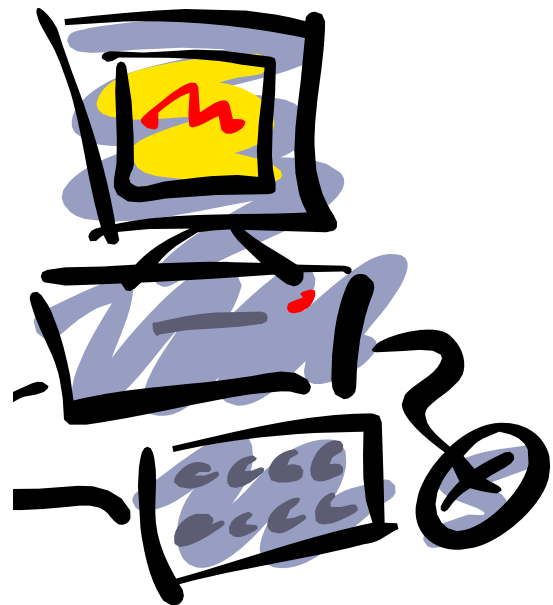


Writing Algorithms



Using Algorithms as a Problem Solving Tool
- An Introduction

ICT



Designing Solutions to Computer Based Problems

In this unit of work you will learn how to design programs that make a computer perform some simple tasks (or solve some simple problems).

If you sit down in front of a computer and try to write a program to solve a problem, you will be trying to do four out of five things at once. These are :

1. ANALYSE THE PROBLEM
2. DESIGN A SOLUTION/PROGRAM
3. CODE/ENTER THE PROGRAM
4. TEST THE PROGRAM
5. EVALUATE THE SOLUTION
[This comes later]

It is impossible to do four things correctly at the same time, We will therefore concentrate on each step, one at a time.



1. ANALYSE THE PROBLEM

When you analyse a problem, you look at the problem in front of you and decide what you need to do to solve the problem.

2. DESIGN A SOLUTION/PROGRAM

Program design is the most important part in producing a computer program. This is the stage where you decide how your program will work to meet the decisions you made during analysis.

Program design does not require the use of a computer - you will carry out your design using pencil and paper.

In your design, you will use a method called **TOP DOWN DESIGN**. In using this method, you simply write down the steps in the correct order, that you are required to perform, to solve the problem.

When all these steps are put together you have what is called an **ALGORITHM**.

Algorithm:

A step-by-step set of instructions for solving a problem in a limited number of steps. The instructions for each step are exact and precise and can be carried out by a computer.

Creating an Algorithm

To begin with we will look at three methods used in creating an algorithm, these are

STEPPING, LOOPING and CHOOSING.

□ STEPPING OUT

First of all we will look at a method of creating an algorithm called STEPPING. This is where all the instructions needed to solve our problem are set out one after the other. Here are some examples.

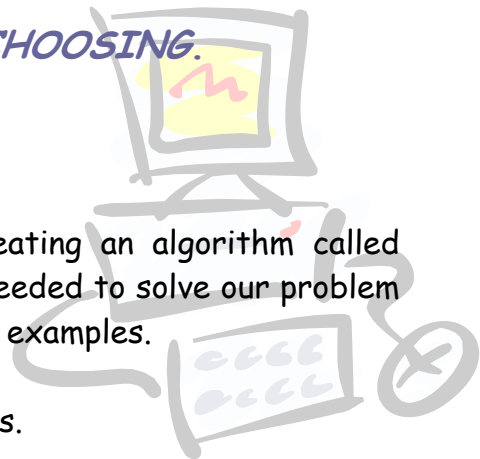
PROBLEM: To find the sum of two numbers.

ALGORITHM: 1 add the two numbers together
2 write down the answer.

PROBLEM: To change the temperature in Fahrenheit to Centigrade.

ALGORITHM: 1 subtract 32
2 multiply the result in (1) by 5
3 divide the result in (2) by 9
4 write down the answer.

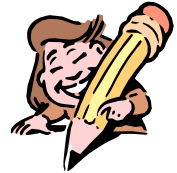
Both of these examples list the algorithm in an ordered set of steps. Now it's your turn to try a few.



