

IISEE lecture for group training

Fortran programming for beginner seismologists

Lesson 1

Lecturer

Tatsuhiko Hara

Why Fortran?

- Easy to learn
- Many aspects common among computer programming languages
- Many seismological software have been developed using Fortran.
- Programming is effective to improve your understanding (it is impossible to write a code without proper understanding).

Starting editor

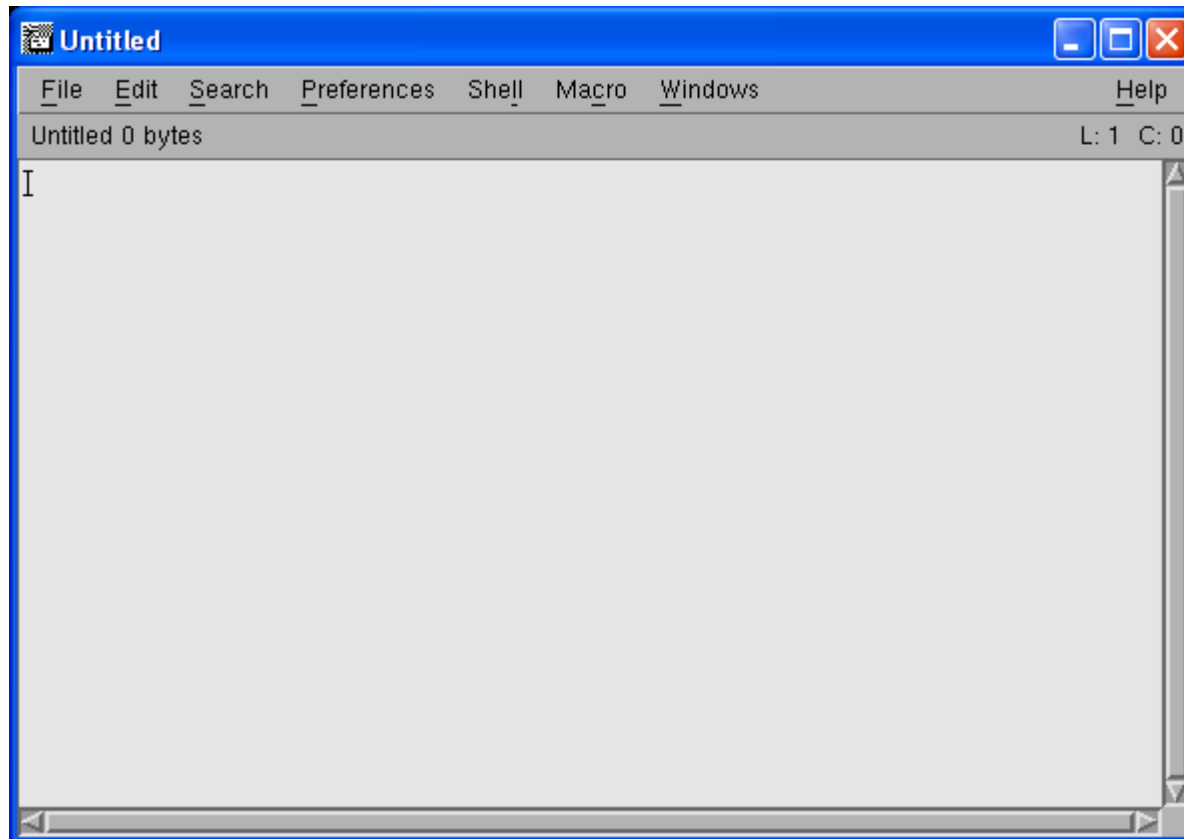
- We use “nedit” to edit programs.
- Type as:

```
$ nedit &
```

then you will get the window shown in the next slide.

