

Downloading basic hello package using Linux using non-interactive network:

$ man wget

Wget id the non interactive network download

$ wget <ftp://ftp.gnu.org/gnu/hello/hello-2.10.tar.gz>

$ tar xzf hello-2.10.tar.gz

$ cd hello-2.10

$ man ./hello.1

Man page for the hello package

$ ./configure

It will check for all the dependencies that are required for the build and install process. After configuring it will create a make file.

$ less Makefile

$ make

It will run the make file in the root directory of the package and it may call other make files as well.

$ sudo make install

It will do the installation.

$ which hello

Gives the path where hello package go installed

$ hello

$ man hello

Now we can get the man page directly.

$ sudo make uninstall

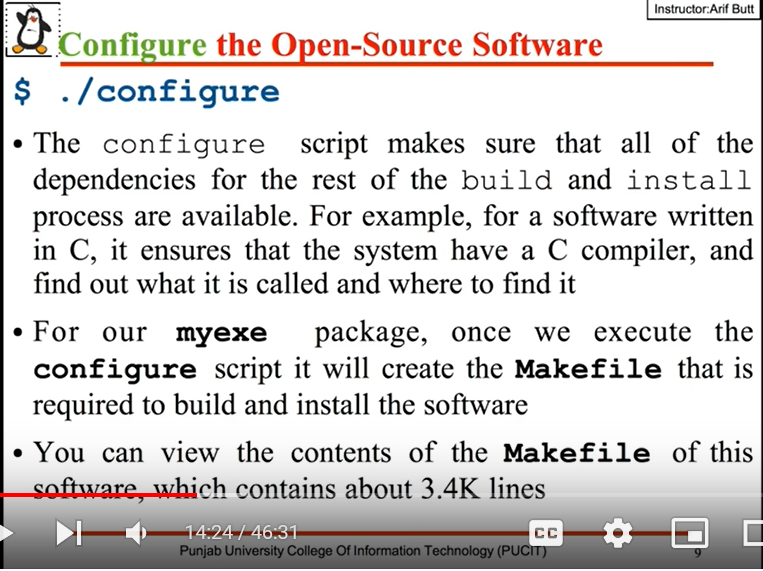
To uninstall the targets

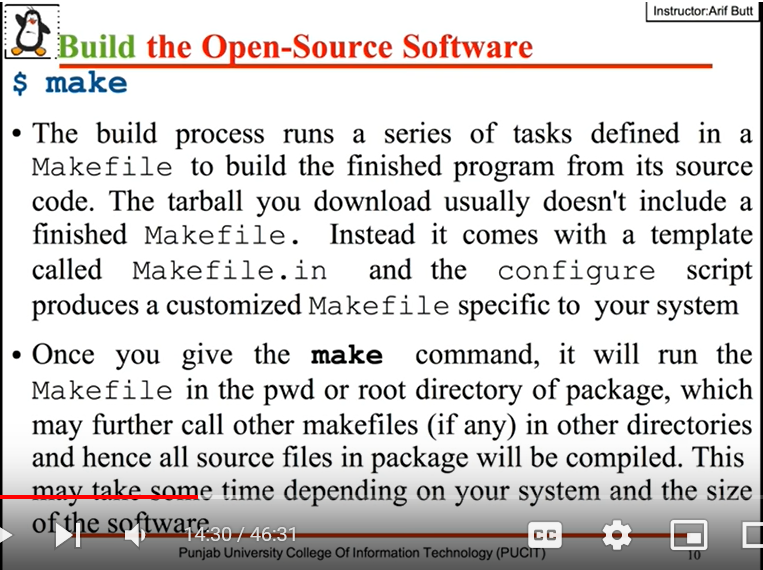
$ which hello

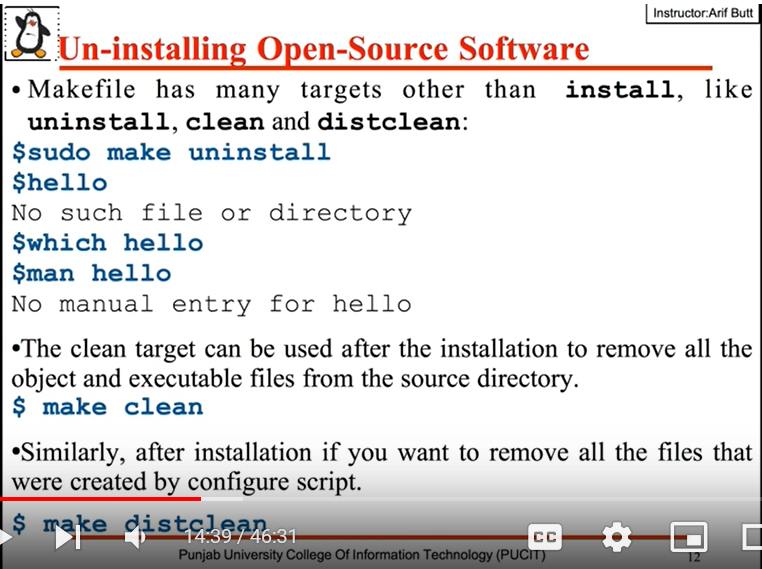
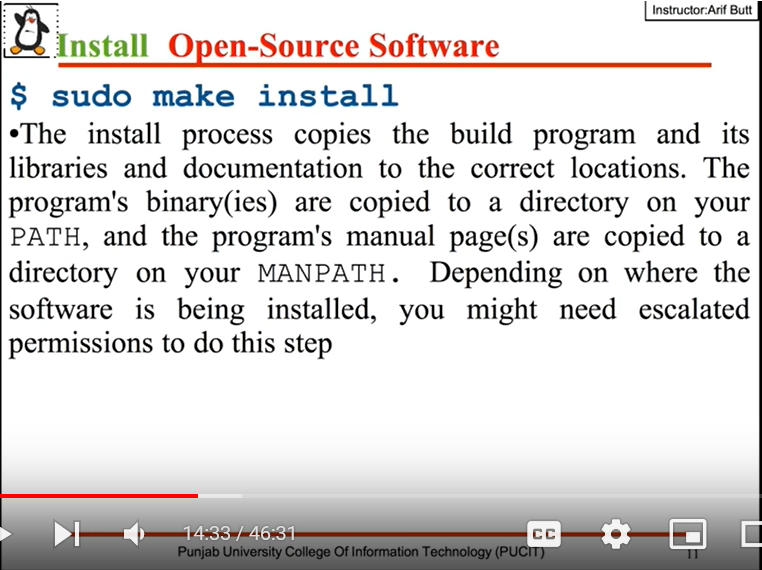
$ man hello

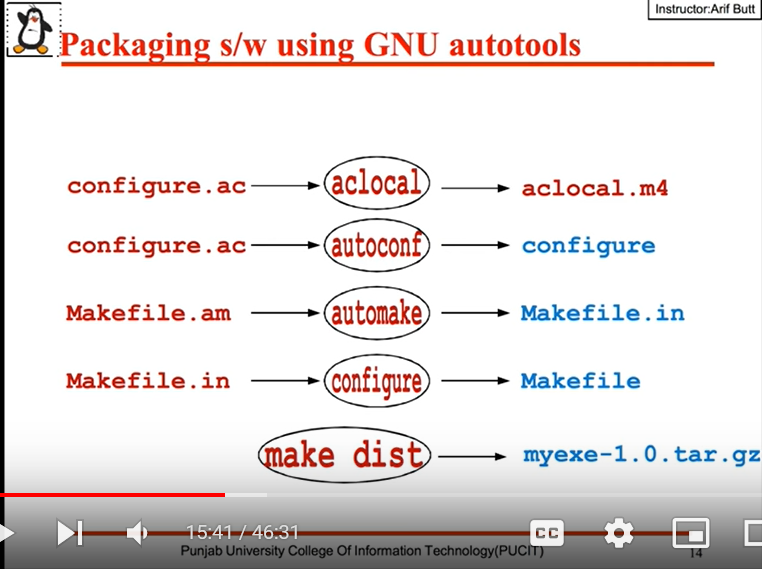
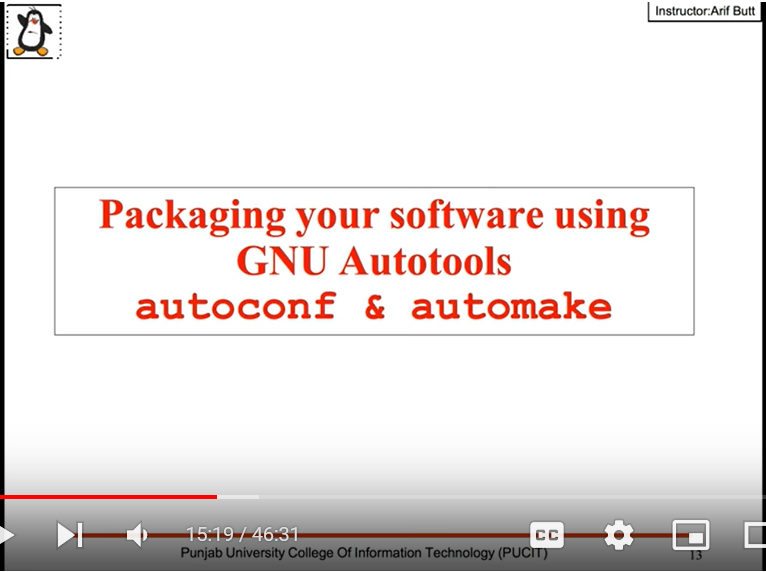
$ hello

