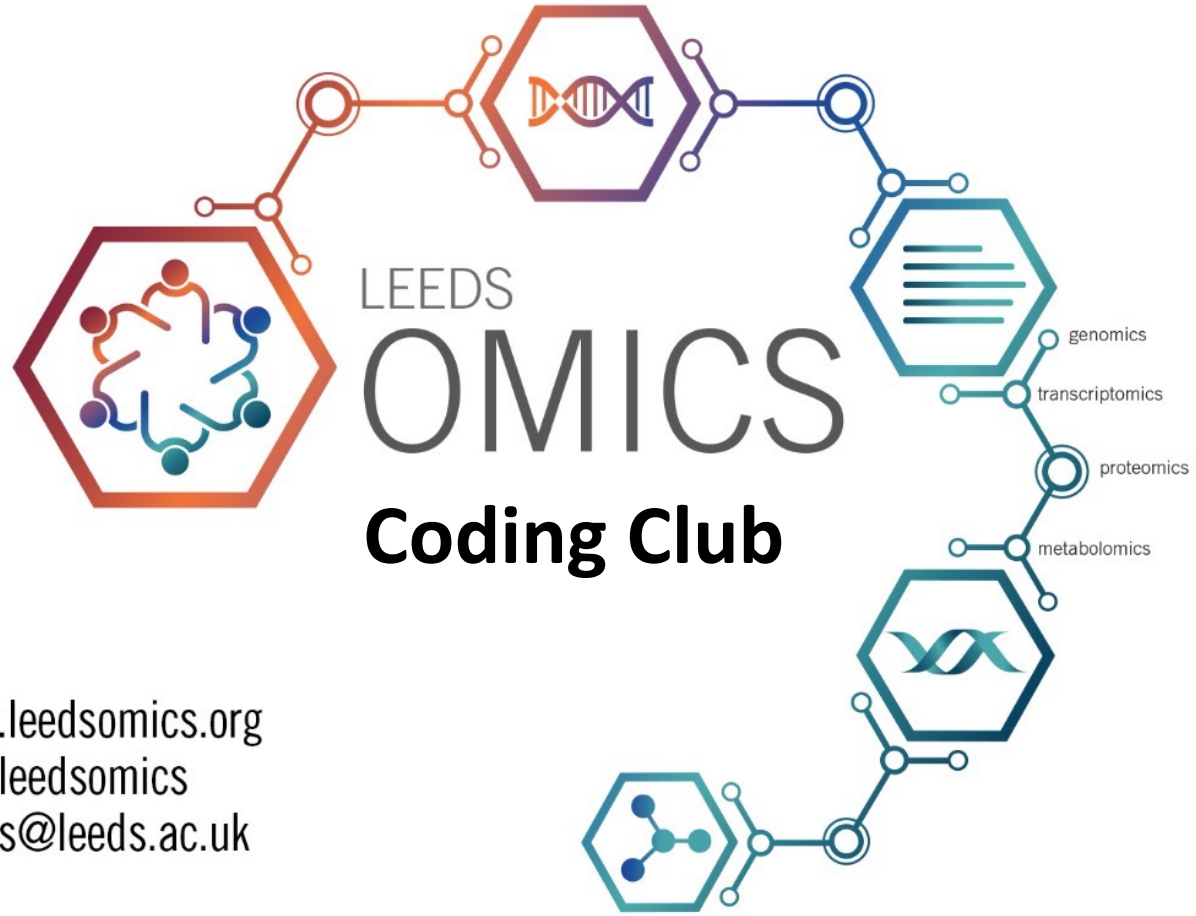


Coding Club

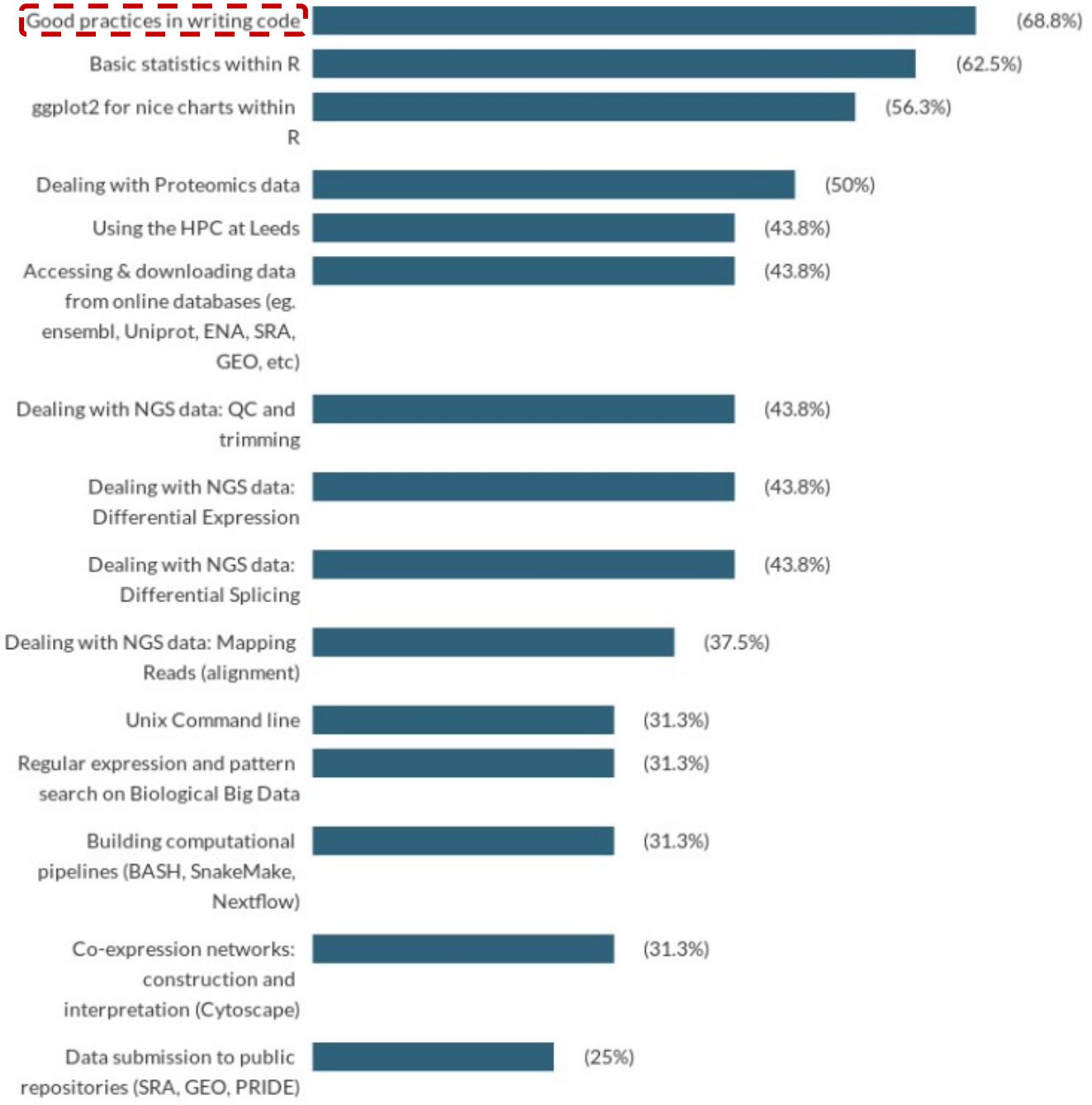
www.leedsomics.org
@leedsomics
omics@leeds.ac.uk

Good practices in writing code

Club Moderators: Elton Vasconcelos and Euan McDonnell

Topics to be addressed - **Survey Result (2021-22)**

1st session



Common practice before writing the actual code

How to write a Pseudo Code?

