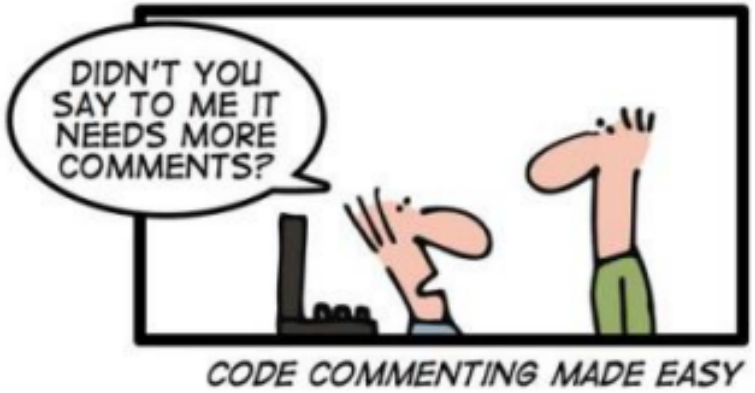
Documentation and Man Pages

```
Module Code: COMP1712

Module Name: Computer Architectures and Operating Systems

Credits: 15

Module Leader: Seb Blair BEng(H) PGCAP MIET MIHEEM FHEA
```

Download as a PDF

Why Documentation

- **You**
  - put down the project and return to it much later
  - want people to use it and give you credit
- **Others**
  - would be encouraged to contribute
  - more easily use your code
- **Science / Engineering**
  - Advances
  - Open collaboration
  - Reproducibility and transparency



README.md

A brief description of the project

- Installation instructions
- A short example/tutorial
- Contributors
- Licenses
- Citations
- Contacts

Project Title

Describtion

One Paragraph of project description goes here

Prerequisites

- List all the depnedancies
- List what to install and how to install it

Installation

- A step by step instructions on how to install the software

```
Example that shows how the software works
```

Contributing

Issue Tracker: github.com/project/issues

License

Provide Licensing information

Citation

- How this software can be cited.
- Provide a DOI that was generated.

