PHYS-4007/5007: Computational Physics

Programming Tutorial
Help with Fortran and Python on the Brown Hall 264 Computers

1 Fortran Tutorial.

1.1 Code Creation.

Last week we created two directories in your login directory using

```
> mkdir fortran
> mkdir python
```

and we created a Fortran code in the fortran subdirectory using emacs as shown below:

```
123456789T123456...
      PROGRAM MYCODE
C
C This is my first Fortran code.
C
      INTEGER I, J, I_INDEX(100)
      REAL PI, LLCOEFF(100)
      REAL*8  DPI, SSCOEFF(100), BBOUT(100)
C
      PI = 3.14159
      DPI = 3.14159265359D0
C
C Calculate integer and single precision arrays.
C
      DO 52 I = 1, 100
         I_INDEX(I) = I*2
         LLCOEFF(I) = FLOAT(I**2) * SIN(PI*FLOAT(I)/10.)
  52  CONTINUE
C
C Calculate double precision arrays.  This DO loop has an error.
C
      DO 1005 J = 1, 100
         SSCOEFF(J) = COS(DPI*DFLOAT(J)/8.0D0)
         BBOUT(J) = SSCOEFF(J) * DEXP(-DFLOAT(J)/300.D0)
  52  CONTINUE
```

```
C
C Output this data to the screen in ordered columns.
C
C This next DO loop has an error in it.
C
      DO 2344 I = 1, 200
         WRITE(*, 2340) I, I_INDEX(I), LLCOEFF(I), SSCOEFF(I),
     1      BBOUT(I)
C
C This FORMAT statement has two errors in it,
C one minor and one severe.
C
 2340    FORMAT(2X,2I5,2X,F8.4,2X,1PE12.2)
 2344 CONTINUE
C
      STOP
      END
```

This code was to be saved on a USB stick at the end of last week's tutorial. If you are using a different computer than last week, make sure you copy this code file from your USB stick to the fortran subdirectory on the machine you are currently using.

1.2 Compiling the Code.

In the code above note that I purposely included a few programming errors, some you will see when compiling and others when you run the code. You will need to fix these in order for the code to run successfully (see the next subsection). Also note that one should be careful with variable types when carrying out calculations:

- One should only have integer-type for integer statement arithmetic,
- only floats (*i.e.*, REAL) for single-precision calculation statements,
- only doubles (*i.e.*, REAL*8 or DOUBLE PRECISION) for double-precision calculation statements,
- only characters in string statements, etc.

On the Tutorial web page linked to the course web page, there is a PDF document that lists some of the standard math, type conversion, and operational functions found in Fortran. Print this out and keep it as reference for when you are doing Fortran programming. Make sure you pay attention to these tables when doing your homework, exams, and optional project for this course. Information on working with character strings in Fortran can be found in Appendix B of the course notes.

At the Unix prompt from your fortran subdirectory, enter the following command:

```
> gfortran –o mycode.exe mycode.f
```

Note that you will see a variety of warning and error messages output to your terminal window.

1.3 Fixing the Compile Error in Our Sample Fortran Code

Edit your file with emacs and make the following correction to your code:

```
> emacs mycode.f &
```

You should have noted one error, while compiling the code:

- The 'DO 1005' loop has an error in the numbered CONTINUE statement that closes this DO loop. As can be seen, instead of '52   CONTINUE' we should have written '1005 CONTINUE' — this error occurred since I did a clip and paste from the previous DO loop and I forgot to change the label number associated with CONTINUE.

Now recompile your code with:

```
> gfortran –o mycode.exe mycode.f
```

At this point we should see no more errors.

1.4 Running this Fortran Code

Assuming we have fixed the compile error, the gfortran command above will make an executable file called mycode.exe. To run this code one simply types at the Unix prompt (remember this is labeled here as '>'):

```
> ./mycode.exe
```

Here the './' tells Unix that the executable file is in the current directory. This will result in an output of 200 lines of numbers in column format showing the results of the calculations in your code. Some of the output will look 'strange' due to the errors I purposely included in this code. We will address these operational errors now.

