# Unit 1 Problem Solving

## Short Introduction

Problem solving is the process of solving complex problems. This unit introduces the methodologies to understand a problem and solve it in an effective manner.

## Students' Learning Outcomes

### 1. Problem Solving Steps

- Defining a problem
- Understanding a problem
- Planning a solution
- Defining candid solutions
- Selecting the best solution

### 2. Flowcharts

- Defining a flowchart
- Explaining the importance of a flowchart for problem solving
- Determining the requirements for a flowchart
- Using flowchart symbols
- Drawing flowcharts for sample problems

### 3. Algorithm

- Defining an algorithm
- Describing role of algorithm in problem solving
- Formulating an algorithm
- Writing algorithms for sample problems
- Understanding efficiency of algorithms
- Differentiating between algorithms and flowcharts

### 4. Test data

- Understanding the concept of test data
- Describing importance of testing
- Understanding types of test cases

### 5. Verification and validation

- Understanding the concept of verification
- Understanding the concept of validation

### 6. Identification and correction of errors

- Using trace table for testing
- Using invalid test data for testing

## 1.1 Problem Solving Steps

In order to solve a problem, it is important to follow a systematic approach. In the following we discuss different steps that we can follow to solve a problem systematically.

### 1.1.1 Defining a Problem

A well-defined problem is the one that does not contain ambiguities. All the conditions are clearly specified and it has a clear goal. It is easy to understand and solve.

Given a problem statement, first we need to see whether the problem is defined well or not. If the problem is not defined well then we can use one of the following strategies to define the problem.

**Gain Background Knowledge:** We try to know the situation and circumstances in which the problem is happening. In this way, we can identify the given state. It also helps to know what a good solution will look like. How we shall be able to measure the solution.

**Use Guesses:** We try to guess the unknown information through appropriate guesses. These guesses may be bases upon our past experiences.

**Draw a Picture:** If the problem is not well-defined, we can draw a picture and fill the undefined information. Figure 1-1 shows pictorial representation of a problem.

**Do you know?**

Albert Einstein said, "If I were given one hour to save the planet, I would spend 59 minutes defining the problem and one minute resolving it".

**Remember!**

Pictures speak louder than words.

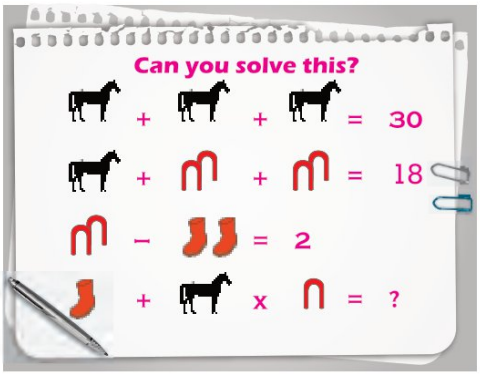



**Figure 1-1 Problem shown in a picture form**

## 1.1.2 Understanding a Problem

It is important to understand the problem before jumping into the solution of the problem. For example, a riddle or a puzzle can be answered only after clear understanding. A clear understanding of a problem makes it easier to solve and helps to save money, time and resources. Understanding of a problem usually includes identification of the 5 Ws (what, who, when, where, and why). Problem analysis is the process to figure out these 5 Ws from a problem statement. Problem analysis helps to understand a given problem. These are the basic elements which lead towards the solution of a given problem. For example, consider the following problem statement:


Figure 1-2 From problem to solution

> "Suppose your class teacher assigns you a task to prepare a list of students in your school whose names start with letter 'A'. The list is required in order to prepare an alphabetical directory of all school students and there is only one week to complete the task."

We can analyse this problem by identifying 5Ws in the problem statement as given below:

- **What:** List of students' names starting with letter 'A'.
- **Who:** Students.
- **Why:** To prepare the directory of students.
- **When:** Within a week.
- **Where:** School.

Figure 1-2 shows the metaphorical representation of problem where the red light presents a problem, the yellow light represents its analysis and the green light presents the solution. It shows that problem analysis makes us closer to a solution.

**Activity 1.1**

Students are put in groups of two or three, and each group is provided two different lists of students' names.

One list contains the marks of students in mathematics subject while the other list is for physics subject. Each group is supposed to prepare the following lists.

a) Top 5% students in mathematics.
b) Top 5% students in physics.
c) Students having more than 90% marks in both subjects.

Identify 5 Ws for this problem.

## 1.1.3 Planning a Solution

After analyzing a problem, we formulate a plan that may lead us towards the solution of a problem. This phase includes finding the right strategy for problem solving. Some of the strategies are:


Figure 1-3 Planning for success

- **Divide and Conquer:** This strategy divides a complex problem into smaller problems.
- **Guess, Check and Improve:** The designer guesses a solution to a problem and then checks the correctness of the solution. If the solution is not according to expectations, then he/she refines the solution. The refinement is an iterative process.
- **Act it Out:** In this strategy the designer defines the list of "to-do" tasks. Afterwards he/she performs the task.
- **Prototype (Draw):** This technique draws a pictorial representation of the solution. It is not the final solution. However, it may help a designer to understand the important components of the solution.

The selection of a strategy depends upon the problem. It is quite important that one strategy maybe more suitable to implement a solution than the other one. Very specifically, the selection of the strategy depends upon the nature of a problem.

## 1.1.4 Defining Candid Solutions

The word candid refers to something spontaneous and unplanned. For example, if you are asked to find number of students in your school who can play cricket. You can estimate by finding cricket players in your class and then multiplying it by the total number of classes in your school. Your answer in this way is the candid solution. To find exact number of cricket players, you have to opt some other way, like visiting each class or getting data from teachers. One can think of a candid solution anytime. A candid solution can help to save time. In Figure 1-4, there are different ways shown to reach a certain place (which can be reached either by going across the wall or by going sideways) and the one you think can work, is the candid solution. It is not necessary that the candid solution is the actual solution of a problem.

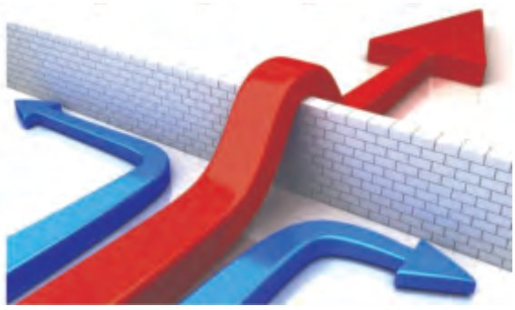


**Figure 1-4 Multiple solutions of a problem**

### Activity 1.2

Your task is to find average height of your class fellows. Give a candid answer and also the method to find the exact solution.

One method is to use a measuring tape. Mark the height on the tape and then read the exact measurement from the tape. After recording the height of every student, you can calculate the average height of all the students in your classroom.

Or you can even find out the candid height of a student through some object of known height, like a book. Let's say that the height of your textbook is 8cm. You can mark the height of the book on a wall. Using the book several times, you can make a scale with intervals of 8cm. Then, by standing next to the wall you can get a candid solution to the student's height.