Contact

- Link to e-mail addresses or URIs

Tools for Documentantion

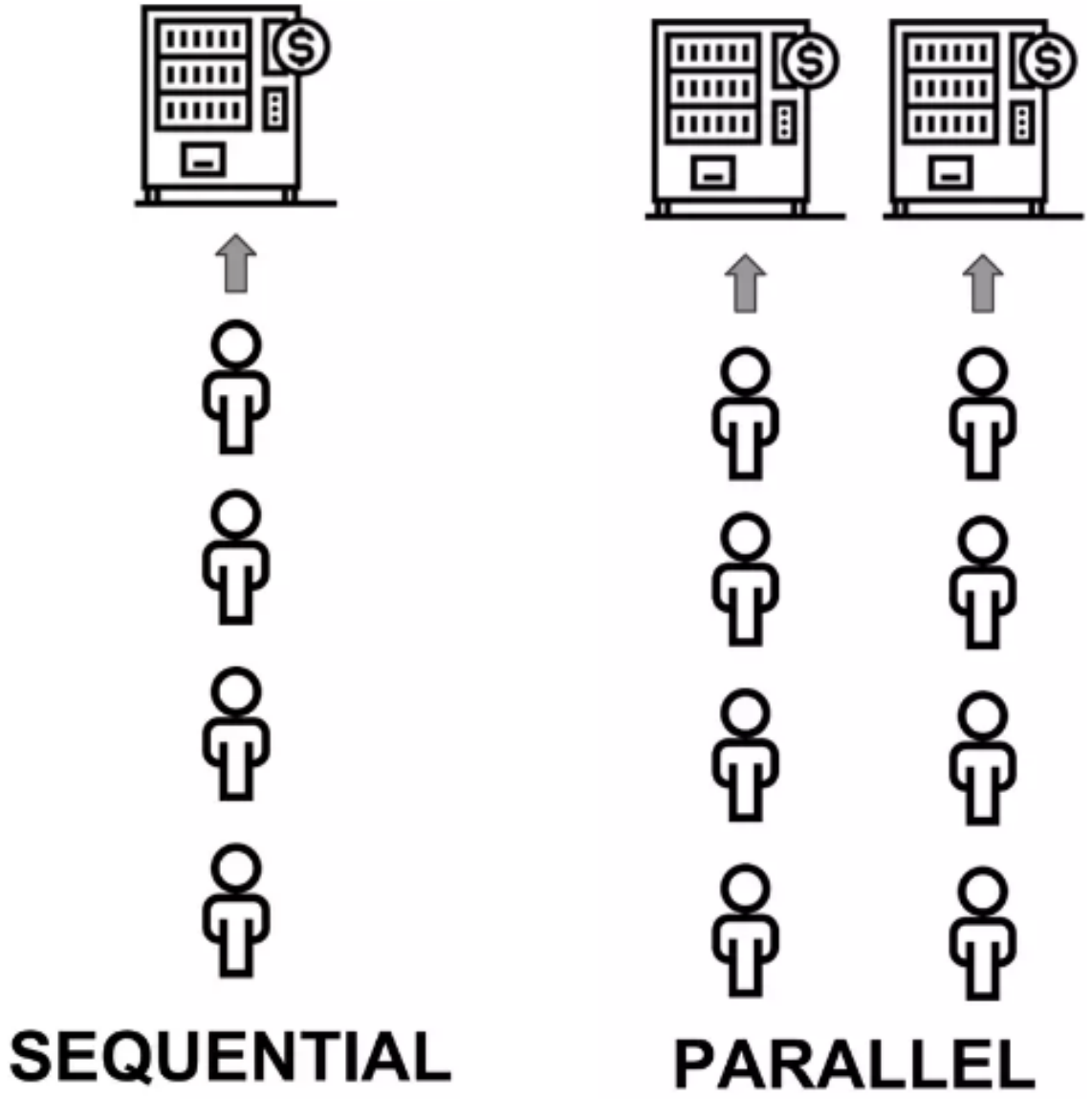
- **Python**
  - Sphinx, Doctest, Numpydoc
- **R**
  - R Markdown, Kite
- **C++**
  - BoostBook, QuickBook, GhostDoc
- **Java**
  - Javadoc
- **Ruby**
  - Docurium
- **Doxygen**
  - C, C#, PHP, Java, Python, and Fortran.

Tools for Documentation for Bash

- `shdoc` (Shell Documentation)
  - converts comments written according to the API into markdown
  - Community project,https://github.com/reconquest/shdoc
  - `shdoc < somescript.sh > somescriptdoc.md`
- Plain Old Documentation (POD)
  - lightweight markup language written in `perl`
  - `perldoc -F somescript.sh -oman -d somescriptdoc.gz`

Divergence Dilemma

- bash scripts are documented in seperate manual files `man-pages`.
- `man-pages` are standard and well established
- as with all documentation code develops faster and is released, thus creates a divergence, as in code <-> documentation become out of sync.



Literate Programming

- a computer program is given as an explanation of how it works in a natural language, such as English, interspersed (embedded) with snippets of macros and traditional source code, from which compilable source code can be generated.