Note: When you put “&” after the command, that command is executed as a background job.

nedit



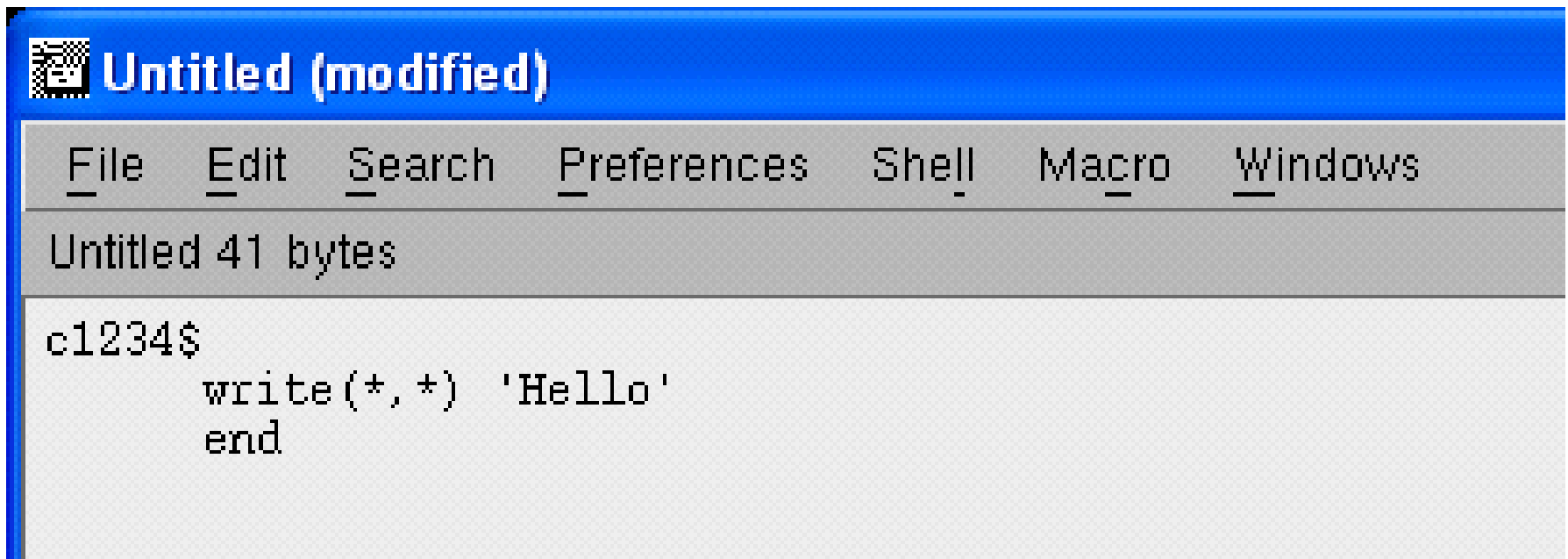
The first program

- Write the following in the nedit window

```
C1234$
```

```
    write(*,*) 'Hello'  
end
```

In the actual editor



The image shows a screenshot of a text editor window. The title bar is blue and contains a small icon on the left and the text "Untitled (modified)". Below the title bar is a menu bar with the following items: File, Edit, Search, Preferences, Shell, Macro, and Windows. Each item has a small underline character below it. Below the menu bar is a status bar that says "Untitled 41 bytes". The main editing area contains the following text:

```
c1234$  
    write(*,*) 'Hello'  
end
```

Save the file

- Save the program as `hello.f`:
 - Click “File” → “Save As” in the editor
 - Then write the name of the file
 - Click “OK”

EXERCISE 1-5

- a) Use a `ls` command in the xterm to confirm that the new file is created.
- b) Type as `cat hello.f` to check the content.

Compile and Execution

- Type the following command to compile the program:

```
$ g77 hello.f
```

- Use a *ls* command to confirm that “a.exe” is created.
- Then type as:

```
$ ./a.exe
```