EXERCISE:

In your workbook, write down an algorithm for each of the following problems:-

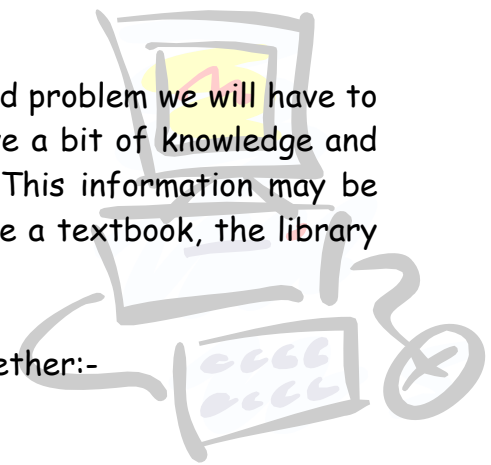
- 1) Find the average of four numbers.
- 2) Change the time in seconds to minutes.
- 3) Find the product of two numbers (this means to multiply the two numbers).
- 4) Change a volume in pints to litres (there are 2.2 pints in every litre).
- 5) Find the difference between two numbers.



All of the problems you have just worked on are fairly simple. Now we will look at some slightly more complicated problems.

To write an algorithm to solve a more complicated problem we will have to use a bit more brain power. We will need to have a bit of knowledge and may have to find out some more information. This information may be found from a few different sources, for example a textbook, the library or from your teacher.

As an example, let's work through a problem together:-



PROBLEM: Find the volume of a cone given its diameter and height.

In this problem we know what the diameter and height are of the cone and we are asked to find its volume. To solve this problem, we must find out the mathematical formula that allows us to calculate the volume of the cone. Where could we find this formula if we don't already know it? - We could look in a mathematics textbook or we could even try asking our teacher.



Our kind teacher has told us that the formula we need is:

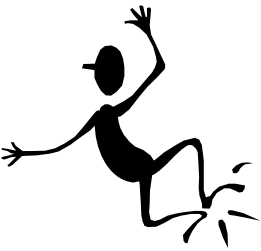
$$\text{Volume} = \frac{\pi \cdot r^2 \cdot h}{3} \text{ which in plain English means}$$

$$\text{Volume} = 3.14 \times \text{radius} \times \text{radius} \times \text{height} \div 3$$

We still have a bit of a problem here, we don't know how to find the radius of the cone, so again we ask our friendly teacher who tells us that the radius is half of the diameter.

So given this information we can now write our algorithm:-

- ALGORITHM:**
- 1 divide the diameter by 2 to give the radius
 - 2 multiply 3.14 by the radius
 - 3 multiply the result in (2) by the radius
 - 4 multiply the result in (3) by the height
 - 5 divide the result in (4) by 3 to give the volume
 - 6 write down the answer.



This gives the volume of the cone ~ fairly straight forward!

Another example is:-

PROBLEM: Heat up a can of soup

- ALGORITHM:**
- 1 open can using can opener
 - 2 pour contents of can into saucepan
 - 3 place saucepan on ring of cooker
 - 4 turn on correct cooker ring
 - 5 stir soup until warm



This may seem a bit of a silly example but it does show us that the order of the events is important since we cannot pour the contents of the can into the saucepan before we open the can.



EXERCISE:

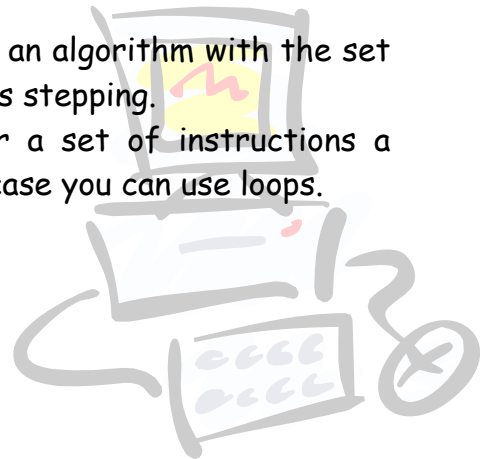
In your jotters write down an algorithm for each of the following problems (remember you may not have all of the knowledge you need to solve some of these problems ~ ask for help):-

- 1) To find the average speed of a car given the journey time and the distance travelled.
- 2) To find the volume of a cube given the length of a side.
- 3) To find the volume of a pyramid, given the length and breadth of its base and its height.