Pseudo code is a term which is often used in programming and algorithm based fields. It is a methodology that allows the programmer to represent the implementation of an algorithm. Simply, we can say that it's the cooked up representation of an algorithm. Often at times, algorithms are represented with the help of pseudo codes as they can be interpreted by programmers no matter what their programming background or knowledge is. Pseudo code, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge.

Algorithm: It's an organized logical sequence of the actions or the approach towards a particular problem. A programmer implements an algorithm to solve a problem. Algorithms are expressed using natural verbal but somewhat technical annotations.

Pseudo code: It's simply an implementation of an algorithm in the form of annotations and informative text written in plain English. It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer.

Advantages of Pseudocode

- Improves the readability of any approach. It's one of the best approaches to start implementation of an algorithm.
- Acts as a bridge between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential. And that's where a pseudo-code proves vital.
- The main goal of a pseudo code is to explain what exactly each line of a program should do, hence making the code construction phase easier for the programmer.

https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/

Biological context example: Systematic detection of whether blast hits are sense or antisense

`Blast tabular output format (12 columns by default)`

1.	qseqid	query (e.g., gene) sequence id
2.	sseqid	subject (e.g., reference genome) sequence id
3.	pident	percentage of identical matches
4.	length	alignment length
5.	mismatch	number of mismatches
6.	gapopen	number of gap openings
7.	qstart	start of alignment in query
8.	qend	end of alignment in query
9.	sstart	start of alignment in subject
10.	send	end of alignment in subject
11.	evalue	expect value
12.	bitscore	**bit score**

Biological context example: Systematic detection of whether blast hits are sense or antisense

Pseudocode

```
Open blast_tabular_output file
While line is not equal to null string        #read the whole file, line by line until it ends
        If sbjct_start_coord is less then sbjct_end_coord
                Print "sense"
        Else
                Print "antisense"
Close file
```

