block.
What is the acceleration of the block?
<endouttext/>
<numericalresponse answer=”$acceleration” id=”11”>
   <textline/>
</numericalresponse>
</problem>
```

Here’s the rundown of what we just did:

- A `<script>` tag was inserted to allow for a chunk of Perl to be evaluated in the problem.
- I snuck in a comment (#) inside the script just because sometimes commenting your code is a good thing. Here I used real words for variable names, so hopefully it is pretty self-explanatory what is going on.
- The `&random(low,high,delta)` function returns a random number between low and high with a resolution of delta (if delta is 0.1, it will round to the nearest tenth’s digit).
- Using `&random()` we select a reasonable mass and a reasonable force.
- We then calculate the acceleration.

- Lastly, we stick the appropriate variables in the problem text and the `<numericalresponse>` tag.

Here's the resulting problem:



A 7.4-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 23.2 N to the right is exerted horizontally on the block. What is the acceleration of the block?

Tries 0

Answer for Part: 0 | 3.13513513513514

Making It More Meaningful

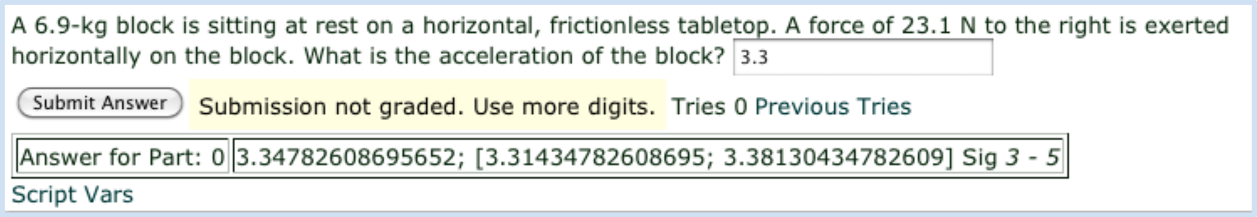
That's not too bad, but there are some issues. The major concern is that a student currently needs to enter exactly 3.13513513513514 (if given the random numbers above) to get the answer correct. The other is that we have not requested that the student enter units yet.

A <numericalresponse> tag can take problem parameters. The two we commonly use are Significant Digits (sig) and Numerical Tolerance (tol). These two additional lines are lines that I often just copy and paste from one problem to another so that I get the syntax right.

```
<numericalresponse answer="$acceleration" id="11">
    <responseparam name="sig" type="int_range"
             description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
             description="Numerical Tolerance" default="1%" />
   <textline />
</numericalresponse>
```

The first response parameter sets the number of significant digits to a range between 3 and 5. While you can be very picky about significant digits, if you're looking for the exact application of significant digits, you need to be very careful about the coding of the problem and the display of the variables (note that the mass in the above example only has 2 significant digits).

Quite honestly, we use sig to make sure that students enter at least 3 digits (so that we're sure they haven't rounded the correct answer too much) and cap it at 5 so they don't get carried away. If they enter too few digits LON-CAPA presents what one could call the 'Yellow Box of Confusion' and doesn't even check the answer. Likewise, it does not subtract a try.



A 6.9-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 23.1 N to the right is exerted horizontally on the block. What is the acceleration of the block? 3.3

Submission not graded. Use more digits. Tries 0 Previous Tries

Answer for Part: 0 | 3.34782608695652; [3.31434782608695; 3.38130434782609] Sig *3 - 5*

The second parameter imposes a tolerance of +/-1% on the answer. Notice that the computer will accept any number between 3.3143 and 3.3813. Choosing an appropriate tolerance can be an art form. If the tolerance is too small students with the right answer may not get the answer right because of rounding, and if the tolerance is too large, students may be more likely to get the right answer just by guessing. 1% to 2% is generally pretty reasonable for these standard problems. If the problem involves the subtraction of two numbers which are close together, you might use a larger tolerance and/or provide more significant digits in the display of the initial variables. If the problem involves reading numbers off a graph or making calculations using numbers off a graph, you might also provide a larger tolerance (and then make sure students who make the common errors can't get the answer accidentally).

If an author has chosen a number of significant digits or a numerical tolerance which you don't like, you can override them in the course by choosing new parameters. Just exercise reasonable judgment as the author may have had a good reason for choosing the parameters as they did.

Units

LON-CAPA does a pretty good job of handling units. Here we can place the units for the answer in the `<numericalresponse>` tag directly:

```
<numericalresponse answer="$acceleration" unit="m/s^2" id="11">
```

LON-CAPA now knows that the answer provided is in m/s^2. If the student does not enter units that are dimensionally consistent (length/time squared), they will get the 'Yellow box of confusion' and not be docked a try. If they DO enter units that are dimensionally correct, for example `'mi/hr^2',` the computer will convert their answer from mi/hr^2 into `m/s`^{2} and then check the answer. Generally this goes in the students' favor, but in the situation where they calculate the right number for m/s^2 but enter cm/s^2, they will lose a try and generally have no idea why they have it wrong.

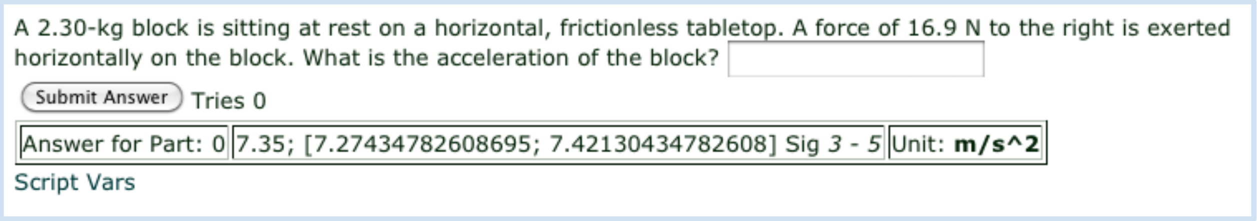
Making It Prettier

So there are some tags that can help us clean things up a bit, and this isn't always just about aesthetics. Notice that the first number only shows as two significant digits. In fact if the mass was 5.00 it would only show as 5. We can fix that by using the `<num>` tag, which provides formatting for numbers. In this case we use a format of `3s` for both variables and then also stick a `3s` in the `<numericalresponse>` tag as a format parameter. In each of these cases, the number will be presented with 3 significant digits. 5 will show up as 5.00, 54 will show up as 54.0, and 3.347826 will show up as 3.35.

Having done this there will be less confusion on the part of the student as to the precision provided in the problem.

```
<startouttext />
A <num format="3s">$mass</num>-kg block is sitting at rest on
a horizontal, frictionless tabletop.
A force of <num format="3s">$force</num> N to the right
is exerted horizontally on the block.
What is the acceleration of the block?
<endouttext />
<numericalresponse answer="$acceleration" unit="m/s^2"
format="3s" id="11">
```

Here's how the problem looks now:



A 2.30-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 16.9 N to the right is exerted horizontally on the block. What is the acceleration of the block?

Answer for Part: 0	7.35; [7.27434782608695; 7.42130434782608] Sig *3 - 5*	Unit: **m/s^2**

Other formatting options include 2f (floating point with two digits after the decimal point) and 2e (scientific or exponential notation with two digits after the decimal point). Just to be clear, you can use any number within reason. 5e would give 5 digits after the decimal point.

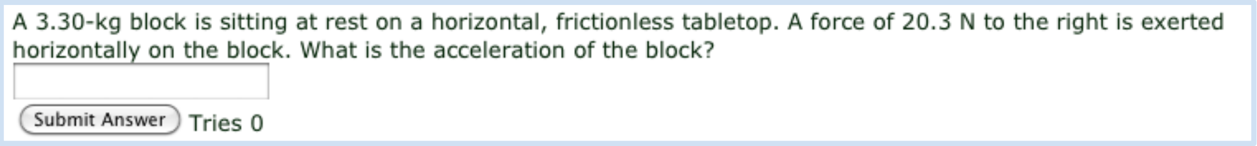
There are also output tags for algebraic expressions, chemical formulas, and mathematical expressions. Check out the Output Tags link above the text box in construction space for a more complete list of tags.

A Few More Formatting Issues

Okay, so sometimes it is about aesthetics. Say you don't like the box showing up right by the end of the sentence. You can insert a line break using the standard HTML (web page) line break tag <br/>. This will force a line break within your text.

