Makefile - Macros

The **make** program allows you to use macros, which are similar to variables. Macros are defined in a Makefile as = pairs. An example has been shown below −

```
MACROS  = -me
PSROFF  = groff -Tps
DITROFF = groff -Tdvi
CFLAGS  = -O -systype bsd43
LIBS    = "-lncurses -lm -lsdl"
MYFACE  = ":*)"
```

Special Macros

Before issuing any command in a target rule set, there are certain special macros predefined −

- $@ is the name of the file to be made.
- $? is the names of the changed dependents.

For example, we could use a rule as follows −

```
hello: main.cpp hello.cpp factorial.cpp
   $(CC) $(CFLAGS) $? $(LDFLAGS) -o $@

Alternatively:

hello: main.cpp hello.cpp factorial.cpp
   $(CC) $(CFLAGS) $@.cpp $(LDFLAGS) -o $@
```

In this example, $@ represents *hello* and $? or $@.cpp picks up all the changed source files.

There are two more special macros used in the implicit rules. They are −

- $< the name of the related file that caused the action.
- $* the prefix shared by target and dependent files.

Common implicit rule is for the construction of .o (object) files out of .cpp (source files).

```
.cpp.o:
   $(CC) $(CFLAGS) -c $<

Alternatively:
```

```
.cpp.o:
   $(CC) $(CFLAGS) -c $*.c
```

Conventional Macros

There are various default macros. You can see them by typing "make -p" to print out the defaults. Most are pretty obvious from the rules in which they are used.

These predefined variables, i.e., macros used in implicit rules fall into two classes. They are as follows −

- Macros that are names of programs (such as CC)
- Macros that contain arguments of the programs (such as CFLAGS).

Below is a table of some of the common variables used as names of programs in built-in rules of makefiles −

Sr.No	Variables & Description
1	**AR** Archive-maintaining program; default is `ar'.
2	**AS** Program to compiling assembly files; default is `as'.
3	**CC** Program to compiling C programs; default is `cc'.
4	**CO** Program to checking out files from RCS; default is `co'.
5	**CXX** Program to compiling C++ programs; default is `g++'.
6	**CPP** Program to running the C preprocessor, with results to standard output; default is `$(CC) -E'.
7	**FC** Program to compiling or preprocessing Fortran and Ratfor programs; default is `f77'.
8	**GET** Program to extract a file from SCCS; default is `get'.
9	**LEX** Program to use to turn Lex grammars into source code; default is `lex'.
10	**YACC** Program to use to turn Yacc grammars into source code; default is `yacc'.
11	**LINT**

	Program to use to run lint on source code; default is `lint'.
12	**M2C** Program to use to compile Modula-2 source code; default is `m2c'.
13	**PC** Program for compile Pascal programs; default is `pc'.
14	**MAKEINFO** Program to convert a Texinfo source file into an Info file; default is `makeinfo'.
15	**TEX** Program to make TeX dvi files from TeX source; default is `tex'.
16	**TEXI2DVI** Program to make TeX dvi files from Texinfo source; default is `texi2dvi'.
17	**WEAVE** Program to translate Web into TeX; default is `weave'.
18	**CWEAVE** Program to translate C Web into TeX; default is `cweave'.
19	**TANGLE** Program to translate Web into Pascal; default is `tangle'.
20	**CTANGLE** Program to translate C Web into C; default is `ctangle'.
21	**RM** Command to remove a file; default is `rm -f'.

Here is a table of variables whose values are additional arguments for the programs above. The default values for all of these is the empty string, unless otherwise noted.

Sr.No.	Variables & Description
1	**ARFLAGS** Flags to give the archive-maintaining program; default is `rv'.
2	**ASFLAGS** Extra flags to give to the assembler when explicitly invoked on a `.s' or `.S' file.
3	**CFLAGS** Extra flags to give to the C compiler.
4	**CXXFLAGS** Extra flags to give to the C compiler.
5	**COFLAGS** Extra flags to give to the RCS co program.
6	**CPPFLAGS** Extra flags to give to the C preprocessor and programs, which use it (such as C and Fortran compilers).
7	**FFLAGS** Extra flags to give to the Fortran compiler.
8	**GFLAGS** Extra flags to give to the SCCS get program.
9	**LDFLAGS** Extra flags to give to compilers when they are supposed to invoke the linker, `ld'.
10	**LFLAGS** Extra flags to give to Lex.
11	**YFLAGS**

	Extra flags to give to Yacc.
12	**PFLAGS** Extra flags to give to the Pascal compiler.
13	**RFLAGS** Extra flags to give to the Fortran compiler for Ratfor programs.
14	**LINTFLAGS** Extra flags to give to lint.

NOTE − You can cancel all variables used by implicit rules with the '-R' or '--no-builtin-variables' option.

You can also define macros at the command line as shown below −

```
make CPP = /home/courses/cop4530/spring02
```