### 1.1.5 Selecting the Best Solution

Sometimes we find more than one solutions of a problem and select the best one amongst them. For example, assume that names of all the students in your school are available on a website and you are asked to search a particular name. You can solve this search problem by either of the following methods:


**Figure 1-5 Levels of a solution**

1. Look at each name on the website one by one until the name is found or the list is over.
2. Take printouts and search the required name.
3. Copy names, put them in Excel sheet and sort there in alphabetical order. Searching in a sorted list is comparatively easy.
4. Just press Ctrl+F, when the list is available in a web browser. You can type the name to search automatically.

There can be other solutions as well. Now we can identify a solution that has less number of steps or that seems more effective based on some criteria.

## 1.2 Flowcharts

Flowcharts are helpful to know about the steps used to solve a problem. In the following, we discuss the concept of flowcharts in detail.

### 1.2.1 Definition

A flowchart is a graphical presentation of the steps to solve a problem. We use symbols for each step, and these symbols are connected with the help of arrows to show the flow of processing.

Figure 1-6 shows a flowchart for the simple problem of wearing shoes with socks. It shows that not only the steps are important but also the order to complete a process. A

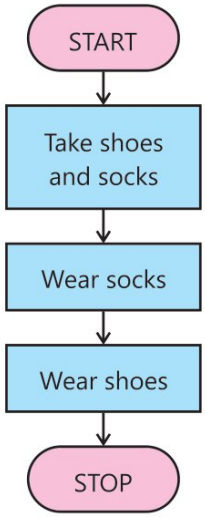



**Figure 1-6 Sample flowchart**

flowchart is used to visually communicate the steps in a process.

## 1.2.2 Importance of Flowcharts in Problem Solving

In problem solving, flowcharts can be used to plan a solution. If a flowchart is already there, we can quickly understand the way a problem is solved. It is more effective to visualize a solution graphically than a text. A graphical representation also makes it effective to verify whether a solution is correct or not. It is also a good way to communicate the solution of a problem to other people.

## 1.2.3 Determining Requirements for a Flowchart

In a flowchart we use input, output, decision making and processing. These concepts are described in the following: Input means taking data from a user, processing it and give some output as shown in Figure 1-7. For a flowchart the requirements are to know about:

- **Inputs**: Input means taking data from the user. It is important to know, how many and what type of inputs are required.
- **Processing**: A flowchart also contains processing steps. The processing steps are used for performing calculations and storing the results of calculations. These may include increasing/decreasing a value, adding/multiplying/dividing two values etc.


**Figure 1-7 Flow from input to output**

- **Decision Making**: To determine whether a statement is *true* or *false*, and taking appropriate steps accordingly, is called decision making.
- **Outputs**: Outputs are used to display information and usually this information exhibits the processed results.

## 1.2.4 Flowchart Symbols

Flowcharts explain a process clearly through symbols and text. They use special shapes to represent different types of actions or steps in a process.

Lines and arrows show the flow of the steps. Table 1-1 shows some of the most widely used symbols in flowcharts.

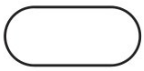
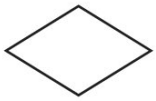
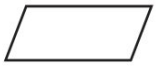
| Symbol | Name | Description |
|---|---|---|
|  | Flow line | It is used to determine the flow of steps in a flowchart. |
|  | Terminal | It indicates start and end of a flowchart. |
|  | Process | It represents operations to change values. |
|  | Decision | It shows a conditional operation that determines which one of the two paths to take. The operation is commonly a yes/no question or a true/false test. |
|  | Input/Output | It indicates the input of data from user or displaying results to user. |
|  | Connector | If a flowchart doesn't fit on a page, then we use connector to connect parts of a flowchart on different pages. |

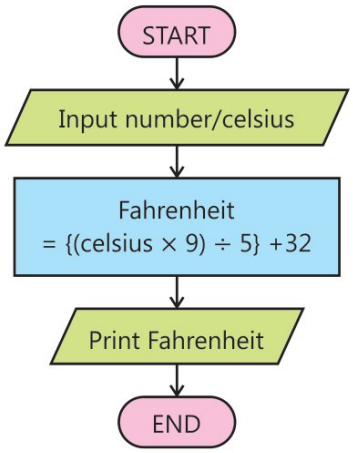
Table 1-1

## 1.2.5 Examples of Flowcharts

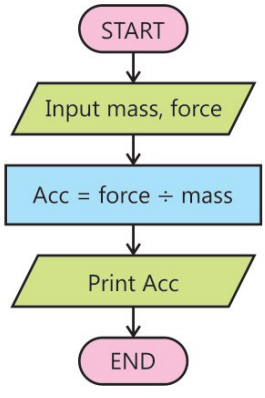
In this section we study problem solving using flowcharts. In a flowchart, *input* means to get data from some input device and store in computer memory. A data in memory is given some particular name. We access data with respect to its given name to perform some computation from which results are obtained. The results are stored in computer memory and we print (display) the results on some *output* device.

1. **To convert Celsius to Fahrenheit temperature.**

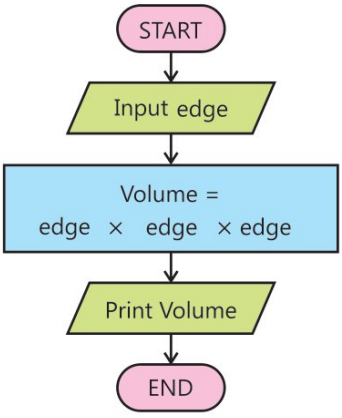
   This flowchart shows that after *Start*, a computer user provides some numeric data which is stored in computer memory with the name *celsius*. In the next step, the result is computed by applying the formula to convert the given Celsius temperature to Fahrenheit temperature. The result is stored with name *Fahrenheit*. The value stored with name *Fahrenheit* is then displayed by using some output device. Next, the flowchart is ended.



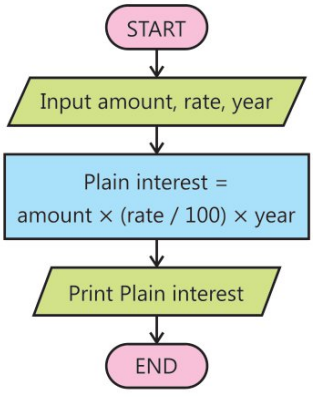

2. **To find acceleration of a moving object with given mass and the force applied.**