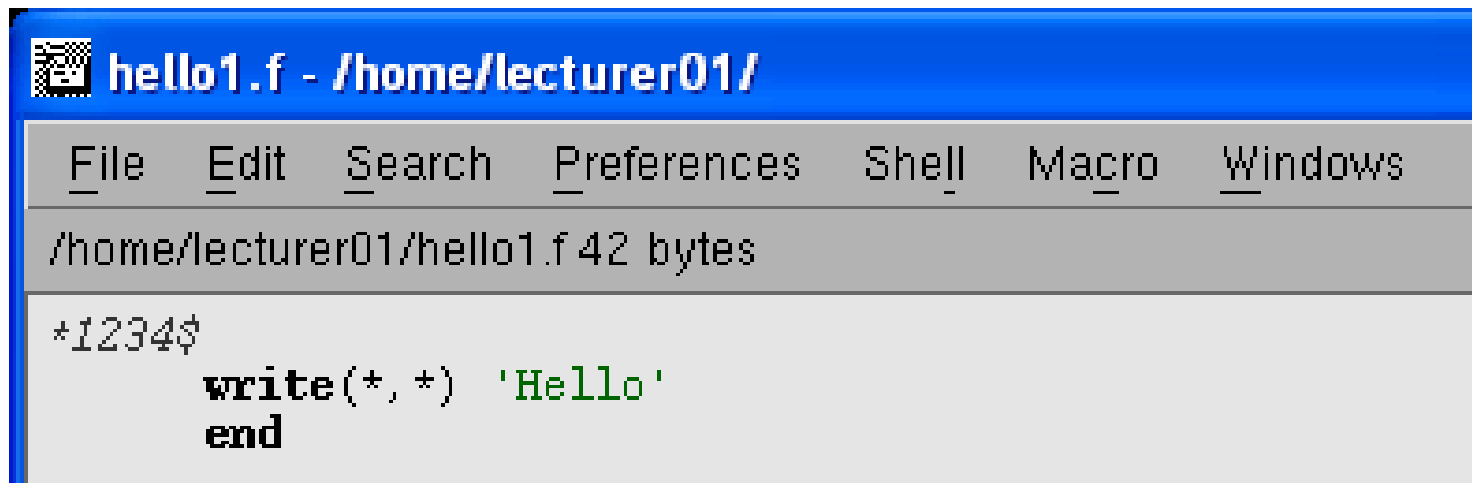
Note: The last command is specified by the relative path. “.” denotes the current directory.

Comment lines (1)

- When you put “c” or “*” at the first column, those lines are interpreted as comment lines.

EXERCISE 1-6

Replace “c” in the first column of the first line in “hello.f” by “*”, and save the modified program as “hello1.f.” Compile and execute this modified program.



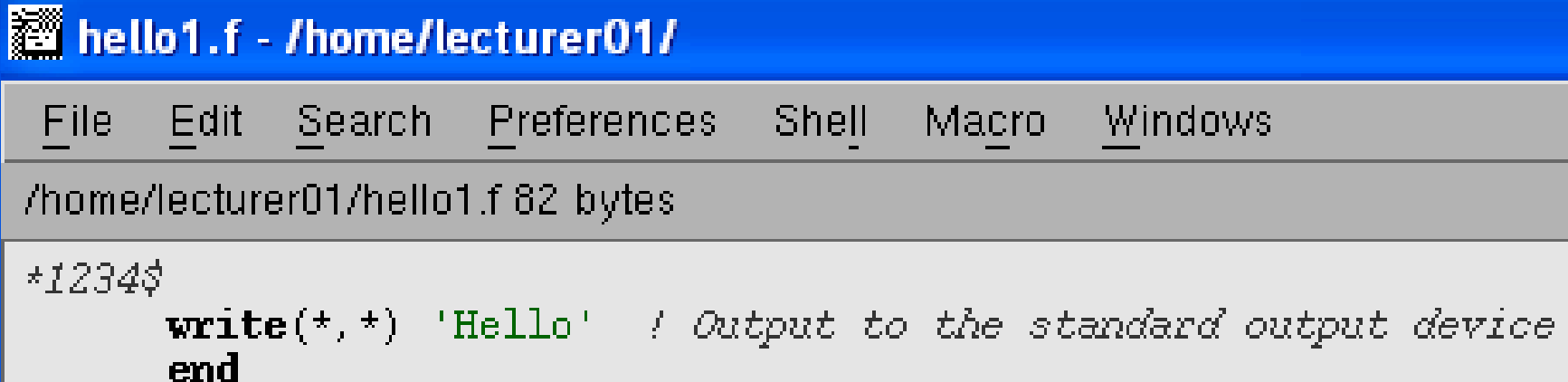
```
hello1.f - /home/lecturer01/  
File Edit Search Preferences Shell Macro Windows  
/home/lecturer01/hello1.f 42 bytes  
*1234$  
    write(*, *) 'Hello'  
end
```

Comment lines (2)

- When you put “!” in a program, the part after “!” is interpreted as comments. Although this is not the FORTRAN77 standard, it is supported by many compilers.

