Flowchart guidelines

Part 2 Computational Physics

May 2, 2016

THE UNIVERSITY OF
MELBOURNE

Introduction

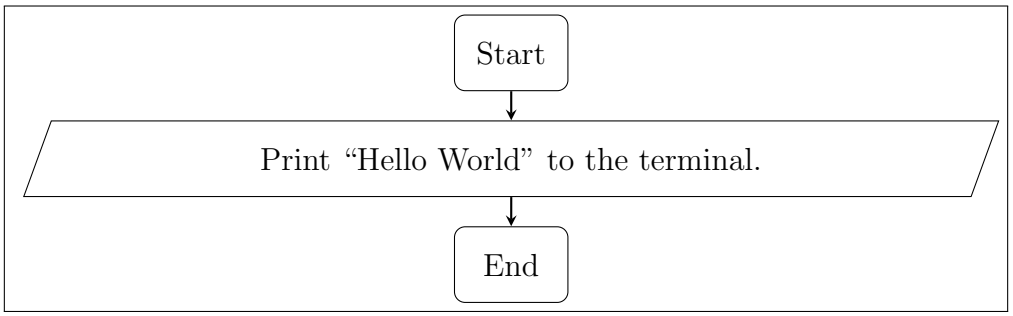
Here we give some tips and a brief description of what is expected in your flowcharts.

Your flowcharts should contain all the **relevant** information that explains the **logic** of your code.

You do **NOT** need to list the libraries you include with the `#include` statements, or THAT you define a main or other functions. You DO need to show how these functions **operate** and how/when you **call** them. You need to show the logical flow of the program. The nitty-gritty of how your code functions is the most important thing. If your program just does one thing, then your flow chart is simple.

Linear flowcharts

For example the flowchart for the "Hello World" flowchart can be as simple as:

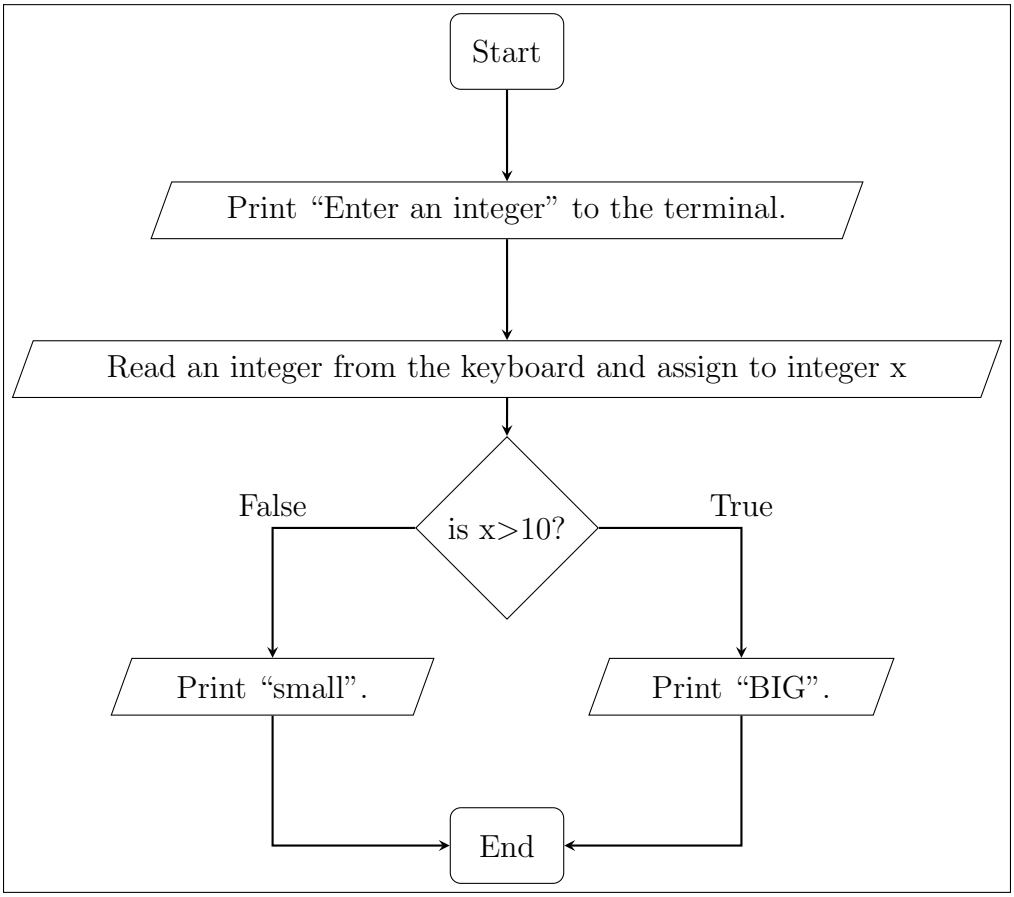


As the logic just follows a sequence of events, the flowchart is linear. Even if we were to have a more complicated list of operations, the flowchart will be linear in all cases except when conditionals (`if`-statements), recursion, or loops are used.

Branching flow charts

If there are any loops or `if`-statements anywhere in your code, your flowchart MUST branch; it cannot be a linear sequence of operations.