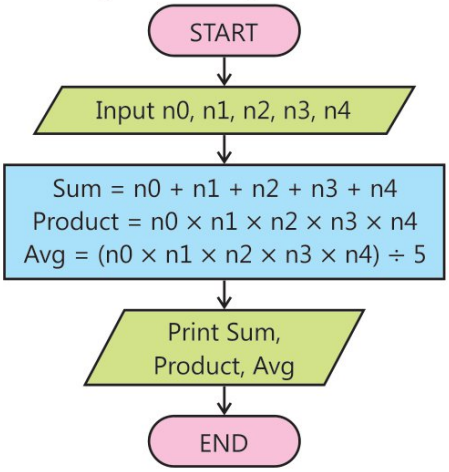

3. **To find the volume of cube.**



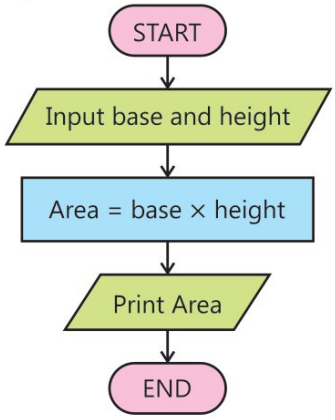

4. **To find plain the interest on an amount.**



**5. To find the sum, product and average of five given numbers.**

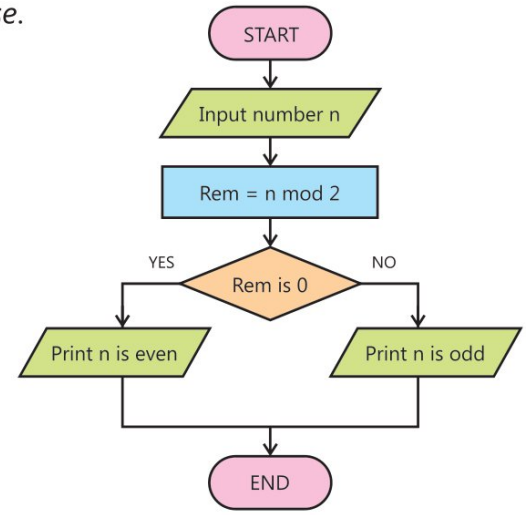



**6. To find the area of a parallelogram.**




**Conditional Flow in Flowcharts:** In the following flowcharts, we study that a flow between steps can depend upon a certain condition. If the condition is *true* then the flow is different from when the condition is *false*. A condition is always evaluated as either *true* or *false*.

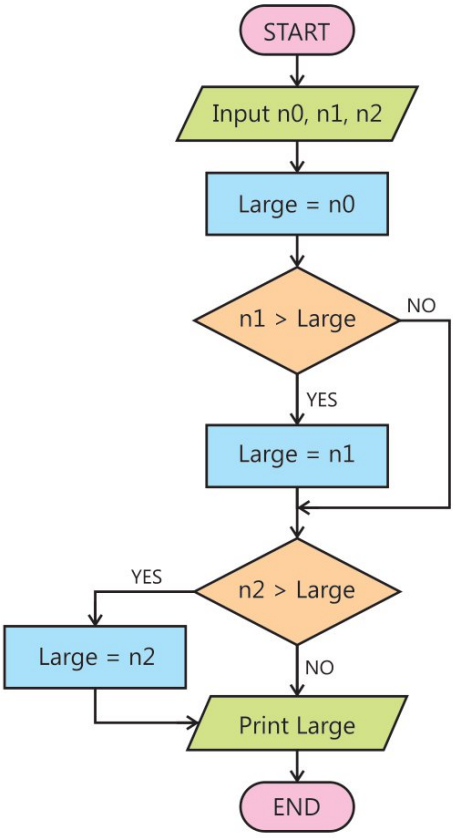
**7. To determine whether a given number is odd or even.**



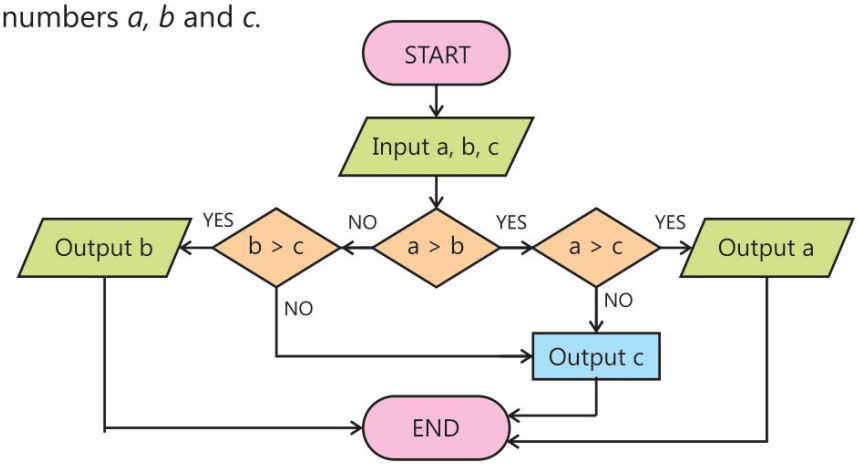

In the above flowchart, after the *start* step a numeric value is taken from a user and stored in computer memory with name *n*. Then this value is divided by 2 and remainder is stored in memory with name *Rem*. To calculate the remainder, *mod* function is applied. We perform a conditional operation to check whether *n* is completely divisible by 2 or not. This is done by comparing the value of *Rem* with 0. If *Rem* is 0, the conditional operation gives a *true*

value. It means that *n* is even because it is completely divisible by 2. On the other hand if *Rem* is not 0, the conditional operation gives a *false* value. It means that *n* is odd.