EXERCISE 1-7

Modify “hello1.f” as below. Then, compile and execute the modified program.



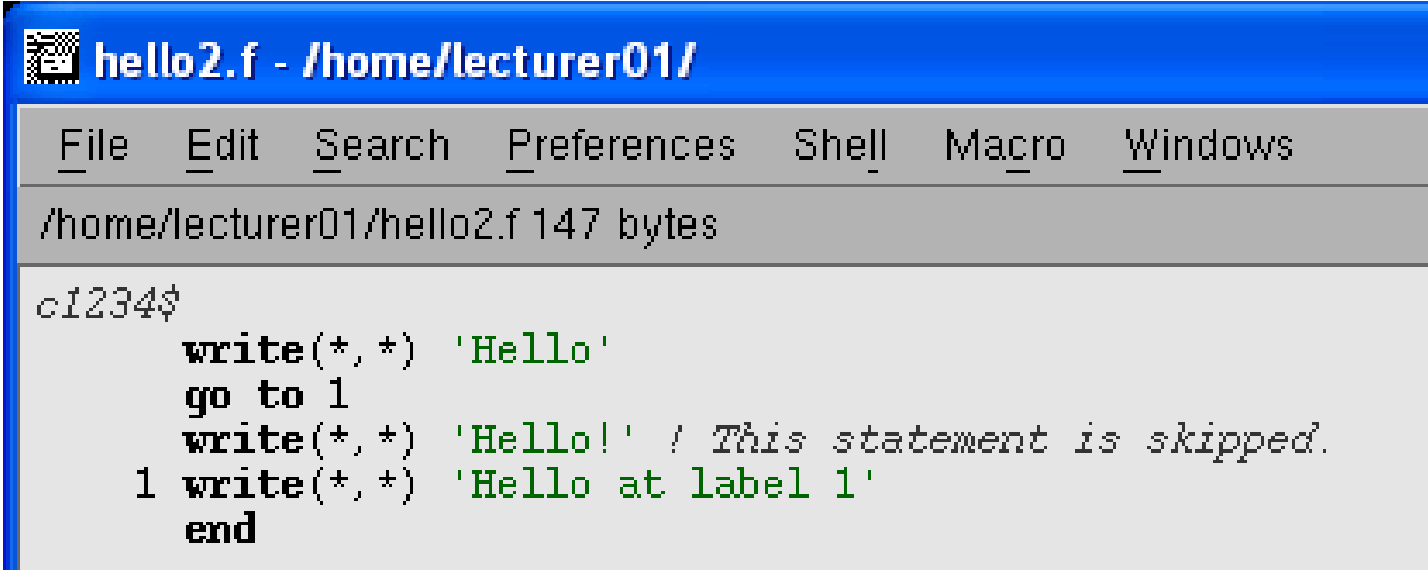
```
hello1.f - /home/lecturer01/  
File Edit Search Preferences Shell Macro Windows  
/home/lecturer01/hello1.f 82 bytes  
*1234$  
  write(*,*) 'Hello' ! Output to the standard output device  
  end
```

Statement labels

- Between the second and fifth columns, you can put statement labels.

EXERCISE 1-8

Modify “hello.f” as below and save it as “hello2.f”. Then, compile and execute the modified program.



