Flowchart symbols with meaning, example, in programming, diagram, functions, process

A flowchart uses the special designs or boxes of different shapes of geometry to represent different types of instructions or steps involved in a process. These are called **flowchart symbols**. Here you will see flowchart symbols and meaning with example and diagram.

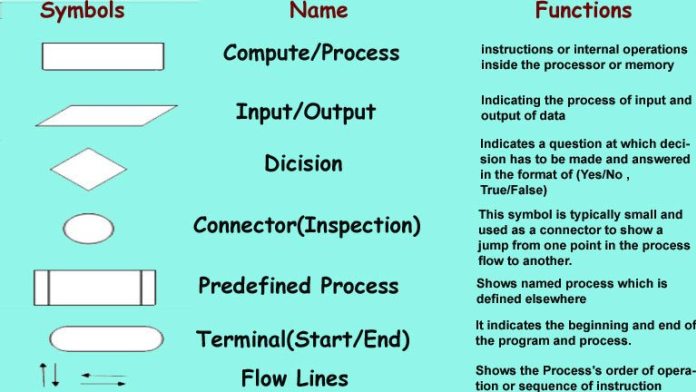
Flowchart symbols are used by a user or a programmer when drawing or making a flowchart whether on paper or online software.

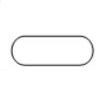
In Flowchart symbols, lines and the arrow shows the sequence of the Process or instructions and the connections among them.

**Programming flowchart symbols**

Now we will discuss some of the most common flowchart symbols in programming or simple use.

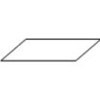
**Flowchart symbols with meaning & their functions.**



**Terminal:**The terminal symbol indicates the Start and End points of the system.

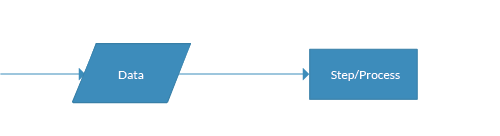
It usually contains the words start, begin, end inside the terminal design/shape to make process/things more obvious.

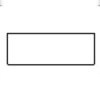
Also, it is the first and the last symbol in program logic. Represented as the shape of the stadium, rounded rectangle, . 

**Input/Output:**It indicates the input and output of data in process/program logic.

Represented as a shape of a Parallelogram.

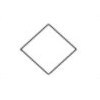
This symbol represents a step where a programmer/user is prompted to enter information/data manually and displaying the result as output.



**Compute/Process:**This symbol represents a set of operations that change value, form, the location of data.

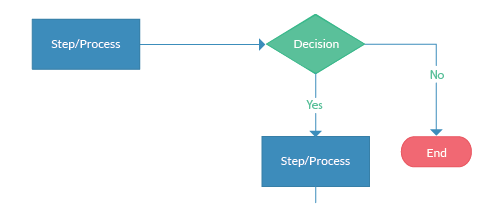
It represents arithmetic and data movement instruction, it is like a workhorse of a flowchart diagram. This symbol looks like a shape of a rectangle.



**Decision:**It indicates a question/point at which the decision has to be made and answered in the format of (yes/No, True/False).

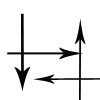
In this symbol, the Flowchart path may then split off into different branches depending on the answer.

This flowchart symbol represented as the shape of Diamond



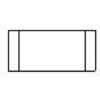
## **Are flowchart symbols difficult?**

**No! Then read ahead!!https://i1.wp.com/www.howtrending.com/wp-content/uploads/2018/12/05557867.jpg?resize=50%2C50&ssl=1**

**Flowlines(Arrowhead):** This Flowchart symbol shows the process’s order of the operation and the exact sequence in which instructions are to be executed in a flowchart.

In a flowchart line coming from one symbol and pointing at another.

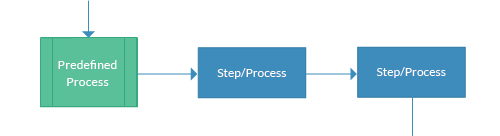
The arrowheads are added to represents the flow, whether it is right-to-left, left-to-right, top-to-bottom, etc.

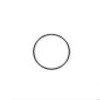
**Predefined Process:** This symbol shows the named process which is defined elsewhere.

It means that the flowchart for the predefined process has to be already drawn, and you should reference the flowchart for more information.

This flowchart symbol represented as a shape of a rectangle with double-struck vertical edges.

When you make a flowchart with the help of a software program, it is also called Subroutine.



**Connector/Inspection:**In Flowchart, this symbol is typically small and used as a connector to show a jump from one point in the process flow to another.

This symbol particularly useful for the large flowchart.

In process mapping, this symbol is full-sized and shows an **inspection** point in the process flow.

This symbol is represented by a small circle with a letter inside.

These connectors usually labeled with capital letters (A, B, AA) to show matching jump points.

# https://creately.com/sites/default/files/objectdetail/examples/sequentialaccess.png

## Advantages of flowchart:

1. The Flowchart is an excellent way of communicating the logic of a program.
2. It is easy and efficient to analyze problem using flowchart.
3. During program development cycle, the flowchart plays the role of a guide or a blueprint. Which makes program development process easier.
4. After successful development of a program, it needs continuous timely maintenance during the course of its operation. The flowchart makes program or system maintenance easier.
5. It helps the programmer to write the program code.
6. It is easy to convert the flowchart into any programming language code as it does not use any specific programming language concept.

### Disadvantage of flowchart

1. The flowchart can be complex when the logic of a program is quite complicated.
2. Drawing flowchart is a time-consuming task.
3. Difficult to alter the flowchart. Sometimes, the designer needs to redraw the complete flowchart to change the logic of the flowchart or to alter the flowchart.
4. Since it uses special sets of symbols for every action, it is quite a tedious task to develop a flowchart as it requires special tools to draw the necessary symbols.
5. In the case of a complex flowchart, other programmers might have a difficult time understanding the logic and process of the flowchart.
6. It is just a visualization of a program, it cannot function like an actual program.

# Difference between High Level and Low level languages

Both **High level language** and **low level language** are the [programming languages’s types](https://www.geeksforgeeks.org/introduction-to-programming-languages/).

The main difference between **high level language** and **low level language** is that, Programmers can easily understand or interpret or compile the high level language in comparison of machine. On the other hand, Machine can easily understand the low level language in comparison of human beings.

Examples of high level languages are [C](http://www.geeksforgeeks.org/c/), [C++](http://www.geeksforgeeks.org/c-plus-plus/), [Java](http://www.geeksforgeeks.org/java/), [Python](https://www.geeksforgeeks.org/python-programming-language/), etc

Let’s see the difference between high level and low level languages:

|  |  |  |
| --- | --- | --- |
| **S.NO** | **HIGH LEVEL LANGUAGE** | **LOW LEVEL LANGUAGE** |
| 1. | It is programmer friendly language. | It is a machine friendly language. |
| 2. | High level language is less memory efficient. | Low level language is high memory efficient. |
| 3. | It is easy to understand. | It is tough to understand. |
| 4. | It is simple to debug. | It is complex to debug comparatively. |
| 5. | It is simple to maintain. | It is complex to maintain comparatively. |
| 6. | It is portable. | It is non-portable. |
| 7. | It can run on any platform. | It is machine-dependent. |
| 8. | It needs compiler or interpreter for translation. | It needs assembler for translation. |