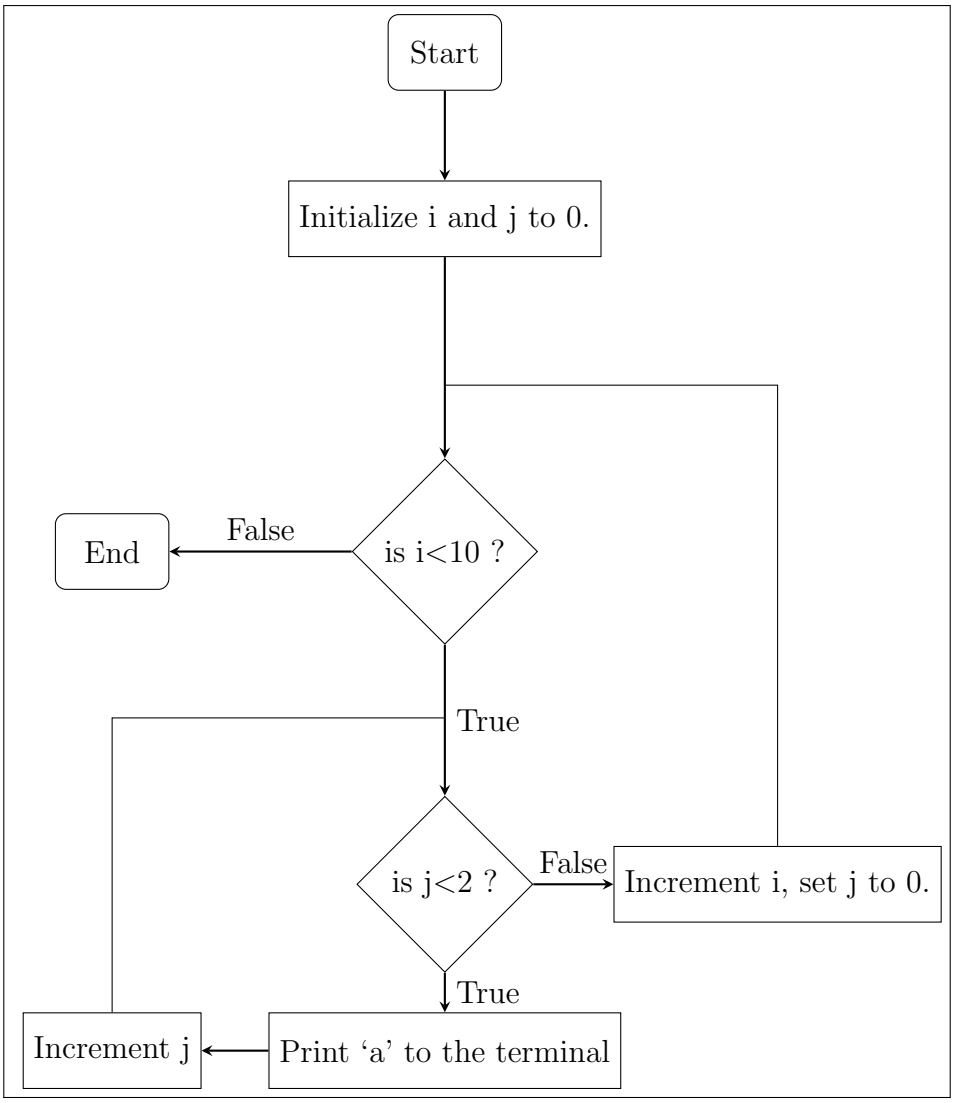
For example, reading a number from the keyboard, and printing an output depending on whether or not the number is bigger than 10:



For small programs it is best to include every step. As soon as your programs become more involved, you can gloss over steps that don't help to understand how the function/program operates. For example in the above, the second operation, printing a prompt to the user, is not necessary information and can be merged with the third, whereas all the other steps are necessary.

Representing loops

A common mistake people make is when their code includes `for` loops but their flowchart doesn't show this. Here is an example of nested `for`-loops; a double `for`-loop that prints 'a' twenty times:



Common errors

Please do not write your C code in the flow chart. The idea is to explain the program logic using English. Using symbols as shorthand is fine, but do not simply copy your code into the flow chart. '`int i=0`' is fine (although ideally you would explicitly state that you are initializing i to 0) but this is as much C code as is acceptable. You should NEVER have '`for(i=0;i<10;i++)`' in your flowchart. You need to go through the logic in **words**.

A common mistake is to reference variables before they are ever defined. If you use a variable then you have to create it first. For example checking if $i < 10$ would have no meaning if i hasn't been assigned a value before that point in the flowchart.

Other common errors include using the wrong box type. A very common example of this is using a diamond box when no decision is being made (or not using one when a decision is being made).

You must also show all the relevant variable assignments. A common mistake is to update a variable in the code but not show this in the flow-chart. For example, in the Fibonacci sequence, not explicitly showing that you reassign the three values as you move along the sequence.

You can and should *modulate* your flowcharts, meaning reference a flowchart within a flowchart. If you call a function more than once, or it is messy to include in the main flow of your flowchart, you should draw the flowchart for that function separately. You can then reference this function (in a square/processing box) from other flowcharts.