**8. To display the larger one out of the three given unequal numbers.**



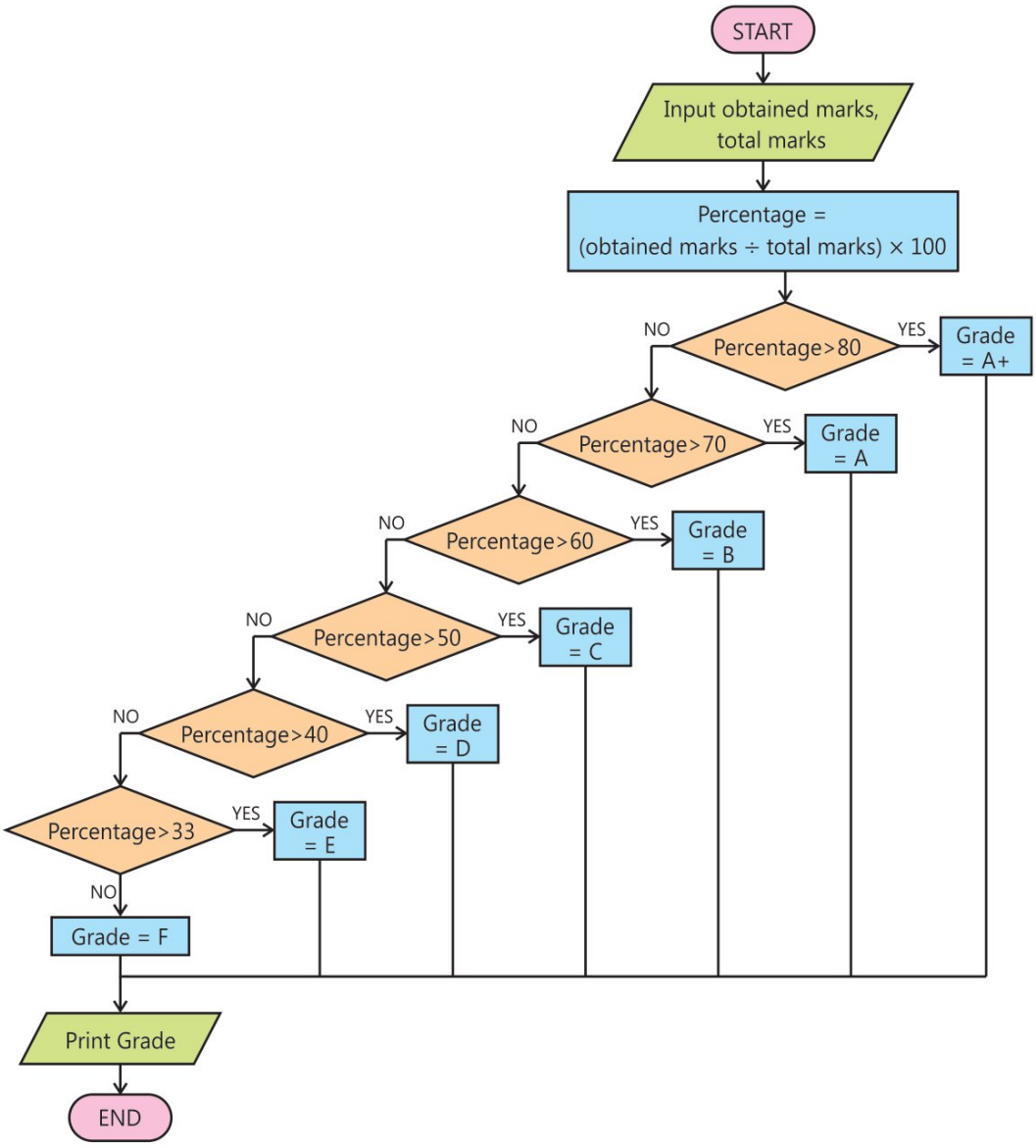

Following flowchart shows another solution to find a maximum value among three numbers *a, b* and *c*.



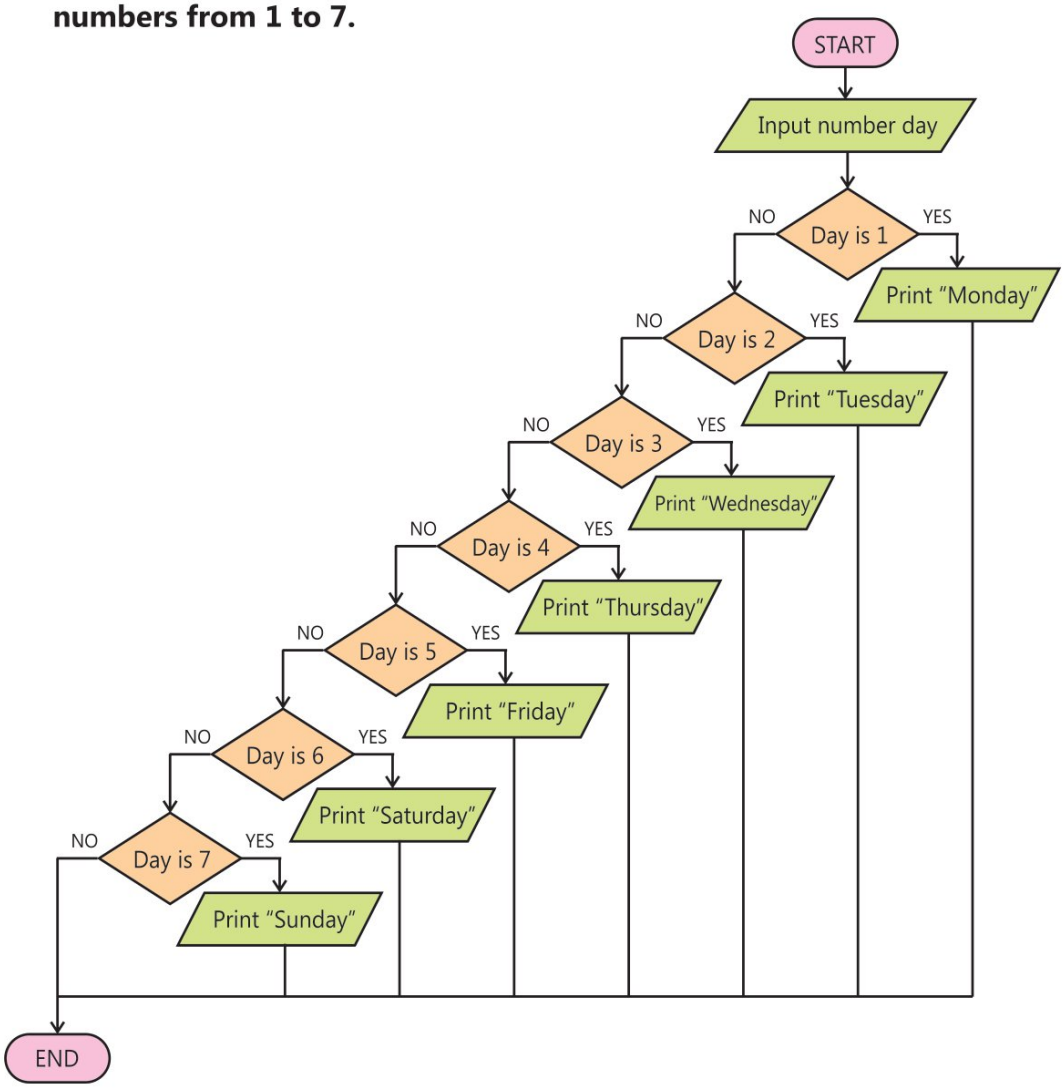
## Activity 1.3

Investigate both the approaches presented to find a maximum value among three numbers and compare them. Which approach has advantage(s) over the other? Write your findings and discuss with your class teacher.

### 9. To assign grade to a subject based on total marks and obtained marks.



**10. To determine name of a week day from a given number where weekdays are assumed from Monday to Sunday and their respective numbers from 1 to 7.**