□ GOING LOOPY ! [LOOPING]

So far, all of the problems we have solved used an algorithm with the set of instructions in a set order that we refer to as stepping. What if you have to repeat an instruction or a set of instructions a number of times to find your solution? In this case you can use loops. We will look at three different types of loops:-

- i. the **REPEAT UNTIL** loop
- ii. the **WHILE** loop
- iii. and the **FOR** loop



now we will look at each of these a little closer....

i. REPEAT UNTIL LOOP

This type of loop keeps on carrying out a command or commands UNTIL a given condition is satisfied, the condition is given with the UNTIL command, for example:-

PROBLEM: To wash a car

ALGORITHM: 1 **REPEAT**
2 wash with warm soapy water
3 **UNTIL** the whole car is clean



EXERCISE:

In your jotters write down an algorithm for each of the following problems

(All of these problems can be solved by using REPEAT UNTIL loops):-

- 1) To keep delivering newspapers until your paper bag is empty
- 2) To paint a wall
- 3) To complete a set of multiple choice questions



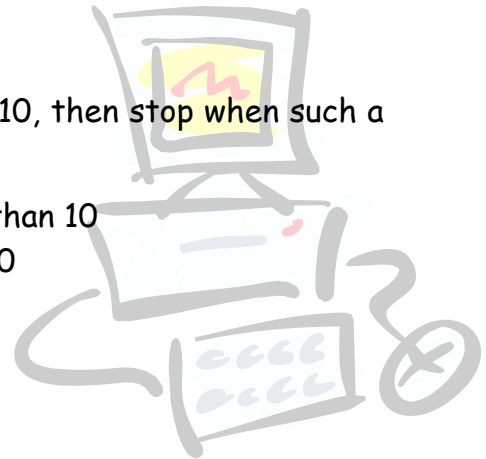
ii. **WHILE LOOP**

In this type of loop the condition is given along with the WHILE command and the loop keeps on carrying out a command or commands until an END WHILE command is given, for example:-

PROBLEM: To ask for a number more than 10, then stop when such a number is given

ALGORITHM:

- 1 WHILE number given is less than 10
- 2 ask for a number more than 10
- 3 END WHILE



EXERCISE:

In your jotters write down an algorithm for each of the following problems

(All of these problems can be solved by using WHILE loops):-

- 4) To keep asking for a number less than 5 and stop when a number less than 5 is given.
- 5) To keep asking for a password, and to give the message "accepted" when the correct password is given.
- 6) To ask for the length of one side of a square, then keep asking for guesses for the area of the square until the correct area is given (think about stepping and looping).



iii. **FOR LOOP**

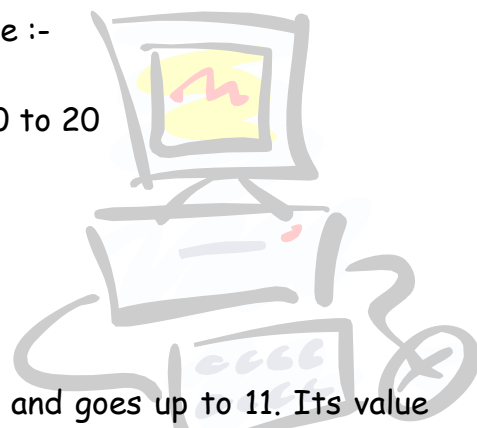
A FOR loop keeps on carrying out a command or commands, **FOR** a given number of times. In this type of loop the number of times it is repeated must be known. The commands to be repeated are sandwiched between the FOR and END FOR commands, for example :-

PROBLEM: To print the numbers from 10 to 20

ALGORITHM: 1 **FOR** number = 10 to 20
2 Print number
3 **END FOR**

NOTE:

In this loop, the value of **number** starts at 1 and goes up to 11. Its value increases by one each time it goes through the loop until it reaches 11. Since the value of **number** changes or varies, **number** is called a variable. A variable can be given any name provided the name remains the same throughout your program. We will use the word variable many times in this work.



