Shebang! (first line of a script)	#!/bin/bash	#!/bin/tcsh
Multiple commands on the same line (semicolon)	command1; command2; command3	
Extending commands across multiple lines (backslash)	command1 argument | command2 | command3 | \ command 4 | command5 > file	
Variable assignment	VAR=”Here is a string”	set VAR=”Here is a string”
Setting environment variables	export VAR=”Here is a string” *No spaces around the = sign!*	setenv VAR “Here is a string” *No = when using setenv!*
Unsetting a variable	unset VAR	unset VAR

If statements
 	
Can use == != && || and others.
String sorting with < and >
	if [[$VAR1 == $VAR2]]; then
	  echo "True"
	else
	  echo "False"
	fi
		if ($VAR1 == $VAR2) then
		  echo "True"
		else
		  echo "False"
		endif

If statements with file property testing (see property table below)	if [[-d $VAR]]; then
	  echo "Directory!
	fi
		if (-d $VAR) then
		  echo "Directory!"
		endif

Passing arguments to a script Corresponding variables	myscript.sh arg1 arg2 arg3 … argN $1 $2 $3 … $N	
Assigning command output to variables (backtick)	VAR=`command1; command2; command3` (bash) Set VAR=”`command1; command2; command3`” (tcsh)	
String replacement	NEWVAR=${VAR/search/replace}	set NEWVAR= "$VAR:gas/search/replace/"

For loop on a list	for i in 1 2 3 4 5; do
	  echo $i
	done
		foreach i (1 2 3 4 5)
		  echo $i
		end

For loop using wildcards	for i in *.in; do
	  touch ${i/.in/.out}
	done
		foreach i (*.in)
		  touch "$i:gas/.in/.out/"
		end

For loop using commands	for i in `cat files`; do
	  grep "string" $i >> list
	done
		foreach i (`cat files`)
		  grep "string" $i >> list
		end

Test	**bash**	**tcsh**
Is a directory	-d	-d
If file exists	-a,-e	-e
Is a regular file (like .txt)	-f	-f
Readable	-r	-r
Writeable	-w	-w
Executable	-x	-x
Is owned by user	-O	-o
Is owned by group	-G	-g
Is a symbolic link	-h, -L	-l
If the string given is zero length	-z	-z
If the string is length is non-zero	-n	-s

Compilers	GCC	Intel	PGI
C	gcc	icc	pgcc
C++	g++	icpc	pgCC
Fortran77	g77	--	pgf77
Fortran90	gfortran	ifort	pgf90
Optimization	-O3	-fast	-fastsse

Compiler usage: **gcc source.c –o source.x**
gcc –c source.c
gcc source.o –o source.x

-o flag is for specifying the output name. If you don't give –o, the name of the output will be **a.out**

-c flag is for compiling to an object file (object.o), without linking (c is for compile). In order to use the object you need to compile again to link the file.

-g flag is for setting up debugging information in the software. In order to use that information, you need to use debugging software (like GDB or TotalView). Use printf/write statements for easy debugging.

./configure – Used to set up a and test the compiling environment for a software package.

./configure –prefix=<PATH> - used to specify the installation path for installing a software package, where <PATH> is the destination of make install

make – Used to compile a complicated software package with many source files. Must be used with a **Makefile**

make –f filename – specifies what makefile to use (defaults to **Makefile**)

make install – used after make to install the software package