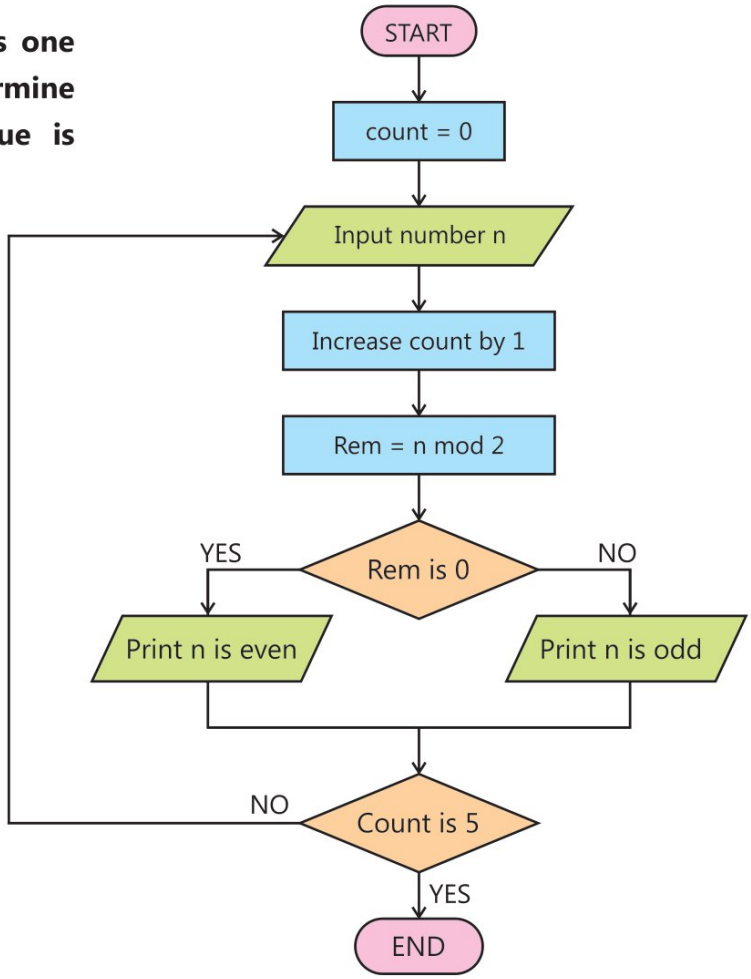

## Activity 1.4

In the above flowchart, a user is supposed to provide the input value from 1 to 7, but it is possible that the input value is less than 1 or more than 7. We need to take care of such values and display appropriate messages, like "Your value is less than 1" or "Your value is more than 7". Modify the above flowchart with respect to proper error messages.

**Repeating Steps in Flowcharts:** In all of the above flowcharts, the flow is from top to bottom but it is also possible that the flow may go to a previous step, especially when we need to repeat some steps.

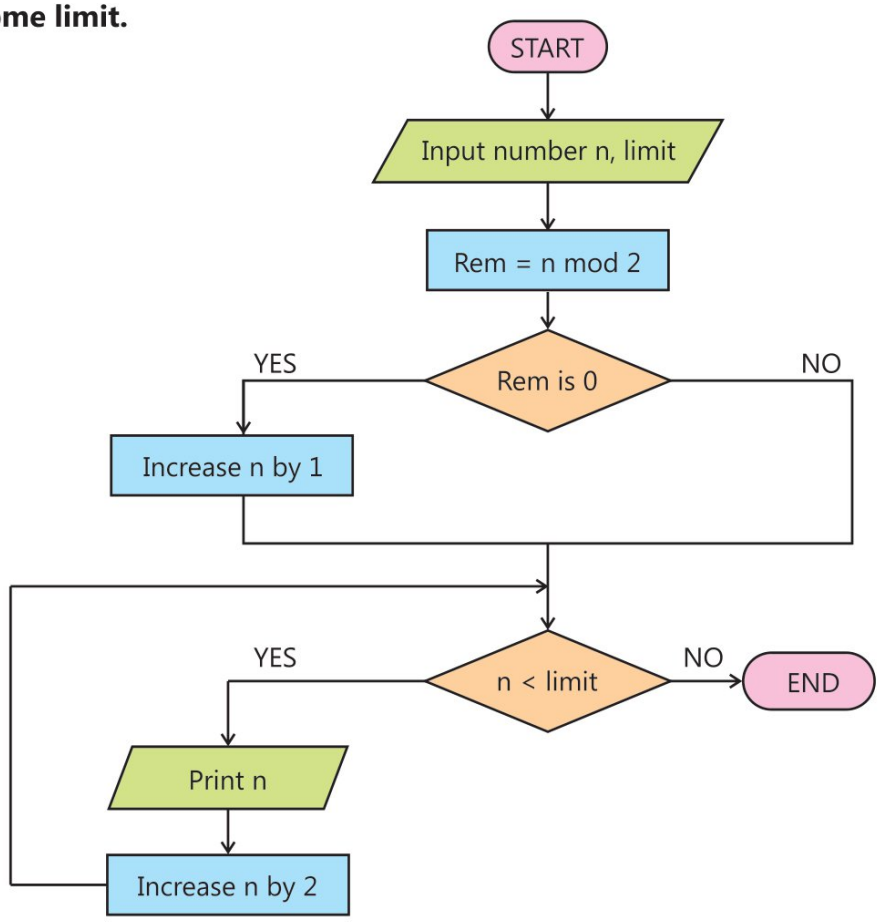
In the following, we show some flowcharts containing steps that are repeated.

**11. To input 5 values one by one and determine if the given value is odd or even.**




The above algorithm uses a *Count* to keep counting the number of inputs given by a user. So, we initially give it the value 0 and increase it by 1 after every input. After printing even or odd we compare it with 5. If the *Count* reaches to 5, we stop the flow or otherwise transfer it to input another value.

**12. To find a sequence of odd numbers starting from a given number till some limit.**




## 1.3 Algorithm

Studying algorithms is a fundamental part of computer science. In this section, we discuss the concept of algorithms in detail.

### 1.3.1 Definition

An algorithm is a set of steps to solve a problem. It is written in a natural language, so it is easily understandable by humans. For example, to solve the problem of preparing tea, we can follow the following steps.

a) Start.
b) Take a kettle.

c) Pour water in it.
d) Put the kettle on fire.
e) Add sugar and milk.
f) Wait till it boils.
g) Remove the kettle from fire.
h) End.

The above set of steps can be called an algorithm for tea preparation.

We can also solve certain problems with the help of a computer. For that purpose, we first formulate an algorithm for the problem and then translate that algorithm into a set of instructions for the computer. Usually an algorithm takes an input and then after processing produces some output as shown in Figure 1-6.

**Do you know?**

The word "algorithm" comes from the name of Arabic writer Muḥammad ibn Mūsā al-Khwārizmī.

## 1.3.2 Role of Algorithms in Problem Solving

An algorithm has a vital role in problem solving as it provides a step-by-step guide to the problem solver. It is a complete description of the solution. Usually a computer programmer first writes an algorithm and then translates it into the code of some programming language. Sometimes, the designer of the program first makes a flowchart to solve a problem and then encodes the flowchart into an algorithm. Figure 1-8 shows the role of algorithm in problem solving.

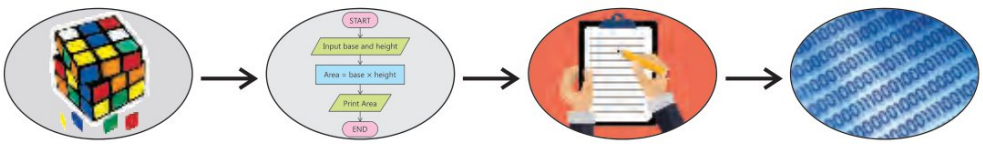



**Figure 1-8 Role of Algorithm**

## 1.3.3 Formulation of an Algorithm

There are different notations (keywords) to write an algorithm. We use the notations presented in Table 1-2 to write an algorithm.

| Notation | Meaning |
|---|---|
| Start | It is the starting point of an algorithm. Every algorithm must have one starting (entry) point. |
| Input | It is used to get input from a user and store it in computer memory with some name. |
| Set | It is used to give name to data in computer memory. It is also used to update the value of existing data. |
| if, else | It is used to check the condition. For example, the condition like *if (a<b).* A condition is evaluated as *true* or *false*. In case the condition is *true* then the statements related with *if* part are executed otherwise the statements of *else* part are executed. **Usage:** Suppose a=5 and b=7, if(a<5) Set *c* to10 else Set *c* to 20.<br>Writing *else* part is optional. |
| Goto | It is used to transfer control to a certain step of an algorithm. It is usually required in loops. |
| Output | It is used to display values. |
| Stop | It is the termination point of an algorithm. |

**Table 1.2 Notations for writing an algorithm**

## 1.3.4 Examples of Algorithms

**1. To find the sum, product and average of five given numbers.**

Step 1. Start

Step 2. Input numbers, $n0, n1, n2, n3, n4$

Step 3. Set *sum* to $n0 + n1 + n2 + n3 + n4$.