- In the 'DO 2344' loop, variable I steps from 1 (one) to 200, yet the variables that are being printed are only 100 element arrays. As such, there is no telling what would be output for the I = 101 to I = 200 elements in this output — it could be zeros, or it could be "jibberish." Change this DO loop ending number from 100 to 200.
- In the '2340 FORMAT' statement, our first 2 variables are integers and we allowed 5 spaces for each integer which is more than sufficient for the values stored in these variables. However, the first floating point number array, LLCOEFF, will contain values that will be too large for the 8 spaces we have allowed with the 4 spaces after the decimal point. This results in '*******' being printed instead of numbers for some of the values. We can fix this by changing the '8' to a '10' in the 'F' FORMAT code (*i.e.*, F10.4).
- There are two arrays that are being printed out in scientific format, SSCOEFF(I) and BBOUT(I), yet the 2340 FORMAT statement has only one format code listed for these variables (*i.e.*, 1PE12.2). As such, this will either cause the code to crash during a run or produce strange output for the subsequent numbers in the following lines of output. To fix this, we just need to change '1PE12.2' to '1P2E12.2' in the FORMAT statement.

Once these changes are made, recompile your code with the gfortran command as shown above, then rerun your code with

```
> ./mycode.exe
```

At this point, we should no longer have any errors associated with the code. Here is this code again printed with the corrections in place. One can access this corrected code on the Tutorial web page linked to the course web page by clicking on the second 'mycode.f' link.

```
123456789T123456...
      PROGRAM MYCODE
C
C This is my first Fortran code.
C
      INTEGER I, J, I_INDEX(100)
      REAL PI, LLCOEFF(100)
      REAL*8  DPI, SSCOEFF(100), BBOUT(100)
C
      PI = 3.14159
      DPI = 3.14159265359D0
C
C Calculate integer and single precision arrays.
C
      DO 52 I = 1, 100
         I_INDEX(I) = I*2
         LLCOEFF(I) = FLOAT(I**2) * SIN(PI*FLOAT(I)/10.)
```

```
  52  CONTINUE
C
C Calculate double precision arrays.  Fixed the label
C statement number for CONTINUE.
C
      DO 1005 J = 1, 100
         SSCOEFF(J) = COS(DPI*DFLOAT(J)/8.0D0)
         BBOUT(J) = SSCOEFF(J) * DEXP(-DFLOAT(J)/300.D0)
 1005 CONTINUE
C
C Output this data to the screen in ordered columns.
C
C Fixed error in maximum I, was 200, should be 100.
C
      DO 2344 I = 1, 100
         WRITE(*, 2340) I, I_INDEX(I), LLCOEFF(I), SSCOEFF(I),
     1      BBOUT(I)
C
C Fixed errors in FORMAT statement: F8.2 should be F10.2 and
C there should be a 2 in 1PE format code.
C
 2340    FORMAT(2X,2I5,2X,F10.4,2X,1P2E12.2)
 2344 CONTINUE
C
      STOP
      END
```

Feel free to play around with this code you made (by once again editing it in emacs). Try mistyping some of the lines in this code and see what compilation errors this may cause.

2 Python Tutorial.

There are many *free* online tutorials that teach programming in python. We will use one of these now. This site uses version 3 of python. On your terminal screen, leave the fortran subdirectory and go into the python subdirectory:

```
> cd ..
> cd python
```

Now start the python3 interpreter:

```
> python3
```

You will now see the python screen prompt: >>>.

Open up the Firefox web browser software (*i.e.*, the "blue-orange" icon on the left side of the screen), and open the following web site:

https://www.learnpython.org/

Note that on this web site, you can either run the coding tutorial on the web page by clicking the "RUN" icon, or you can type the coding in your python3 terminal if there is only one line of code. For multiple lines of code, I recommend that you create a file with those lines of code using the emacs editor. Go through as many tutorial steps as you like during this tutorial class. I encourage you to also visit this site on your own to teach yourself python.

Once you are through with this python tutorial, enter the following in your python terminal:

```
>>> exit()
```

The parentheses are required since python uses functions to carry out commands.