**Solving Linear Equations**

The following are all examples of linear equations,

3x - 7 = 2

7x + 2 = 3x + 18

3x = 18

All of the above equations can be solved - you can work out a single value for **x** that satisfies the equation.

**Abstraction/Generalisation**

If we are to write a program that can solve a linear equation for us, we need to find a way to describe the steps that will allow us to solve all linear equations. We also need to consider the information that is most important to us.

A generalisation of a linear equation might be written as follows,

**a**x + **b** = **c**x + **d**

This format describes all of the equations of this type (if there are brackets or division lines, an extra step of working is needed to put the equation into that form). The values represented by the letters **a - d** are the important ones. In order to understand how our program works, we need to be able to map an equation onto this system.

Have a go at filling in the missing values in the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Equation** | **a** | **b** | **c** | **d** |
| x + 9 = 21 | 1 | 9 | 0 | 21 |
| x + 7 = 20 |  |  |  |  |
| 5x = 50 | 5 | 0 | 0 | 50 |
| 2x + 6 = 0 | 2 | 6 | 0 | 0 |
| 2x + 4 = 10 |  |  |  |  |
| 57 = 7x + 8 | 0 | 57 | 7 | 8 |
| 3 - x = 0 | -1 | 3 | 0 | 0 |
| 31 = 5x + 6 |  |  |  |  |
| 3x + 3 = 5x - 4 | 3 | 3 | 5 | -4 |
| 7x + 2 = 3x + 18 |  |  |  |  |

**Rule**

We can solve the equation if either a or c is **not** 0. Negative values and decimals are fine.

**Solving**

First, a couple of examples of how we might solve some equations.

**57 = 7x + 8**

7x + 8 = 57 *swap* *the sides of the equation around to put the largest number of x's on the left*

7x + 8 - 8 = 57 - 8 *subtract 8 from both sides of the equation*

7x = 49

x = 7 *divide both sides of the equation by 7*

**3x + 4 = 5x - 4**5x - 4 = 3x + 4 *swap* *the sides of the equation around to put the largest number of x's on the left*
5x - 4 + 4 = 3x + 4 + 4 *add 4 to both sides*
5x = 3x + 8
5x - 3x = 3x - 3x + 8 *subtract 3x from both sides*
2x = 8
x = 4 *divide both sides by 2*

Let's go back to the general equation we had earlier,

**a**x + **b** = **c**x + **d**

When we solved the second equation, we had the following values for a - d,

a = 3, b = 4, c = 5, d = -4

Our first step was to swap the equation around so that the larger number of x's would be on the left hand side. We swapped a with c and b with d. What we did could be described as follows in pseudocode,

**IF a<c Then**

**tmp🡨 a**

**a 🡨 c**

**c 🡨 tmp**

**tmp 🡨 b**

**b 🡨 d**

**d 🡨 tmp**

**End If**

We always need to use a third variable to swap the values stored in two variables. Once we change the value of a, we no longer have its original value to assign to the other variable.

The next thing we did was to add 4 to both sides of the equation. This was a crucial step - it removed the numbers from the left hand side of the equation leaving only x's. What we actually did was subtract -4 from b and -4 from d. Subtracting a negative number moves you to the right on the number line. In pseudocode,

**d = d - b**

**b = 0**

**(The value stored in b is no longer relevant to the solution, you can leave out the b = 0)**

The process we carried out with the constants (b and d), we now repeat with the coefficients (a and c). This gets rid of the x's on the right hand side of the equation. In the example we subtracted 3x from both sides of the equation. That means we subtracted 3 from both a and c.

**a = a - c**

**c = 0**

**(At this point, the value stored in c is no longer relevant, the second line can be left out)**

The last step was to divide both sides by 2. We divided d by a.

**d = d / a**

**a = 1**

At this point, the solution to our equation is stored in the variable d. Our steps were,

|  |  |
| --- | --- |
| **Step** | **Pseudocode** |
| Get user input | Declare a,b,c,d as doubleInput a,b,c,dOutput "{0}x + {1} ={2}x + {3}",a,b,c,d |
| Swap the sides of the equation to put the largest number of x's on the left hand side. | IF a<c Thentmp🡨 aa 🡨 cc 🡨 tmptmp 🡨 bb 🡨 dd 🡨 tmpOutput "Swap sides of the equation to put largest x on the left"Output "{0}x + {1} ={2}x + {3}",a,b,c,dEnd If |
| Get rid of numbers on the left hand side of the equation. | IF b<>0 Thend 🡨d - bOutput "Subtract {0} from both sides of the equation.",bb 🡨 0Output "{0}x={1}x + {2}",a,c,dEnd IF |
| Remove x from the right hand side of the equation | IF c<>0 Thena 🡨 a - cOutput "Subtract {0}x from both sides of the equation.",cOutput "{0}x={1}",a,dc 🡨 0End IF |
| Divide by number of x to get answer. | IF a<>1 thend 🡨 d / aOutput "Divide both sides of the equation by {0}.",aOutput "x = {0}",da 🡨 1End IF |

This isn't perfect but it gives you a solution to the equation and describes the steps being taken. Place a Console.Readline() between each section and you can use this as a support for solving equations like this in Maths.