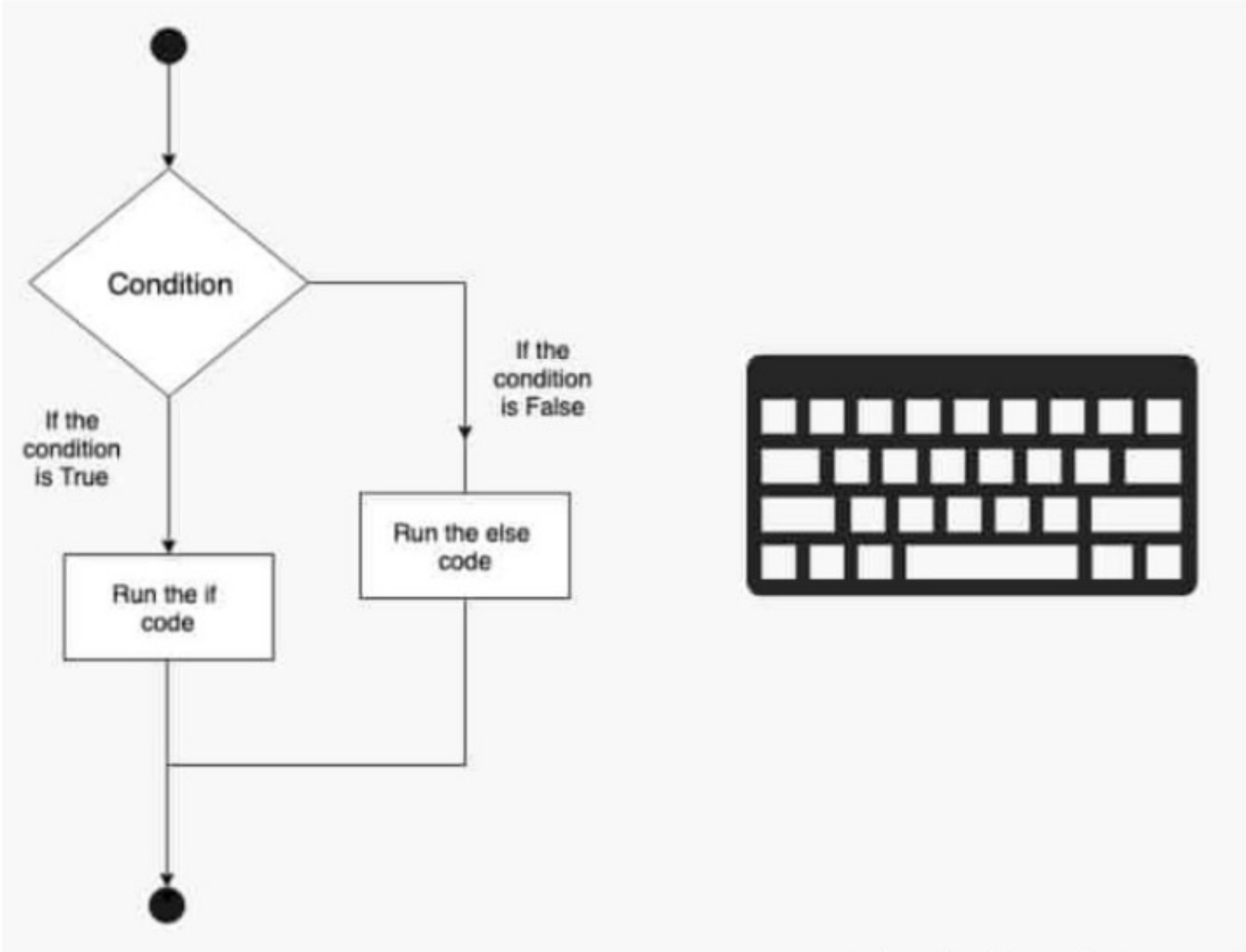
Code

```perl
#!/usr/bin/perl
# Programmer: Elton Vasconcelos (05.Sep.2013)
# Usage: perl blast-sense_antisense-detector.pl [blastN_tabular_output.tab] >outfile.tab
################################################################################################
# IMPORTANT Notes: SENSE and ANTISENSE assignments will be always regarding the query orientation onto the subject
# on the blast output. You ought to know whether your Blast-DB sequences are all in the correct orientation (.g. *AnnotatedTranscripts.fast
# *AnnotatedCDSs.fasta files from TriTrypDB are all good to run this script)
# --> If your blast analysis was against a whole genome DB, the "SENSE" assignment will mean "Forward Strand" and the "ANTIsense" assignmen
# will mean "Reverse Strand" on the given chromosome where the query was aligned.

open (FILE, "$ARGV[0]") or die ("Can't open infile $ARGV[0]\n");
my $line = <FILE>;
chomp($line);
my @array;
while ($line ne "") {
    @array = split(/\t/,$line);
    if ($array[8] < $array[9]) { #$array[8] and [9] are the columns where the sbjct coordinates are placed
        print("$line\tSENSE\n");
    }
    else {
       print("$line\tANTIsense\n");
    }
    $line = <FILE>;
    chomp($line);
}
close(FILE);
```