Now it will not work.









$ sudo apt-get install autoconf

$ sudo yum install autoconf

$ autoconf –version

$ sudo apt-get install automake

$ sudo yum install automake

$ automake --version

Note: mymath.h file contains prototypes/signatures of the functions. mysub.c, mydiv.c, myadd.c, mymyl.c contains function definitions. And prog1.c is driver program.

Create configure.ac file which contains basic macros.

$ cat configure.ac

$ ls /usr/share/autoconf/autoconf/

Here we can see lot of m4 macros.

$ aclocal

It will read the configure.ac file and it will create aclocal.m4 file which contains macros and autom4te.cache dir which is required for auroconf.

$ autoconf

It will generate the configure script.

Now crate Make file

$ cat Makefile.am

$ automake

$ automake --add-missing

Now Makefile.am will create Makefile.in file

$ ./configure

Now it will generate Makefile

$ less Makefile

$ make dist

To make the distribution. We can see myexe-1.0.tar.gz file has been created.

$ make distcheck

It will check for the distribution whether it has been correctly prepared or not.

Not we can share our distribution with others. For testing copy the distribution in temp file unzip it and try to install it.

$ ./configure && make && sudo make install

To execute all installation commands in one single go.

Not myexe file is crated.

$ ./myexe

$ myexe

$ which myexes

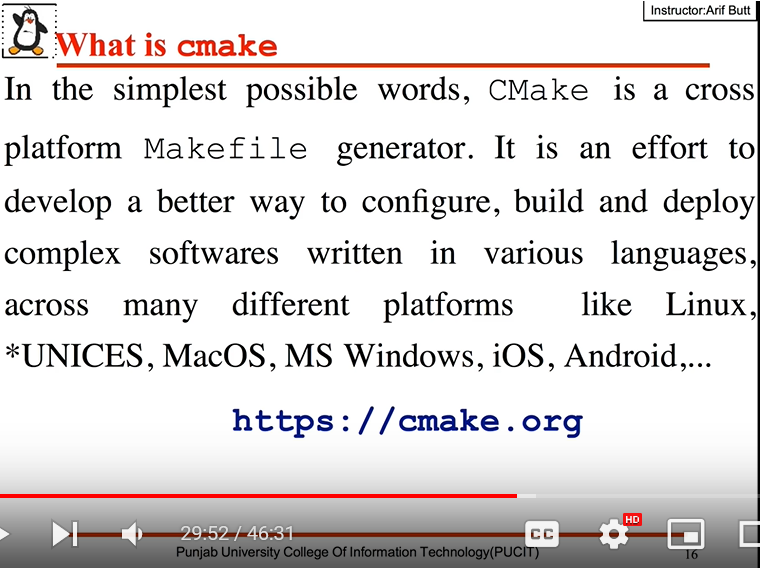
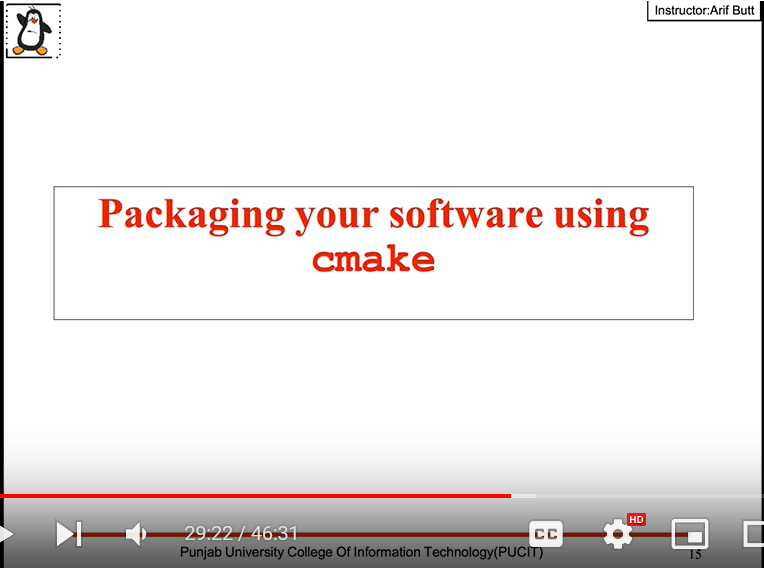
No myexe is a command in our Linux system. It got installed in our usr/bin/ directory.