Step 4. Set *product* to $n0 \times n1 \times n2 \times n3 \times n4$

Step 5. Set *average* to $\frac{n0+n1+n2+n3+n4}{5}$

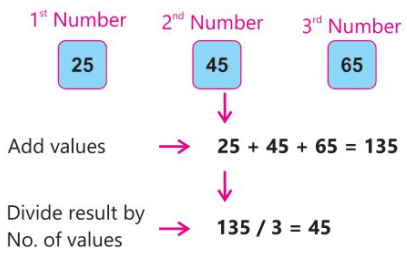
Step 6. Output *sum, product, average*

Step 7. End

In this algorithm, Step 1 shows starting of the algorithm. Step 2 shows that a user provides 5 numeric values and they are stored in computer memory with the names $n0, n1, n2, n3$ and $n4$. Step 3 shows summation of all the input values and storing the result in computer memory with name *sum*. Similarly, Step 4 is used to compute multiplication of all the numbers and store the result with name *product*. In Step 5, formula to calculate average of five numbers is applied and the result is stored with name *average*. The Step 6 shows the results of the steps 3, 4 and 5, respectively. The Step 7 shows end of the algorithm.

**Activity 1.6**

The Figure shows a simple example of calculating average of three numbers, 25, 45 and 65. Write an algorithm with these fixed values to calculate and display average values. Note that in this case no input is required from a user.




Example of calculating Average

**2. To find acceleration of a moving object with given mass and the applied force.**

Step 1. Start
Step 2. Input numbers, *mass, force*
Step 3. Set *acceleration* to $\frac{force}{mass}$
Step 4. Output *acceleration*
Step 5. End


**Figure 1-9 More Force with Same Mass Increases Acceleration**

**3. To find the volume of a cube.**

Step 1. Start
Step 2. Input number, $side$
Step 3. Set *volume* to $side \times side \times side$.
Step 4. Output *volume*
Step 5. End

**Activity 1.7**

Change the above algorithm for finding volume of a Cylinder and Sphere. The formula for the volume of a Sphere is $\frac{4}{3} \times \pi\, r^3$ where *r* is radius. The formula for the volume of cylinder is $\pi r^2 h$ where *r* is radius and *h* is height.

**4. To find the area of a parallelogram.**

Step 1. Start
Step 2. Input numbers, *base, height*
Step 3. Set *area* to $base \times height$
Step 4. Output *area*
Step 5. End

**Activity 1.8**

Change the above algorithm for finding the area of a triangle, rhombus, or trapezium.

**5. To display the larger one out of the three given numbers.**

Step 1. Start
Step 2. Input numbers, $n0, n1, n2$
Step 3. Set *large* to $n0$
Step 4. if $n1 > large$ Set *large* to $n1$
Step 5. if $n2 > large$ Set *large* to $n2$
Step 6. Output *large*
Step 7. End

In this algorithm, Step 1 shows starting of the algorithm. Step 2 shows that a user provides 3 numeric values and they are stored in computer memory with the names $n0, n1$ and $n2$. In this algorithm, initially we assume that

the largest value is $n0$ and we compare it with the other two values. So, Step 3 shows that data of $n0$ is stored in computer memory with a name *large*. Step 4 describes that if the value of $n1$ is greater than *large* then the value of *large* is updated and its new value is the value of $n1$. In case $n1$ is less than or equal to *large* then Step 4 will not affect the value of *large*. Similarly, Step 5 may change the value of *large* provided that $n2$ is greater than *large*. Step 6 displays the resultant value of *large*.

**6. To assign grade to a subject based on the achieved marks.**

Step 1. Start

Step 2. Input numbers, *obtained_marks, total_marks*

Step 3. Set *percentage to* $\frac{obtained\_marks}{total\_marks} \times 100$

Step 4. if $percentage > 80$ Set *grade* to *A+*

else

if $percentage > 70$ Set *grade* to *A*

else

if $percentage > 60$ Set *grade* to *B*

else

if $percentage > 50$ Set *grade* to *C*

else

if $percentage > 40$ Set *grade* to *D*

else

if $percentage > 33$ Set *grade* to *E*

else

Set *grade* to *F*.

Step 5. Output *grade*

Step 6. End

**7. To find the interest on an amount.**

Step 1. Start

Step 2. Input numbers, *amount, rate, years*

Step 3. Set *plain_interest* to $\left(amount \times \frac{rate}{100}\right) \times years$

Step 4. Output *plain_interest*

Step 5. End

8. **To convert Celsius to Fahrenheit temperature and vice versa.**

Step 1. Start

Step 2. Input number, *celsius*

Step 3. Set *fahrenheit* to $\frac{celsius \times 9}{5} + 32$

Step 4. Output *fahrenheit*

Step 5. Input number, *fahrenheit*

Step 6. Set *celsius* to $(fahrenheit - 32) \times \frac{5}{9}$

Step 7. Output *celsius*

Step 8. End

9. **Find even numbers in integers ranging from n1 to n2 (where n2 is greater than n1).**

Step 1. Start

Step 2. Input numbers, $n1, n2$

Step 3. if $(n1 \leq n2)\{$

Step 4. if $(n1$ mod 2 equal 0) Output $n1$

Step 5. Set $n1$ to $n1 + 1$

Step 6. go to Step 3

}

Step 7. End

## 1.3.5 Efficiency of Algorithms

There can be more than one algorithms to solve the same problem. Which one is better, depends upon the efficiency of the available solution algorithms. Efficiency of an algorithm is measured on the basis of two metrics,

1- **Number of steps:** An algorithm is considered more efficient if it takes less number of steps to reach the results.

2- **Space used in computer memory:** We have observed in algorithms that some data is stored in computer memory which is latter used to

give results. An algorithm using less space in computer memory is considered more efficient with respect to memory space.

It is quite possible that one algorithm takes less space in memory and has more number of steps whereas the other algorithm takes more memory and has less number of steps. In this case there is a trade-off between number of steps and the consumed memory. The designer can take decision according to the requirements.

**Example**

Let's suppose we have two algorithms to solve a certain problem. One algorithm has N steps whereas the other algorithm has $N^2$ steps. In this case the former algorithm is considered more efficient than the latter one.

**Example**

We need to compute the following.

$$1 + 2 + 3 + 4 + 5 + \cdots + 99$$

How can we find its answer?

Different minds can find different solutions to solve this problem. One solution is to start adding numbers from beginning and keep adding till the end. Other solution is to start making pairs as $(1 + 99)$, $(2 + 98)$, $(3 + 97), (4 + 96), (5 + 95)$ … $(49,51)$ where each pair gives answer 100. We count the number of pairs and multiply that count with 100 and then in the result we add 50 as it is nowhere in any pair.