- **Conditional Statements in Programming**

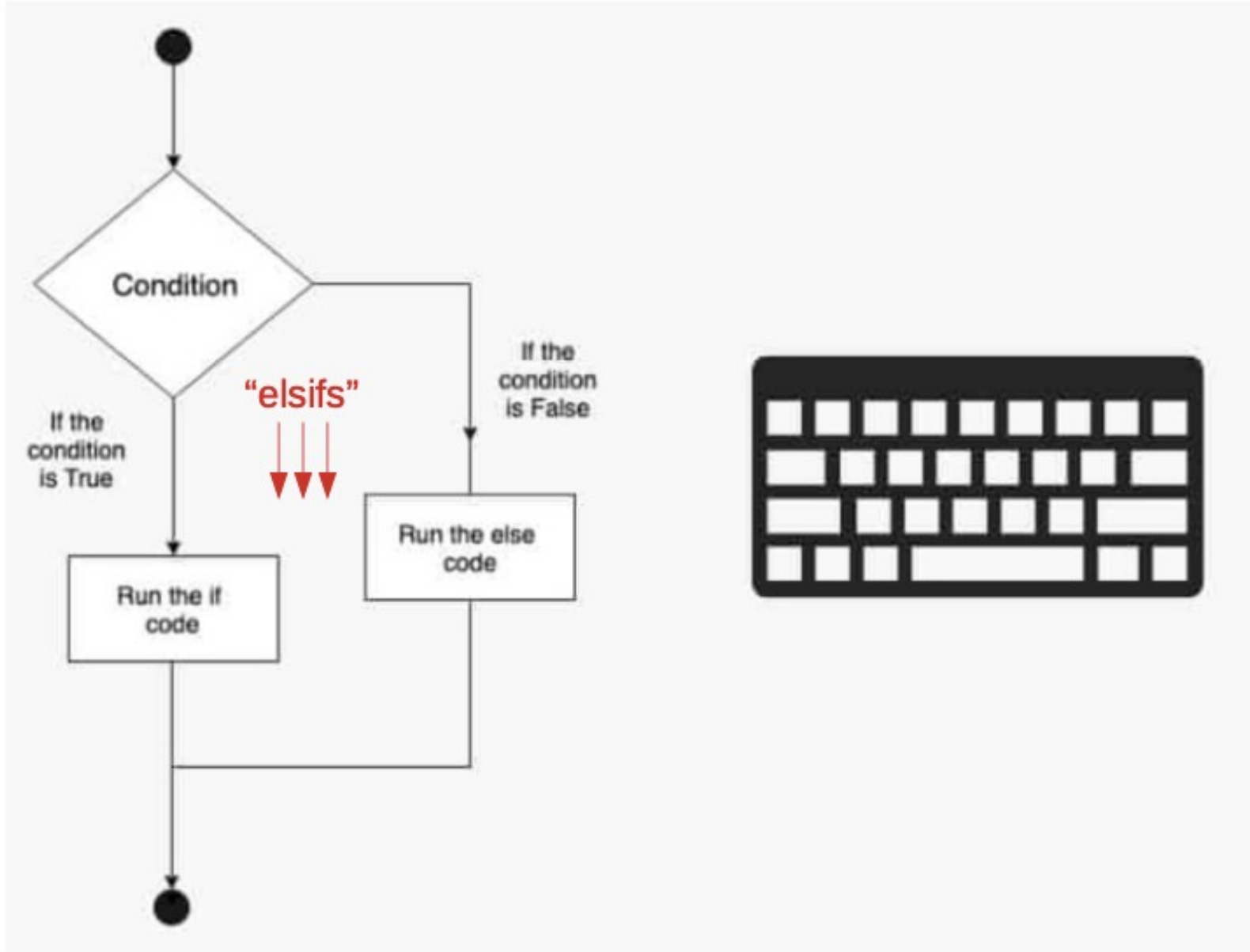
```
if …
else …
```



https://python-tricks.com/conditions-in-python/

- **Conditional Statements in Programming**

```
if …
elsif ...
elsif ...
elsif ...
else …
```