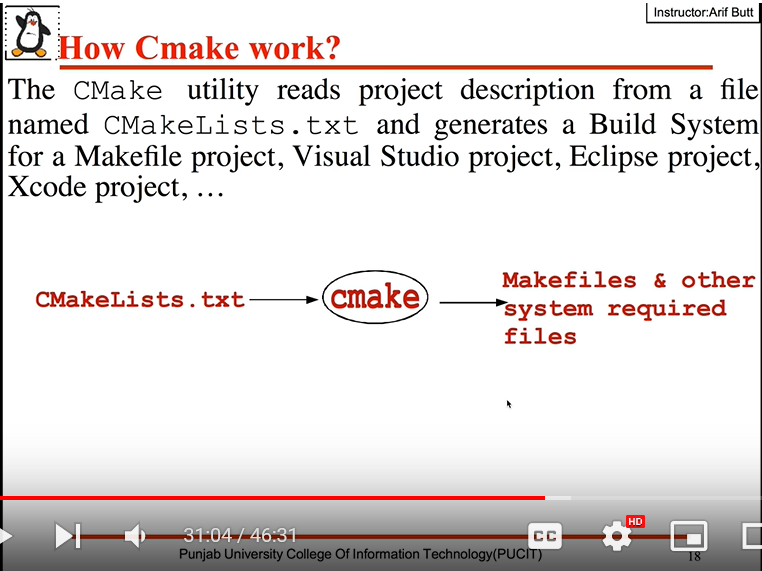
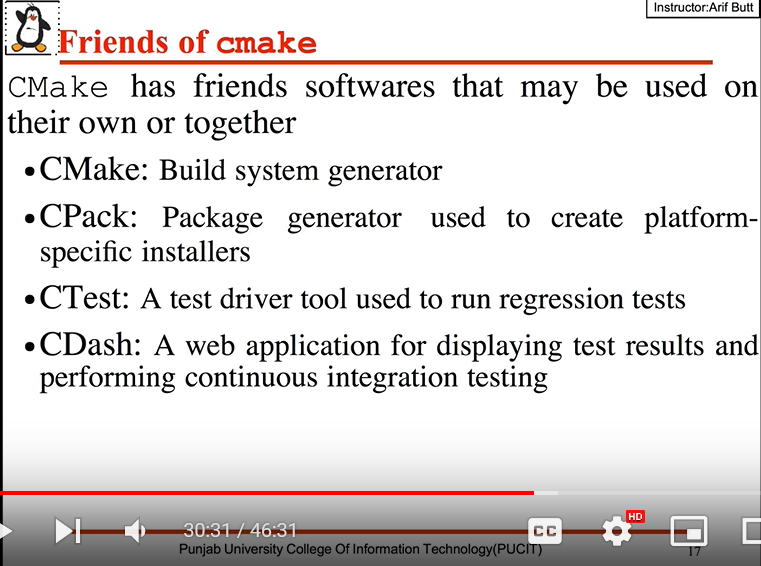
$ sudo make uninstall

To uninstall the command

$ make distclean

It will clear all the generated files.





$ sudo apt-get install cmake -y

$ sudo apt-get install cmake-qt-gui -y

$ man cmake

$ ar -t lib/libarifmat.a

To see the object files

Make a dir build

$ mkdir build

There are two options when ever we are building our S/W(outer source build and inner source build). Outer sourced build has three main advantages.

1. All the generated files remains separated from the source files.
2. We can have several build trees for the same source tree.
3. We can delete existing build and perform a clean build again.

$ cat CMakeLists.txt

Check the content in the file

$ cmake --help-command-list

$ cmake --help-command project | less

Go to build directory and do cmake. It will create Makefile

$ cd build

$ cmake ../

$ make

$ sudo make install

Now myexe file has been created

$ ./myexe

$ myexe

$ which myexe

$ man myadd

$ cpack --config CPackSourceConfig.cmake

For creating packages. It will crate compressed file which we can distribute with the community.