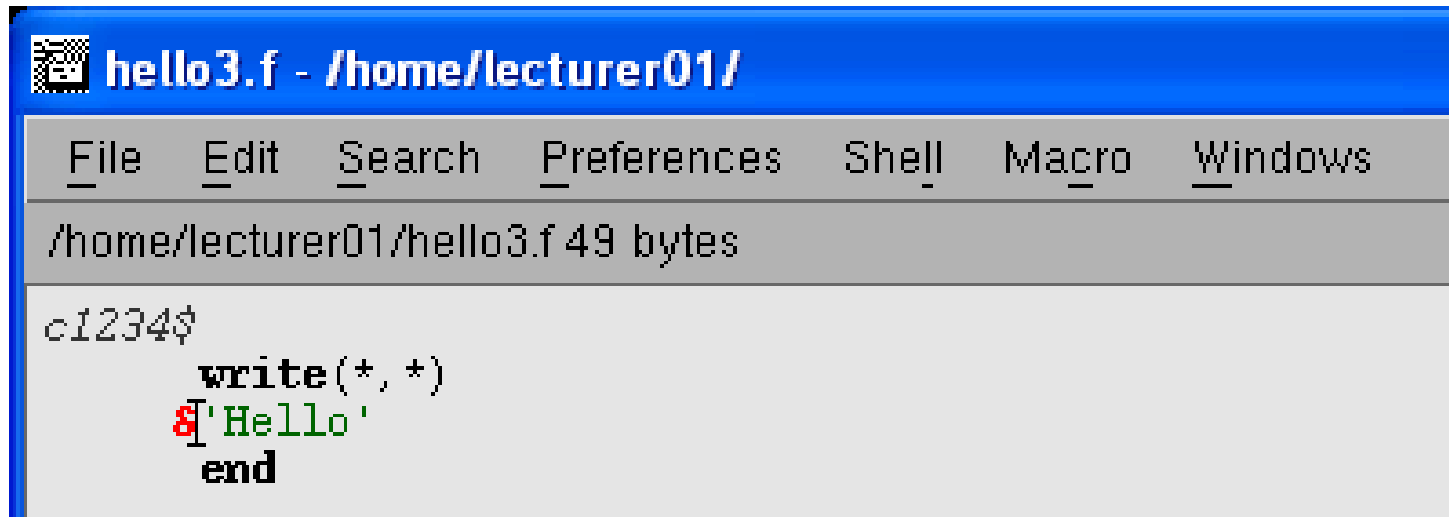
```
hello2.f - /home/lecturer01/  
File Edit Search Preferences Shell Macro Windows  
/home/lecturer01/hello2.f 147 bytes  
c1234$  
    write(*,*) 'Hello'  
    go to 1  
    write(*,*) 'Hello!' / This statement is skipped.  
1 write(*,*) 'Hello at label 1'  
end
```

Continuation of lines

- When you put a character such as “&” at the sixth column, that indicates the continuation from the previous line.

Exercise 1-9

Modify “hello.f” as below and save as “hello3.f”. Then, compile and execute the modified program.



```
hello3.f - /home/lecturer01/  
File Edit Search Preferences Shell Macro Windows  
/home/lecturer01/hello3.f 49 bytes  
c1234$  
    write(*,*)  
    &'Hello'  
    end
```

Where can we put statements?

- We can put FORTRAN statements within the 7th and 72nd columns.

EXERCISE 1-10

Try the programs shown below, and see what happens?

The image shows two screenshots of a FORTRAN editor window titled "hello4.f - /home/lecturer01/". The editor has a menu bar with "File", "Edit", "Search", "Preferences", "Shell", "Macro", "Windows", and "Help". The status bar shows the file path and line/column information: "/home/lecturer01/hello4.f 91 bytes L: 2 C: 73" for the top screenshot and "/home/lecturer01/hello4.f 90 bytes L: 2 C: 72" for the bottom screenshot. The code in both is:

```
c1234$  
  
end  
  
write(*,*) 'Hello'
```

In the top screenshot, the cursor is at column 73, and the text "Hello" is followed by a vertical bar. In the bottom screenshot, the cursor is at column 72, and the text "Hello" is followed by a vertical bar. A callout box with the text "Notice these!" has arrows pointing to the vertical bars in both screenshots, highlighting the column positions where the statements end.

WRITE statement

- The following statement

```
Write(*,*) list
```

prints out a list of items (`list`) to the screen (specified by the first “*”) in the standard format (specified by the second “*”).