Another solution is to use formula $\frac{n(n+1)}{2}$ where $n$ is the last term. So, the solution is just to solve $\frac{99(99+1)}{2}$.

This example shows different approaches to solve one problem and if these approaches are used in computer then accordingly there may be different memory usage and number of steps.

**Do you know?**

There are 64 squares on a chess board. If we place wheat upon each square such that one grain is placed on the first square, two on the second, four on the third, and so on (doubling the number of grains at each square), then there will be 18,446,744,073,709,551,615 grains of wheat on the chessboard at the finish.

**Activity 1.9**

Compare the algorithm 9 presented in Section 1.3.4 with the following one and try to find which one is efficient. Note that both are solving the same problem. In order to answer this question, assume two values for n1 and n2 and start count the number of steps used in both algorithms.

Step 1. Start
Step 2. Input numbers, $n1, n2$
Step 3. if $n1$ is odd, set $n1$ to $n1 + 1$
Step 4. Output $n1$
Step 5. Set $n1$ to $n1 + 2$
Step 6. if $n1 < n2$ go to Step 4
Step 7. End

## 1.3.6 Difference between an Algorithm and a Flowchart

Difference between an algorithm and a flowchart is just like the difference between a story and a movie. As we have studied that a flowchart is a graphical representation of the process to solve a problem but an algorithm writes the same steps in a human understandable language.

**Advantages of a Flowchart**

- Easy to draw.
- Easy to understand problem solving.
- Easy to identify errors (if any).
- Easy to observe flow from one step to the other.

**Disadvantages of a Flowchart**

- More time is required to draw a flowchart.
- Modifying a flowchart is not very easy every time.

- It is not suitable for very large problems.

**Advantages of an Algorithm**

- Easy to write.
- Techniques to write an algorithm are easy to understand.
- To solve a large problem, algorithms are helpful.

**Disadvantages of an Algorithm**

- Modifying an existing algorithm is not very easy every time.
- Showing the flow from one step to the other is not very easy.
- Usage of goto makes it difficult to identify errors.

## 1.4 Test Data

After solving a problem, we need to test whether the solution is correct or not, and for testing, we need "Test Data". For example, if we want to test the algorithm (discussed earlier) to find the largest among three given numbers $n0, n1$, and $n2$, then we need three values. These values can be positive, negative or zero, e.g., $(n0 = 5, n1 = 15, n2 = 3)$, $( n0 = 27, n1 = -6, n2 = 35 )$, $(n0 = 24, n1 = 0, n2 = 11)$, etc. So, for thinking about testing, we also need to think about test data.

**Activity 1.10**

Assume that you are given an automatic attendance system for testing. In this system, a camera observes each student entering in the classroom. The camera is connected to a computer which contains the database of pictures of all the students. The solution compares each student with the pictures in database and mark the attendance is picture is found there.

You are asked to provide test data for the system. Write your points in a way that can help the solution provider to check and improve quality of the solution. You can think about different dresses in different weathers, identical twins, different haircuts or any other points where one can look different in front of camera.

### 1.4.1 Importance of Testing

Testing is essential to point out the defects and errors made during finding a solution to some problem. It helps in improving a solution. If one solves a problem and someone else uses that solution for commercial purposes, then the commercial activities depend upon the correctness of that solution. For example, if we develop a solution for finance management and some bank starts using it then any error in that solution may result in a financial loss. So, testing is important for a solution.

**Do you know?**

The space shuttle Challenger STS-51L spaceflight ended in a tragedy on Jan. 28, 1986 only 73 seconds after liftoff.

A car is delivered to a customer after testing. Upon launching a new car, it is usually tested with a robot driver who hits the car with a wall. It is used to test whether the air bags and other security systems are functioning or not. Moreover, it also allows the car designers to suggest further security measures to reduce the damage. This test can help to make a car safe. So, testing helps to improve quality.

### 1.4.2 Types of Test Data

Creation of proper and sufficient test data is one of the key activities to improve quality of a solution. Each type of solution requires different data.

Types of Test Data include:

- **Valid test data:** It is the test data that complies with the input requirements of the algorithm. If an algorithm is supposed to take a numeric value between 1 and 100 as input, then any value between 1 and 100 is a valid test data.
- **Invalid test data:** It is the data that does not comply with the input requirements of the algorithm. It is necessary to make sure that the solution correctly works for invalid values, shows the relevant messages notifying the user that the provided input values are

improper.

- **Boundary test data values:** A solution is tested on extreme values. For example, to calculate interest we can consider principal amount as 0 or a very huge amount.
- **Wrong data formats:** It is wise to check how the system reacts on entering data in an inappropriate format. For example, giving an alphabet as input when a numeric value is expected.
- **Absent data;** It is also important to investigate that the solution still works if less number of inputs are given than expected. For example, if a system asks to enter driving license number, then every one cannot provide this information. It is important to see how the system reacts in such situations.

## 1.5 Verification and Validation

### 1.5.1 Verification

Verification means to test if the solution is actually solving the same problem for which it was designed. For example, if you are asked to give a solution for calculating compound interest then verification means to know that it is giving results for compound interest not for the plain interest.

### 1.5.2 Validation

Validation means to test whether the solution is correct or not. For example, if you are asked to give a solution for calculating compound interest then validation means to know whether it is finding the correct compound interest or not. If a solution is verified, then it is validated with the help of test data as discussed in previous section.

**Example 1**

Let's assume that you go to a pizza shop and order a chicken pizza. You state your requirement that it should be less spicy. You also expect that it would taste good. When the pizza arrives, you can observe that it is a chicken pizza. This is called verification. Now, when you eat the pizza, you can check whether

it is less spicy or not, it tastes good or not. This is called validation.

**Example 2**

Let's assume that you are asked to write an algorithm that takes as input a list of numbers. The algorithm should display the list arranged in ascending order. After writing the algorithm you submit it to your teacher. Your teacher provides a list of numbers to the algorithm. If your algorithm displays a list of numbers then it is verified. Instead if your algorithm displays an answer in *yes* or *no*, or displays something else, then it is not verified. If your algorithm is verified, your teacher moves to the next step of validation. He checks whether the list of numbers displayed are actually in ascending order or not. If the list is in ascending order and no element is missing then your solution is also validated.

## 1.6 Identification and Correction of Errors

If an algorithm is failed during verification, then it is important to identify the root cause of failure and then to correct it. Sometimes the error is logical. It means the solution is working but not giving required results.

For example, to recruit students for our school volleyball team, we need students having height between 144 cm and 164 cm. To count qualified students, we develop the following algorithm.

Step 1. Start
Step 2. Set count to 0
Step 3. Set all_heights to [154, 140, 155, 164, 144, 166, 160, 143]
Step 4. For each height in the list all_heights
Step 5. If $height > 144$ and $height \leq 164$ then Set count to count + 1
Step 6. Output count
Step 7. Stop

The above algorithm works but does not count all students. There is a deliberate error on Step 5. The symbol $>$ has been used instead of $\geq$ before the number 144. So, the students having height 144cm will not be counted. This is a logical error. We can identify this type of errors using a trace table as

discussed below.

