

3.1

Use variables

In this lesson

You will learn:

- how to plan a program that uses variables and start to make the program.

The requirement

Before you plan a program, you must decide what the program requirement is. In this unit you will create a program to meet this requirement:

Make a program for a student so they can practise the seven times table.

Program plan

Before a programmer starts work, they make a **program plan**.

A program plan sets out the steps to solve a problem. Another word for a plan to solve a problem is an **algorithm**.

A program plan sets out the inputs and outputs of a program. It also sets out the processes done by the computer, such as calculations.

Here is a plan for this program:

Input: Prompt with a question (for example, 'What is 7 times 5?').

Input the user's 'answer'.

Process: Store the 'solution' to the question.

Compare the user's answer to the solution. Do they match?

Output: If the answer matches the solution, output 'You got it right!'

Else, output 'You got it wrong!'

The word 'else' is used in programming. In this example, the computer will output 'You got it wrong!' if the answer does not match the solution.

Variables

A **variable** stores a value. You give every variable a name. Then you can use the value in your program. Two variables are mentioned in the program plan.

In Scratch there is a ready-made variable called 'answer'. You can make a sprite ask a question. The user's answer is stored in 'answer'.

Spiral back



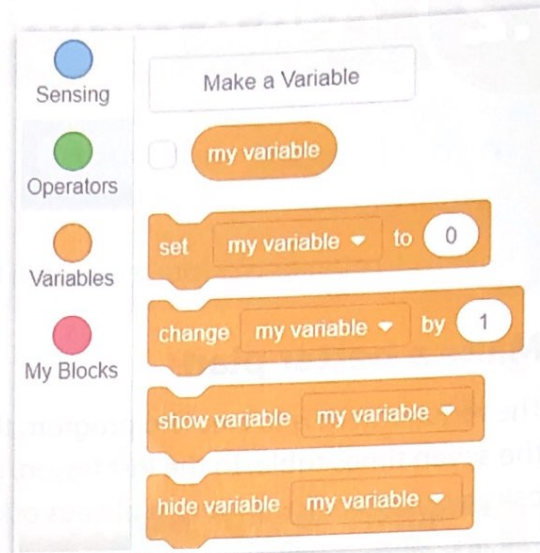
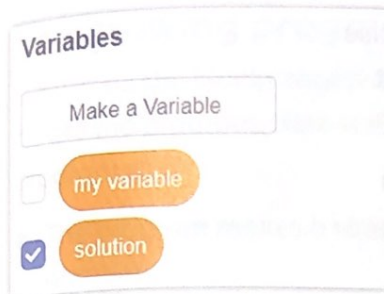
Last year, you learned how to make and use variables in Scratch programs. If you need help with making variables in this lesson, look back at the lessons in Student Book 4.

You will also make a new variable called 'solution'. This variable will store the correct solution to the question.

Click on 'Make a Variable'.

Type the variable name 'solution'.

The variable name is ticked. That means it will show on the main screen. Click to remove the tick. You don't want the user to see the solution.



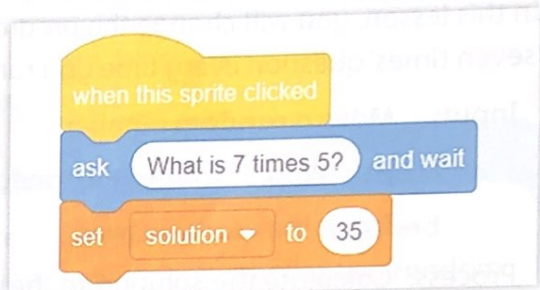
Scratch program

The finished Scratch program is shown opposite.

This program matches part of the program plan.

- It asks a question.
- It stores a solution.

In the next lesson you will extend the program to match the whole plan.



Activity

Start Scratch. Create the program shown in this lesson. You can use any 'seven times' question.

Click on the sprite to run the program. The sprite asks a question. There is a box to enter your answer.

Save the program. Open the 'File' menu and select 'Save to your computer'.

Think again

The variables in this program are called 'solution' and 'answer'. Why are these good variable names?

Think of alternative variable names which would be just as good.



Extra challenge

Extend the program to tell the user if their answer is right or wrong.

3.2

Ask a random question

In this lesson

You will learn:

→ how to plan and create a program that uses a random number.

Make a better plan

The requirement is to create a program that allows the user to practise the seven times table. In the last lesson, you created a program that asks a question. The program always asks the same question.

It will be better if the program asks a different 'seven times' question each time. That will give the user more practice of the seven times table.

In this lesson, you will change the program so that it asks a different 'seven times' question every time you run it. Here is the improved plan.

Input: Make a **random** number.

Ask "What is 7 times the random number?"

Input the user's 'answer'.

Process: Calculate the 'solution' to the question.

Compare the user's answer to the solution. Do they match?

Output: If the answer matches the solution, output 'You got it right!'

Else, output 'You got it wrong!'

In the last lesson you made a new variable called 'solution'. Now make another variable called 'random'. This variable will store the random number.


