Who wants to be the weakest millionaire?

TV game shows like Who Wants To Be A Millionaire? and The Weakest Link are very popular. Part of their popularity lies in the fact that they have interesting rules for the contestants to play against. These rules give the shows their tension, but someone has to make the rules up to begin with. So how would you go about designing the format for a good game show?

Cash in a box

Well, we need some prize money – the amount of cash that the player will walk off with in their pocket. Lets add a prop, a shiny box that our host puts the money into, and lets label this box with big letters ‘PRIZE’. So, a very simple game show would ask the contestant a question. If they get it right, they win £100, for example, which our host puts into the ‘PRIZE’ box. At the end, the player gets what is in the box, so ‘PRIZE’ can refer to the box but also, more importantly, the value of money in the box.

Simply answer the question

How could we write the instructions for this simple quiz? Well how about:

```
If (Answer is correct)
   {PRIZE = £100}
```

We have used PRIZE = £100 to mean put £100 in the prize box. So, as we wanted, if the answer’s right, the value in PRIZE goes up to £100.

Double your money!

Simple enough, now suppose that we make the game a little more challenging, so what will happen is that each time the player gets the answer correct they double their money. If they get the answer wrong then they lose the lot (a bit like Who Wants To Be A Millionaire?). So, how would we write that?

Your starter for 100?

We have a problem to begin with. If there is no money in PRIZE to start with, then doubling it when you get the answer right will give you twice as much nothing. Hardly fun to watch. So we have to put some cash in the box to begin, (or we give them a simple question to start with). Let’s be kind. Let’s put £100 in the box to start.

```
PRIZE = £100
```

If they get the question right we can then say that PRIZE = 2 * PRIZE. What this means is that the new money in the box will be twice the money that was in the box beforehand (we’ve used * for multiplication).

Twice in the box

We have our starting situation (we need something to double remember)

```
PRIZE = £100
```

And if the player gets the question right

```
If (Answer is correct)
   {PRIZE = 2*PRIZE}
```

But we were being nasty. What happens when they get the question wrong? They need to lose all the cash! That is we want to set PRIZE=£0: no cash in the box. How could we write this?

We could simply write

```
If (Answer is wrong)
   {PRIZE = £0}
```

After all the answer is either right or else it’s wrong, and here’s an idea about how to write this:

```
If (Answer is correct)
   {PRIZE = 2*PRIZE}
else
   {PRIZE = £0}
```

We have a single line. ‘If the answer is correct, the cash doubles or else (if the answer is wrong, which is the only other option) the cash is lost.’

So for our Who Wants To Be A Millionaire?-type quiz, we have the rules.

```
PRIZE = £100

If (Answer is correct)
   {PRIZE = 2*PRIZE}
else
   {PRIZE = £0}
```

Round and round again

But we want more than one round of the game. Each additional round should become more exciting, as the prize money grows with each correct answer. How do we write this? Well we need some way to say that we do the same thing time and again, as long as we are happy to do it, but that at some stage we want to stop. Hmm. How many rounds do we want to have?

Quick quiz question

Why did a 13-year old girl from Brittan Elementary School in the USA make headlines across the world for saying: ‘Look at this. I’m a grocery item. I’m a piece of meat. I’m an orange.’

Answer

She was outraged that her school had introduced electronic tags to keep a constant track of student movements within the school. As a result, the school suspended its use of the system, which just goes to show there is more to computer science than technical brilliance. To be a successful innovator, you have to understand people too.

Count up

Let's do our quiz round eight times:

```
Do 8 times {a round}
```

So we have a way to have eight rounds, and we know how to do each of the rounds, and how to start the prize fund in the box, so, let's put them together:

```
PRIZE = £100

Do 8 times
{
        If (Answer is correct)
           {PRIZE = 2*PRIZE}
        else
           {PRIZE = £0}
}
```

Does this make sense? We start with setting the contents in the prize box to £100 before we do any rounds, so that first 'PRIZE = £100' will go outside the 'Do it eight times' bit. For each round if the answer is correct the prize money doubles, and as it's the same PRIZE box we use in each round, the money will continue to double for our lucky player.

That's the wrong answer!

What happens if the contestant gets the answer wrong? Well, looking at our 'else' rule, if in any round the answer is wrong, the prize in the box goes to zero. And that's a problem for our player because even if they get the next question right it would just double nothing! Unkind perhaps but would watching the contestants play for, say, five rounds, answering the questions for nothing, make good television?

Therefore, if a contestant gives a wrong answer, we want to stop the quiz and take away the prize money. The game will be over. So, let's add that to our set of rules. Put in the word 'Break' to mean just that: that we jump out of the rounds and end the game.

```
PRIZE = £100

Do 8 times
{
        If (Answer is correct)
           {PRIZE = 2*PRIZE}
        else
           {PRIZE = £0, Break}
}
```

Roll the credits

Well done. So there we have the rules for a 'Millionaire'-type game, explained more-or-less in English and fairly easy to write down. But if you've followed this through, you've actually understood your first computer program. Computer programs look just like this, a series of rules to control the way numbers (or, in our case, cash) are moved around. We call this set of rules an algorithm, the way that we write the instructions is called syntax. And the rest of computer programming? That's just practice.