### 1.6.1 Trace Table

A trace table is a technique used to test algorithms, in order to make sure that no logical errors occur while the algorithm is being processed. The table usually takes the form of a multi-column, multi-row table; with each column showing names of data, and each row showing values of the data at each step. Table 1-3 shows a trace table for the algorithm presented in Section 1.6. The blank means there is no change and -- means that a value is not concerned. In the following table Step 1 has no effect on data. Step 2 is assigning 0 to *count* and in Step 3, list *all_heights* is introduced. In Step 4, there is no change in both *count* and *all_heights* but the data 154 is stored in *height*. It is compared in Step 5 and the value in *count* is updated if data is in given range. Steps 4 and 5 are repeated for each value as shown in Table 1-3.

| | count | all_heights | height |
|---|---|---|---|
| Step 1 | -- | -- | -- |
| Step 2 | 0 | -- | -- |
| Step 3 | | [154, 140, 155, 164, 144, 166, 160, 143] | |
| Step 4 | | | 154 |
| Step 5 | 1 | | |
| Step 4 | | | 140 |
| Step 5 | 1 | | |
| Step 4 | | | 155 |
| Step 5 | 2 | | |
| Step 4 | | | 164 |
| Step 5 | 3 | | |
| Step 4 | | | 144 |
| Step 5 | 3 | | |
| Step 4 | | | 166 |
| Step 5 | 4 | | |
| Step 4 | | | 160 |
| Step 5 | 5 | | |
| Step 4 | | | 143 |
| Step 5 | 5 | | |
| Step 6 | | | |
| Step 7 | | | |

**Table 1-3**

## 1.6.2 Using Invalid Data for Testing

Testing an algorithm using invalid data ensures that the algorithm can gracefully handle unexpected data inputs. If an algorithm requires your age in number of days but you give date of birth as input then the algorithm may not work properly. The purpose of testing using invalid test data is to detect such situations. In this case error messages are shown as output. Moreover, this kind of testing helps you to improve the quality of solution.

**Activity 1.11**

Write all the above discussed algorithms keeping in mind the invalid test data inputs. Class teacher may divide class in few groups and assign them one or more algorithm(s). Students are supposed to discuss and rewrite algorithms so that upon invalid inputs, appropriate messages are displayed.

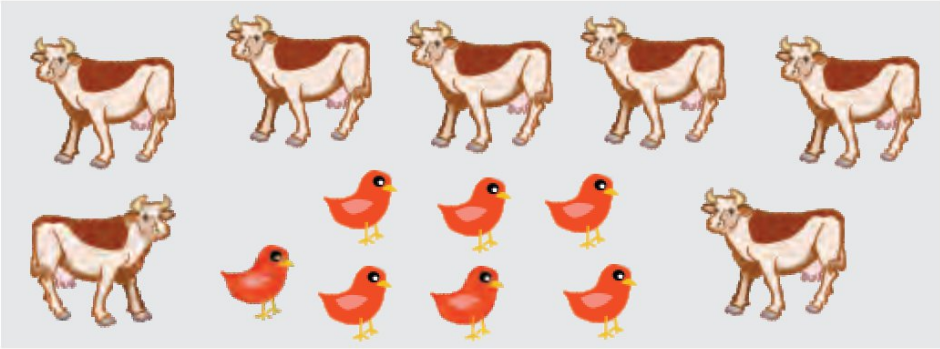
**SUMMARY**

- A problem is a matter or situation needs to be dealt with and overcome.
- Analysing a problem helps to solve that problem quickly.
- A problem may have more than one solution but the best one is the solution comprising less number of steps or the solution is achieved in less amount of time
- An algorithm is a set of rules used for solving problems.
- An algorithm takes some input, processes it, and gives result.
- Algorithms are helpful in decision making
- Flowcharts consist of symbols used for graphical presentation of an algorithm.
- Commonly used symbols for a flowchart are: input/output, decision, arrow, start and stop.
- Validation means to test if the solution is according to given problem.
- Verification means whether the solution is giving the required results or not.
- A trace table is a technique used to test algorithms.

## EXERCISE

### 1.1 Answer the following questions.

1. In a farm there are some cows and birds. If there are total 35 heads and 110 legs then how many cows and birds are there?



2. Define problem analysis. Explain your answer along with an example.
3. Define an algorithm and argue on its role and importance in problem solving.
4. Suppose a problem has multiple algorithms. How would you choose the most efficient one? Explain with examples.
5. How do you determine requirements for a flowchart?
6. Explain types of test data.
7. Describe a trace table.

### 1.2 Choose the correct option.

**1. Which solutions are not reached through proper algorithms or work planning?**

(i) Prepared solution (ii) Candid solution
(iii) strategized solution (iv) best solution

**2. _____ is a graphical representation of an algorithm.**

(i) Matrix (ii) Graph
(iii) Flowchart (iv) solution

**3. Which symbol in the flowchart is used to either start or end the flowchart?**

(i) Terminal (ii) Connector
(iii) Process (iv) decision

**4. _____ means to test if the required solution is there.**

(i) Verification (ii) Algorithm
(iii) Validation (iv) Flowchart

**5. In a _____ error, the solution is working but not giving required results.**

(i) Random error (ii) logical error
(iii) syntax error (iv) Runtime error

## 1.3 Fill in the blanks.

1. Before problem solving, we need to first _____ a problem.
2. An algorithm produces a defined set of _____.
3. A flowchart utilizes various _____ and _____ to map out the order of steps.
4. In flowcharts symbol ◇ is used to show a _____.
5. _____ is used to test the solutions.

## 1.4 Draw the flowcharts for the following problems.

1. Input two numbers *n1* and *n2*. Determine whether *n1* divides *n2* or not.
2. Input a year and determine whether it is a leap year or not.
3. Input a number and calculate its factorial.
4. Find LCM (Lease Common Multiple) of two numbers.
5. Input a number and display its factors.

Design a flowchart to calculate fine amount for Pakistan motorway. Fine is imposed according to the following coding scheme. Input a code and display the respective output.

| Code | Offence | Penalty (Rs.) | Imprisonment |
|---|---|---|---|
| A20 | Driving when disqualified | 1000 | Upto 6 Months |
| A21 | Obtaining or Applying for a driving licence without disclosing particulars of endorsement | 500-1000 | Upto 6 Months |
| A22 | Offence relating to construction of vehicle | 500-1000 | Upto 6 Months |
| A23 | Offence relating to permits | 1000-2000 | Upto 6 Months |
| A24 | Overloading of goods 15 % in excess of permissible limits. | 1000-5000 | Upto 1Month |
| A25 | Overloading of passengers 30 % in excess of permissible limits | 1000-5000 | Upto 1Month |

For more codes, visit http://nhmp.gov.pk

Make the flowchart more comprehensive by adding more codes. Make your algorithm on a chart. Display your chart in your school or community for further awareness on traffic related crimes and their penalties.

Respective teachers can request school administration to arrange a traffic awareness campaign for the community. Students can display their charts as part of the campaign.

## Bibliographic Notes

https://en.wikipedia.org/wiki/Flowchart