Set a random value

This program uses a random value. The computer will pick the random value. The value could be different every time.

Green blocks are 'Operators'. They make new values. This block will pick a random value from 1 to 10. Change the higher value to 12.

Fit these two blocks together to give the random value to the new 'random' variable.

This block goes at the start of the program.

A green Scratch 'pick random' block with the number '1' in a white circle on the left and '12' in a white circle on the right, with the word 'to' between them.An orange Scratch 'set random to pick random' block. It has a dropdown menu showing 'random', followed by the word 'to', and then a green 'pick random' block with '1' and '12' in white circles.

3.4

Ask 10 questions

In this lesson

You will learn:

- how to plan a program that includes a counter loop.

Spiral back



Learn about the forever loop in Student Book 4.

Exit condition

If you have used Scratch before, you may have used a 'forever' loop. Any commands inside this loop will be repeated 'forever' (until the program stops).

There are other types of loop that you can use.

The way that the program stops the loop is called an **exit condition**. It is how you 'exit' from the loop.

Types of loop

There are two main types of loop. They have different exit conditions.

- A **counter loop** (or **fixed loop**) repeats a set number of times.
- A **conditional loop** (or **condition-controlled loop**) is controlled by a logical test.

In this lesson you will plan and create a program with a counter loop.

Change the plan

The program you have made asks one question each time you run the program. A more useful test will ask many questions. In this lesson you will extend the program so that the sprite asks 10 random questions.

Here is the new program plan.

Repeat until there have been 10 loops.

Input: Ask a question and get the user's answer.

Process: Calculate the solution to the question.

Output: If 'answer' = 'solution', output 'You got it right!'

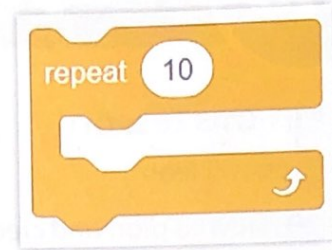
Else, output 'You got it wrong!'

Go back to the top of the loop.

Counter loop

Here is the block that makes a counter loop.

You can see that it counts to 10. Then the loop stops. You could change the value to change the number of times the loop repeats.



Extend the program to match the plan

All the commands in the program must go inside the counter loop. That is because everything must be repeated each time. The completed program is shown opposite.

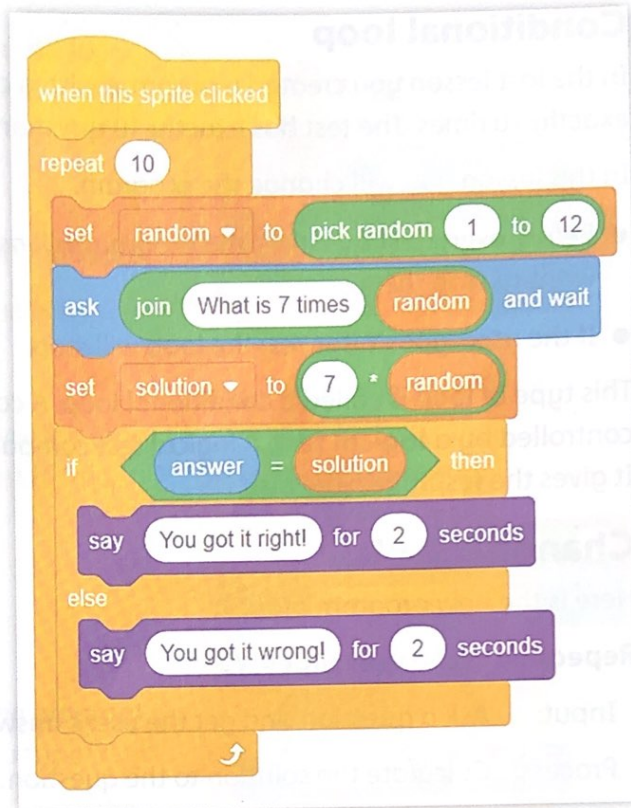


Activity

Load the 'seven times table test' program you made in Lessons 3.1 and 3.2. Extend the program so that it uses a counter loop. Run the program to make sure it works. Save the file.

Think again

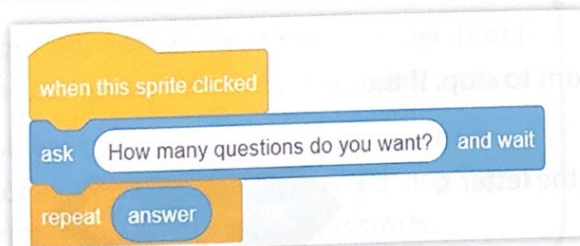
Make a neat version of the program plan. You could write the plan by hand or use a word processor.



Extra challenge

The program you have made always asks 10 questions. Some people might like a different number of questions.

Before the loop starts, ask the user how many questions they want. Then set the number of repeats to the user's answer.



when this sprite clicked

set random ▼ to pick random 1 to 1000

ask join random minus 200 = ? and wait

set solution ▼ to random - 200

repeat until answer = solution

ask Try again. and wait