```
What is the acceleration of the block?
<br/>
<endouttext />
```

Here's the problem:



A 3.30-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 20.3 N to the right is exerted horizontally on the block. What is the acceleration of the block?

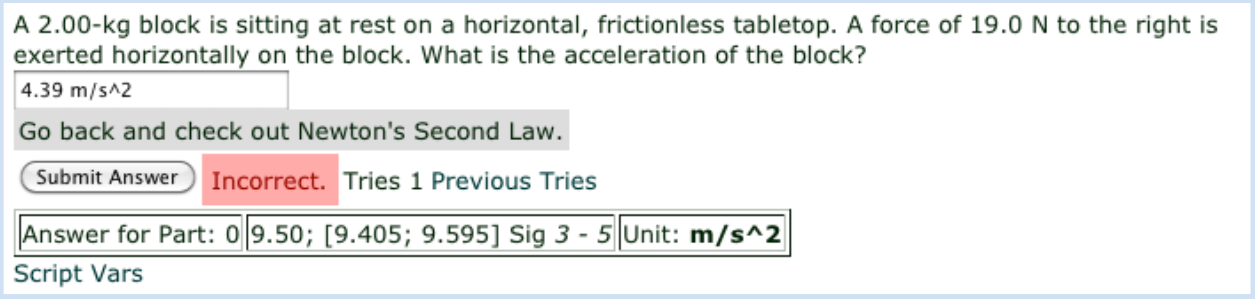
Providing Feedback

How about providing a little bit of assistance? If they get the answer wrong, you can provide a hint. Hints can either be provided as a general hint or as a conditional hint that is only given if a particular answer is entered. So let's give a generic hint to go back and read up on Newton's Second Law if they get the wrong answer.

```
<numericalresponse answer="$acceleration" unit="m/s^2"
format="3s" id="11">
    <responseparam name="sig" type="int_range"
description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
description="Numerical Tolerance" default="1%" />
   <textline readonly="no" />
<hintgroup showoncorrect="no">
    <startouttext />Go back and check out Newton's Second
Law.<endouttext />
</hintgroup>
```

```
</numericalresponse>
```

Inside the <numericalresponse> tag there is now a <hintgroup>. If the answer is wrong, the text within the <hintgroup> is displayed. Here's what it would look like:



A 2.00-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 19.0 N to the right is exerted horizontally on the block. What is the acceleration of the block?

4.39 m/s^2

Go back and check out Newton's Second Law.

Incorrect. Tries 1 Previous Tries

Answer for Part: 0	9.50; [9.405; 9.595] Sig *3* - *5*	Unit: **m/s^2**

Suppose we knew that a common mistake is to use $force*$mass instead of $force/$mass. We could create what is called a conditional hint. This requires three steps:

- we need to create a variable containing the appropriate wrong answer for this version of the problem,
- we need to create a response to see if the student entered this particular answer,
- and we need to provide a hint that is only shown if the hintresponse condition was true.

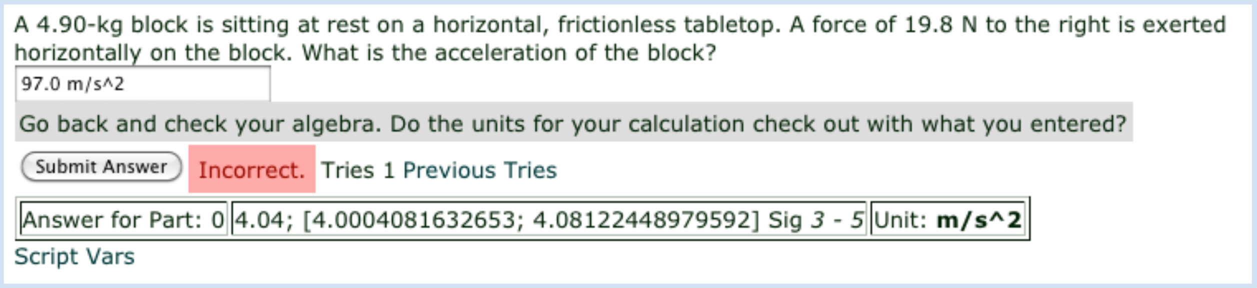
The code looks like this:

```
<problem>
<script type="loncapa/perl">
# Pick mass and force, calculate the acceleration
$mass = &random(2.00,8.00,0.1);
$force = &random(11,25,0.1);
$acceleration = $force/$mass;
# Calculate common wrong answer
$multiplied = $force*$mass;
</script>
<startouttext />
A <num format="3s">$mass</num>-kg block is sitting at rest
 on a horizontal, frictionless tabletop.
A force of <num format="3s">$force</num> N to the right is
exerted horizontally on the block.
What is the acceleration of the block?
<br/>
<endouttext />
<numericalresponse answer="$acceleration" unit="m/s^2"
format="3s" id="11">
    <responseparam name="sig" type="int_range"
            description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
           description="Numerical Tolerance" default="1%" />
   <textline readonly="no" />
```

```
<hintgroup showoncorrect="no">
<numericalhint name="multiplied" answer="$multiplied"
                           unit="m/s^2" format="3s" id="12">
    <responseparam name="sig" type="int_range"
             description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
             description="Numerical Tolerance" default="1%" />
</numericalhint>
<hintpart on="multiplied">
    <startouttext />
Go back and check your algebra. Do
the units for your calculation check out with what you entered?
    <endouttext />
</hintpart>
</hintgroup>
</numericalresponse>
</problem>
```

Notice that you basically need to define the whole answer over again in the <numericalhint> tag, including a unique id.

The problem now looks like this:



A 4.90-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 19.8 N to the right is exerted horizontally on the block. What is the acceleration of the block?

97.0 m/s^2

Go back and check your algebra. Do the units for your calculation check out with what you entered?

Incorrect. Tries 1 Previous Tries

Answer for Part: 0	4.04; [4.0004081632653; 4.08122448979592] Sig *3 - 5*	Unit: **m/s^2**

One More Tweak

Okay, this is it for now. One more tweak and we're done. It's not a small tweak, but something you may want to do often. Now that they have dutifully calculated the acceleration, wouldn't it be great if they also could tell you the distance the block travels in a certain time period?

We want to make a multipart problem. To do this we'll wrap each part in a `<part>` tag and provide an appropriate id (any text will do, but we'll use part1 and part2). Each `<numericalresponse>` tag will be in its own part, so the students can answer each part independently. Each part also gets points depending on the weight parameter (default of one point per part).

Before we do this we need to think carefully about what we wrap and what we don't. We're going to create this great two-part problem, but a colleague of yours may look at this and only want to use part1, or conversely only want to use part2, which makes it a little more challenging for the student to solve the problem (it becomes a two-step problem without a check point in the middle leading them to the answer).

Your colleague (or you) can actually disable a particular part in a problem under the course parameters for the resource. A well-designed problem should work with either part not included. We need to make sure all the calculations are still done, so we will NOT put these inside the <part> tags. We also need to make sure all the needed text for part2 shows up independent of part1, so the basic declaration of the problem needs to be outside the <part> tags as well.

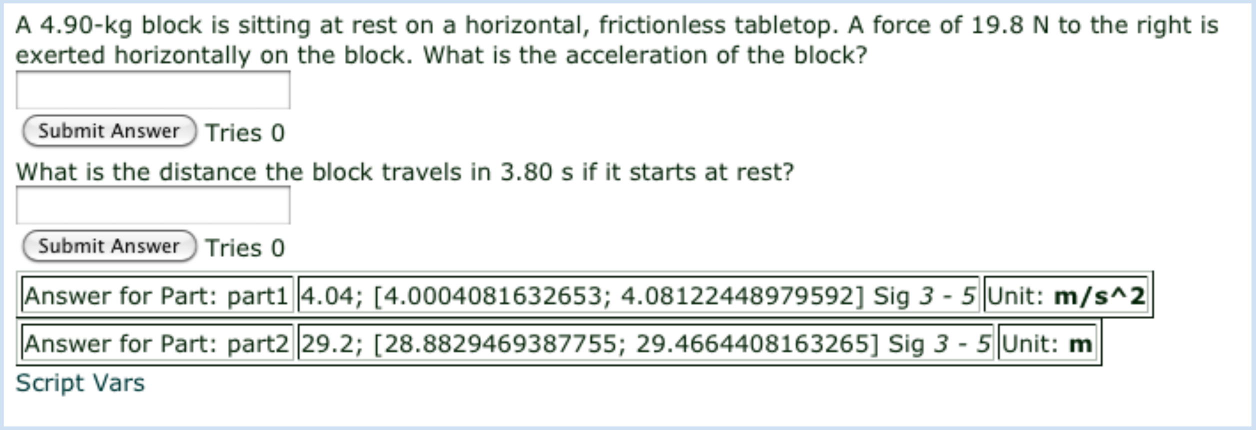
Here's the complete problem:

```
<problem>
<script type="loncapa/perl">
# Pick mass and force, calculate the acceleration
$mass = &random(2.00,8.00,0.1);
$force = &random(11,25,0.1);
$acceleration = $force/$mass;

$time = &random(3.00,5.00,0.01);
$distance = 0.5*$acceleration*$time*$time;
# I could also use ($time^2) or &pow($time,2)

# Calculate common wrong answer
$multiplied = $force*$mass;
</script>
<startouttext />
A <num format="3s">$mass</num>-kg block is sitting at
rest on a horizontal, frictionless tabletop.
A force of <num format="3s">$force</num> N to the right
is exerted horizontally on the block.
<endouttext/>
<part id="part1" display="Part 1">
<startouttext/>
What is the acceleration of the block?
<br/>
<endouttext />
<numericalresponse answer="$acceleration" unit="m/s^2"
format="3s" id="11">
    <responseparam name="sig" type="int_range”
            description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
            description="Numerical Tolerance" default="1%" />
   <textline readonly="no" />
<hintgroup showoncorrect="no">
<numericalhint name="multiplied" answer="$multiplied"
                              unit="m/s^2" format="3s" id="12">
    <responseparam name="sig" type="int_range"
               description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
              description="Numerical Tolerance" default="1%" />
</numericalhint>
```

```
<hintpart on="multiplied">
    <startouttext />
Go back and check your algebra. Do the units for your
calculation check out with what you entered?
    <endouttext />
</hintpart>
</hintgroup>
</numericalresponse>
</part>
<part id="part2" display="Part 2">
<startouttext/>
What is the distance the block travels in
<num format="3s">$time</num> s if it starts at rest?
<br/>
<endouttext />
<numericalresponse answer="$distance" unit="m" format="3s"
                                                       id="21">
    <responseparam name="sig" type="int_range"
               description="Significant Digits" default="3,5" />
    <responseparam name="tol" type="tolerance"
               description="Numerical Tolerance" default="1%" />
   <textline readonly="no" />
</numericalresponse>
</part>
</problem>
```

Okay, it looks a little long and intimidating, but we've built it up bit by bit. The second part uses some text to display the question and a `<numericalresponse>` tag. There are no hints coded for the second, but we could if we wanted to.

The problem now looks like this:



A 4.90-kg block is sitting at rest on a horizontal, frictionless tabletop. A force of 19.8 N to the right is exerted horizontally on the block. What is the acceleration of the block?

Submit Answer Tries 0

What is the distance the block travels in 3.80 s if it starts at rest?

Submit Answer Tries 0

Answer for Part: part1	4.04; [4.0004081632653; 4.08122448979592] Sig *3 - 5*	Unit: **m/s^2**
Answer for Part: part2	29.2; [28.8829469387755; 29.4664408163265] Sig *3 - 5*	Unit: **m**

Script Vars

Once you've dissected a problem like this, it can become a template for a wide variety of numerical problems. You need to change the calculations and the text, but the structure is very similar for a wide variety of problems.