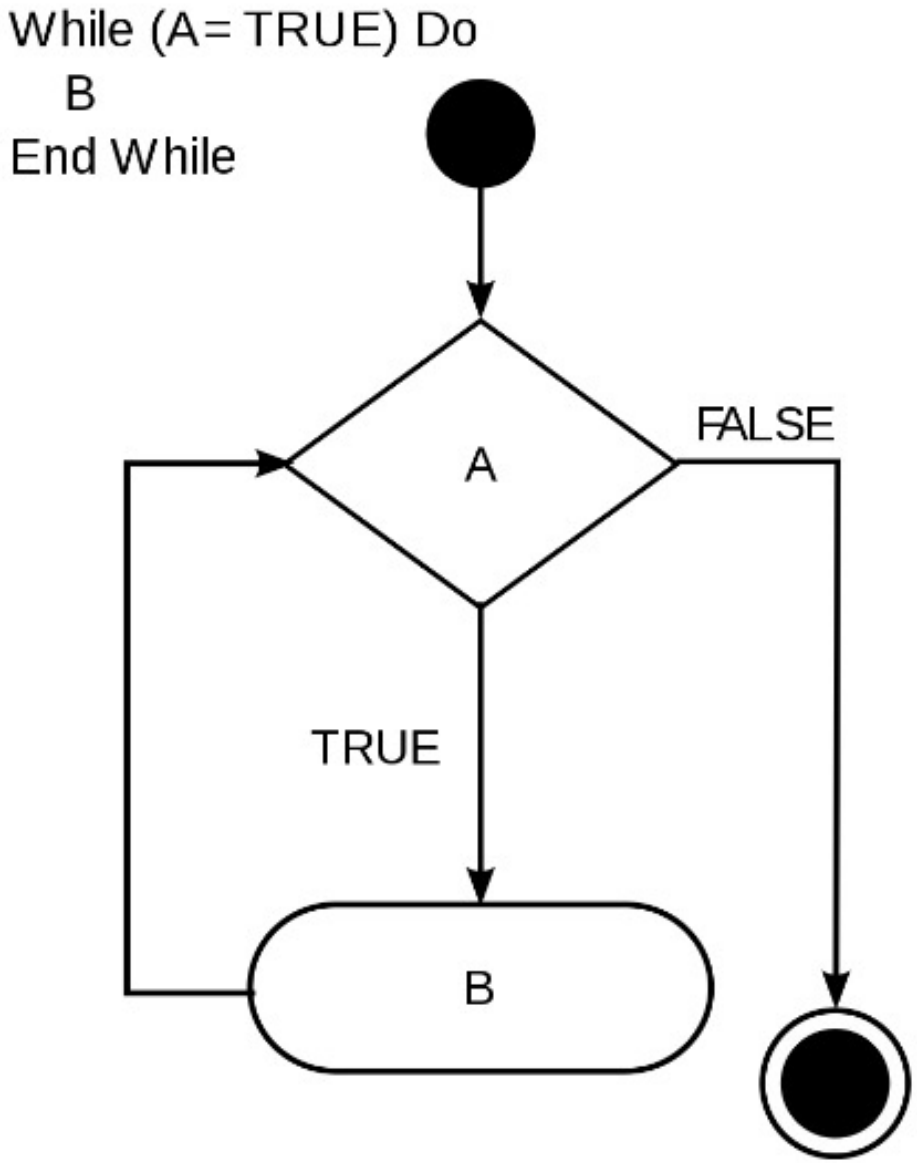


https://python-tricks.com/conditions-in-python/

- **Loopings**

==> control flow statement that allows a code to be executed repeatedly based on a given Boolean condition

```
while …
until …
for …
```



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

- **Variable in programming** → programmer-predefined entity that will store information in the code

Different types of variables/data in the three main languages used in Bioinformatics

PERL

- Scalars ($)
- Arrays (@)
- Hashes (%)

Python

- Numbers
- String
- List
- Tuple
- Dictionary

R

- Numeric
- Character
- Vectors
- Factors
- Data Frames

https://www.tutorialspoint.com/perl/perl_variables.htm

https://www.tutorialspoint.com/python3/python_variable_types.htm

https://www.tutorialspoint.com/r/r_data_types.htm

NOTE: List and Array terminologies in R have different and more complex concepts than in PERL and Python.

Bring your issues on!