VARIABLE :
A quantity or thing, which throughout a calculation, will change in value. Its name should describe what it is representing e.g. number or litre or weight etc.

Another example is:-

PROBLEM: To print the 13 times table from 1 x 13 to 10 x 13

ALGORITHM: 1 **FOR** z = 1 to 10
2 Print 13 x z
3 **END FOR**

In this loop our variable is **z**, who's value starts at one and then increases in value by 1 each time through the loop until **z** has the value of 10. Line (2) of this algorithm prints the value of the variable **z** multiplied by 13.

EXERCISE:

In your jotters write down an algorithm for each of the following problems:

(All of these problems can be solved by using FOR loops):-



- 1) To print the 9 times table from 1×9 to 12×9 .
- 2) To print a message of your choice 4 times.
- 3) To print a table giving the price of petrol from 1 litre to 25 litres, given that the price of petrol is 69.5p per litre.
- 4) To print the values of a number squared from 2 to 20.
- 5) To print a table showing the area of circles with radii from 10cm to 20cm (ask if you are not sure how to find the area of a circle).

GOING LOOPY ! - WHICH ONE TO USE ?

REVIEW EXERCISE:

In your jotters write down an algorithm for each of the following problems:

(All of these problems can be solved by using a loop):-



- 6) To print the 9 times table from 1×9 to 12×9 .
- 7) To print a message of your choice 4 times.
- 8) To print a table giving the price of petrol from 1 litre to 25 litres, given that the price of petrol is 69.5p per litre.
- 9) To print the values of a number squared from 2 to 20.
- 10) To print a table showing the area of circles with radii from 10cm to 20cm (ask if you are not sure how to find the area of a circle).
- 11) To print the 9 times table from 1×9 to 12×9 .

- 12) To print a message of your choice 4 times.
- 13) To print a table giving the price of petrol from 1 litre to 25 litres, given that the price of petrol is 69.5p per litre.
- 14) To print the values of a number squared from 2 to 20.
- 15) To print a table showing the area of circles with radii from 10cm to 20cm (ask if you are not sure how to find the area of a circle).

□ CHOOSING [IFs, THENs AND ELSEs]

So far we have come across the method of stepping commands and looping commands. Now we will look at a method for making a choice or a decision. This technique is called CHOOSING, and uses the commands **IF**, **THEN** and sometimes (but not always) **ELSE**.

With this type of command a condition is given with the **IF** command followed by an action to be taken. If the condition is satisfied, we follow it by the **THEN** command. All a bit complicated? Well here are a couple of examples which should make things a little clearer:-

PROBLEM: To decide if a fire alarm should be sounded

ALGORITHM: 1 **IF** fire is detected ← condition
2 **THEN** sound fire alarm ← action



Another example is:-

PROBLEM: To decide whether or not to go to school

ALGORITHM: 1 **IF** it is a weekday **AND** it is not a holiday
2 **THEN** go to school
3 **ELSE** stay at home

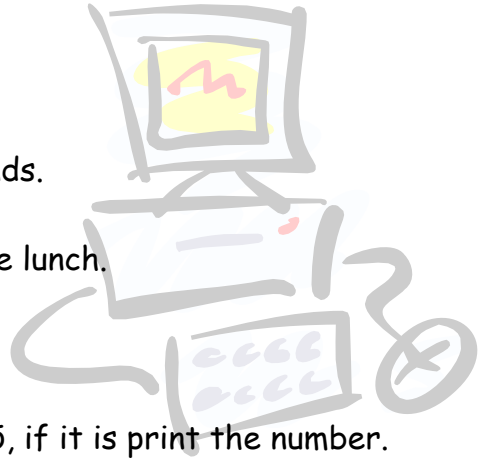
In this example another command, **AND**, was used. This command is used to allow for two or more conditions that have to be satisfied. You will notice that we have also used the **ELSE** command in this example where we are stating an action that must be followed if the conditions are not met.