`man-pages`

- system utility and allows you to read what a program, utility or function does, its arguments, outputs and general behaviour.

```
MAN(1)                                Manual pager utils                                MAN(1)

NAME
       man - an interface to the system reference manuals

SYNOPSIS
       man [man options] [[section] page ...] ...
       man -k [apropos options] regexp ...
       man -K [man options] [section] term ...
       man -f [whatis options] page ...
       man -l [man options] file ...
       man -w|-W [man options] page ...

DESCRIPTION
       man  is  the  system's manual pager.  Each page argument given to man is normally the name of a program, utility or func-
       tion.  The manual page associated with each of these arguments is then found and displayed.  A section, if provided, will
       direct  man to look only in that section of the manual.  The default action is to search in all of the available sections
       following a pre-defined order (see DEFAULTS), and to show only the first page found, even if page exists in several  sec-
       tions.

       The table below shows the section numbers of the manual followed by the types of pages they contain.

       1   Executable programs or shell commands
       2   System calls (functions provided by the kernel)
       3   Library calls (functions within program libraries)
       4   Special files (usually found in /dev)
       5   File formats and conventions, e.g. /etc/passwd
       6   Games
       7   Miscellaneous (including macro packages and conventions), e.g. man(7), groff(7), man-pages(7)
       8   System administration commands (usually only for root)
       9   Kernel routines [Non standard]

       A manual page consists of several sections.

       Conventional  section  names  include NAME, SYNOPSIS, CONFIGURATION, DESCRIPTION, OPTIONS, EXIT STATUS, RETURN VALUE, ER-
       RORS, ENVIRONMENT, FILES, VERSIONS, CONFORMING TO, NOTES, BUGS, EXAMPLE, AUTHORS, and SEE ALSO.

       The following conventions apply to the SYNOPSIS section and can be used as a guide in other sections.

       bold text          type exactly as shown.
       italic text        replace with appropriate argument.
       [-abc]             any or all arguments within [ ] are optional.
       -a|-b              options delimited by | cannot be used together.
       argument ...       argument is repeatable.
       [expression] ...   entire expression within [ ] is repeatable.

       Exact rendering may vary depending on the output device.  For instance, man will usually not be able  to  render  italics
       when running in a terminal, and will typically use underlined or coloured text instead.

       The command or function illustration is a pattern that should match all possible invocations.  In some cases it is advis-
       able to illustrate several exclusive invocations as is shown in the SYNOPSIS section of this manual page.
```

`man-pages` Sections

- Each page is stored in a designated section ->
- So where are Man pages stored?
  `/usr/share/man/man#/page.#.gz`

Section Number	Subject
1	User commands
2	System calls
3	Library functions
4	Devices
5	Files in /dev, configuration files, and drivers
6	Games and screensavers
7	Miscellaneous
8	System administration commands
9	Kernel routines and daemons

`man-pages` Layout

NAME

- The name of the command or function, followed by a one-line description of what it does.

SYNOPSIS

- In the case of a command, a formal description of how to run it and what command line options it takes.

DESCRIPTION

- A textual description of the functioning of the command or function.

EXAMPLES

- Some examples of common usage.

SEE ALSO

- A list of related commands or functions.

Other sections may be present, but these are not well standardised across man pages. Common examples include: **OPTIONS**, **EXIT STATUS**, **RETURN VALUE**, **ENVIRONMENT**, **BUGS**, **FILES**, **AUTHOR**, **REPORTING**

`man-pages` Syntax

```
.TH CORRUPT 1
.SH NAME
corrupt \- modify files by randomly changing bits
.SH SYNOPSIS
.B corrupt
[\fB\-n\fR \fIBITS\fR]
[\fB\-\-bits\fR \fIBITS\fR]
.IR file ...
.SH DESCRIPTION
.B corrupt
modifies files by toggling a randomly chosen bit.
.SH OPTIONS
.TP
.BR \-n ", " \-\-bits =\fIBITS\fR
Set the number of bits to modify.
Default is one bit
```

```
$ gzip < corrupt.1 > corrupt.1.gz
$ man -l corrupt.1.gz
```

```
CORRUPT(1)

NAME
    corrupt - modify files by randomly changing bits

SYNOPSIS
    corrupt [-n BITS] [–bits BITS] file…

DESCRIPTION
    corrupt modifies files by toggling a randomly chosen bit.

OPTIONS
    -n, –bits=BITS
        Set the number of bits to modify. Default is one bit.

CORRUPT(1)
```