Programming Languages: G221.2110
Hand Out 1

B. Mishra

1 Compilers and Interpreters.

1.1 Ada/Ed

1. Ada/Ed is available on ACF5, Sun workstations and for PC's.

2. Do "`man adaed`" on NYU computers for the man page.

3. Put the following statement in your "`.login`" file:

   ```
   setenv ADAED /usr/local
   ```

4. Put your ada program in a file that has the extension `.ada`, for example "`file1.ada`," "`file2.ada`," etc. The compiler will compile your files into a library whose name you must specify, e.g., "`testlib`."

5. To compile and execute your program type the following:

   ```
   %adacomp -n -l testlib file1    --use -n flag to creat a new library
   %adacomp -l testlib file2
   %adacomp -l testlib ...
   %adabind testlib                --binding a given binary library to
                                   --form an executable program
   %adaexec testlib                --run the ada program
   ```

   Note that you need not specify the `.ada` extension of your files. The name following -l will be the file where compiled program is placed. In your Ada program you must put the following statements:

```
with text_io;
procedure foo is
   use text_io;
   package int_io is new integer_io(integer);
   use int_io;

      ...                            --the rest of your program
```

where `foo` is your outermost (main) procedure.

1.2 Ada95

Instructions for Ada 95 will be made available at a later date.

1.3 C

1. Do "`man cc`" on your Unix machine to get the man page.
2. `cc` is the C compiler. It translates programs written in C programming language into executable load modules.

   `cc` accepts several types of filename arguments. For instance, files with names ending in `.c` are taken to be C source programs. They are compiled, and each resulting object program is placed in the current directory. The object file is named after its source file - the suffix `.o` replacing `.c` in the name of the object.
3. *Options*:
   (a) `-g` Produce additional symbol table information for `dbx(1)` and `dbxtool(1)`. When this option is given, the `-O` and `-R` options are suppressed.
   (b) `-o outputfile` Name the output file `outputfile`. `outputfile` must have the appropriate suffix for the type of file to be produced by the compilation. `outputfile` cannot be the same as source file.
   (c) `-O`[level] Optimize the object code. `-O` with the level omitted is equivalent to `-O2`. See the man page for more information.
   (d) `-p` Prepare the object code to collect data for profiling with `prof(1)`. Invokes a run-time recording mechanism that produces a `mon.out` file.
   (e) `-pg` Prepare the object code to collect data for profiling with `gprof(1)`. Invokes a run-time recording mechanism that produces a `gmon.out` file.
   (f) `-w` Do not print warnings.

4. *Files*:

```
a.out                executable output file
file.a               library of object files
file.c               C source file
file.i               C source file after preprocessing with cpp(1)
file.il              inline expansion file
file.o               object file
```

1.4 C++

(`CC` on NYU computers. On the PC's, you can use Borland, Microsoft compiler, which is available in the ACF PC lab. Just go to the lab and choose C++ from the main menu on the screen).

1. Do "`man CC`" on your Unix machine to get the man page.

2. Similar to `cc`, except that you must type in `CC` instead of `cc`.

1.5 Java

1. Save the source file with the name `"filename.java"`.

2. Compile using `javac` to get `"filename.class"`.

3. Run the `"filename"` with the `java` interpreter.

```
>javac filename.java

> java filename
```

4. *Using Java Applets*: First create `"appletname.class"` as before. Create an `HTML` page (in a file named `"appletname.html"`) as follows:

```
<HTML>
<HEAD>
<TITLE>applet title</TITLE>
</HEAD><BODY>
<P>Java applet goes here:
<APPLET CODE="appletname.class" WIDTH=150 HEIGHT=25>
```

```
</BODY>
</HTML>
```

5. You can view the applet using a Java-enhanced browser (e.g., `Netscape 2.0`).
6. If you don't have a Java-enhanced Web browser, use `appletviewer` to view the Java applet

```
>applet viewer appletname.html
```

1.6 LISP

1. The LISP environment is invoked by typing "**kcl**" at the Unix prompt. You should get the kcl prompt, ">".
2. Exit kcl with the function "**(bye)**".
3. You can use kcl as an interpreter; e.g.,

```
>(+ 2 3)
5

>
```

   To execute (interpret) LISP code stored in a file, enter

```
>(load "filename.lisp")
```

4. To compile and execute code that you have stored in a file, do

```
>(compile-file "filename.lisp")
  ...
>(load "filename.o")
```

1.7 Pascal

1. Do "`man -k Pascal`" or "`man pc`" on your Unix machine to get the man page.