EXERCISE:



In your jotters write down an algorithm for each of the following problems (All of these problems can be solved by using the choosing method):-

- 1) To decide whether or not to wash your hands.
- 2) To decide whether or not it is time to make lunch.
- 3) To print the largest of two given numbers.
- 4) To decide if a number is between 10 and 15, if it is print the number.

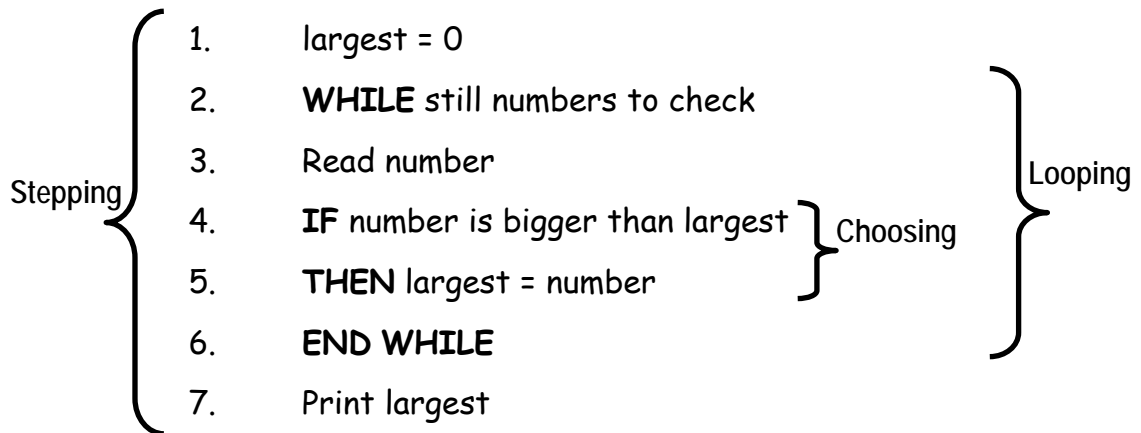


ALL TOGETHER NOW..

When you are programming you will find very few problems that can be solved using just stepping, looping or choosing. In fact most problems will probably need a combination of all three techniques. These examples combine STEPPING, LOOPING and CHOOSING to find a solution to the problems.

PROBLEM: To find the biggest number in a large set of numbers.

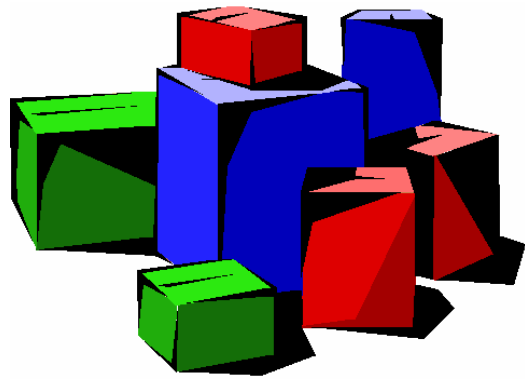
ALGORITHM:



PROBLEM: To count the number of red boxes in a set of red, white and blue boxes.

ALGORITHM:

1. number = 0
2. **WHILE** still boxes to check
3. Read colour
4. **IF** colour is red
5. **THEN** number = number +1
6. **ELSE** number = number
7. **END WHILE**



Now its your turn to try out all this new found knowledge in the writing of **ALGORITHMS** using **STEPPING**, **LOOPING** with **VARIABLES** and **CHOOSING**.



EXERCISE:

In your jotters write down an algorithm for each of the following problems (Stepping, looping and choosing will have to be used in different combinations for each problem):-

- 1) To print the smallest number in a large set of numbers.
- 2) To print the average of a large set of numbers.
- 3) To check a list of job applicants and reject all those who either smoke or drink.
- 4) To ask for a number, less than 50, and then keep adding 1 to it until 50 is reached.

PROBLEMS PROBLEMS PROBLEMS !!

If you are asked to find a solution to a major problem, it can sometimes be very difficult to deal with the complete problem all at the same time. For example building a car is a major problem and no-one knows how to make every single part of a car. A number of different people are involved in building a car, each responsible for their own bit of the car's manufacture. The problem of making the car is thus broken down into smaller manageable tasks. Each task can then be further broken down until we are left with a number of step-by-step sets of instructions in a limited number of steps. The instructions for each step are exact and precise.

Top Down Design uses the same method to break a programming problem down into manageable steps. First of all we break the problem down into smaller steps and then produce a Top Down Design for each step. In this way sub-problems are produced which can be refined into manageable steps.

An example of this is:-

PROBLEM: Ask for the length, breadth and height of a room.
Then calculate and display the volume of the room.

ALGORITHM:

- 1 ask for the dimensions
- 2 calculate the volume
- 3 display the volume

Step 1: 1. Ask for the dimensions

Refinement:

- 1.1 get the length
- 1.2 get the breadth
- 1.3 get the height

Step 2: 2. Calculate the volume

Refinement: 2.1 $\text{volume} = \text{length} \times \text{breadth} \times \text{height}$

Step 3 does not need any further refinement, so the design is now complete.

This was a fairly straightforward problem and design. Another, slightly more complicated example of Top Down Design is explained below:-

PROBLEM: To repair a puncture on a bike wheel.

ALGORITHM:

1. remove the tyre
2. repair the puncture
3. replace the tyre



Step 1:

1. Remove the tyre

Refinement:

- 1.1 turn bike upside down
- 1.2 lever off one side of the tyre
- 1.3 remove the tube from inside the tyre

Step 2:

2. Repair the puncture

Refinement:

- 2.1 find the position of the hole in the tube
- 2.2 clean the area around the hole
- 2.3 apply glue and patch

Step 3:

3. Replace the tyre

Refinement:

- 3.1 push tube back inside tyre
- 3.2 replace tyre back onto wheel
- 3.3 blow up tyre
- 3.4 turn bike correct way up

Sometimes refinements may be required to some of the sub-problems, for example if we cannot find the hole in the tube, the following refinement can be made to 2.1:-

- Step 2.1:** 2.1 Find the position of the hole in the tube
- Refinement:** 2.1.1 **WHILE** hole cannot be found
- 2.1.2 Dip tube in water
- 2.1.3 **END WHILE**

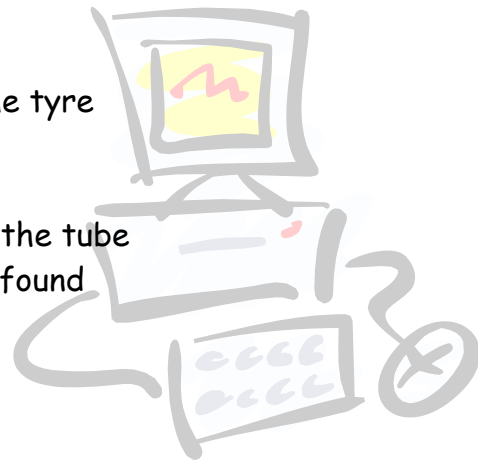
If we now put all of these refined steps together we are left with the complete algorithm needed to solve our problem.

1. Remove the tyre
 2. Repair the puncture
 3. Replace the tyre
-

1. Remove the tyre
 - 1.1 turn bike upside down
 - 1.2 lever off one side of the tyre
 - 1.3 remove the tube from inside the tyre

2. Repair the puncture
 - 2.1 find the position of the hole in the tube
 - 2.1.1 **WHILE** hole cannot be found
 - 2.1.2 Dip tube in water
 - 2.1.3 **END WHILE**
 - 2.2 clean the area around the hole
 - 2.3 apply glue and patch

3. Replace the tyre
 - 3.1 push tube back inside tyre
 - 3.2 replace tyre back onto wheel
 - 3.3 blow up tyre
 - 3.4 turn bike correct way up



EXERCISE:

In your jotters write down a Top Down Design for each of the following problems (Show your initial algorithm and refined sub-problems as shown in the above example):-



- 1) To make a butter and jam sandwich.
- 2) To ask for the length of all the walls in a room, ask for the height of the room, calculate and then display the total area of the room.
- 3) To calculate an employee's wages using these conditions:
Ask for the employee's name and how many hours they worked.
Up to 40 hours are paid at £4.00 per hour, all hours over 40 hours are paid at "time and a half"
Display the employee's name, their basic rate pay, their overtime pay and their total pay.