Computers in Chemistry – Lecture III

Prof. Dr. Stephan Irle

Quantum Chemistry Group

Nagoya University

Get this lecture online

- Please go to: http://qc.chem.nagoya-u.ac.jp
- Click on "Teaching"
- Click on "PDF" link of "3.1 Lecture III – Introduction to computer programming"

```
1.2 Assignment 1 (PDF)
2.1 Lecture II - Use your computer, see chemistry (PDF)
2.2 Example outputs: h2o.out h2o-freq.out benzene.out
2.3 Molden 4.8 program for MacOSX: molden4.8.macosX
2.4 Assignment 2 (PDF)
2.5 Introduction to UNIX (Japanese)
3.1 Lecture III - Introduction to computer programming (PDF)
3.2 Assignment 3 (PDF)
3.3 Example programs: hello.c hello.f
```

Today's Lecture

- Overview of programming languages
- Scientific computer languages: Python, C and FORTRAN
- Example codes in these languages
- How to write a program on your Mac
- How to compile and run ("execute") a short program on your Mac

Overview I

- Microsoft Office: Word, Excel, Powerpoint, etc.

Microsoft Word   Microsoft Excel   Microsoft PowerPoint

 – limited programming capability

 – predefined functions only (except for Visual Basic, limited use)

- Calculator: Simple calculus possible

Overview II

- Computer only understands '0' and '1' (machine language)
- Two basic variations of programming languages:
  - Interpreter based
  - Compiler based
- Interpreter: program in machine language; takes a program and "runs" it. Needs to run independent from the program.
  - Example: Basic, python, Java
- Compiler: needs to be run once, produces machine-readable code (0010011 etc.). Once program has been "compiled", the compiler is no longer necessary.
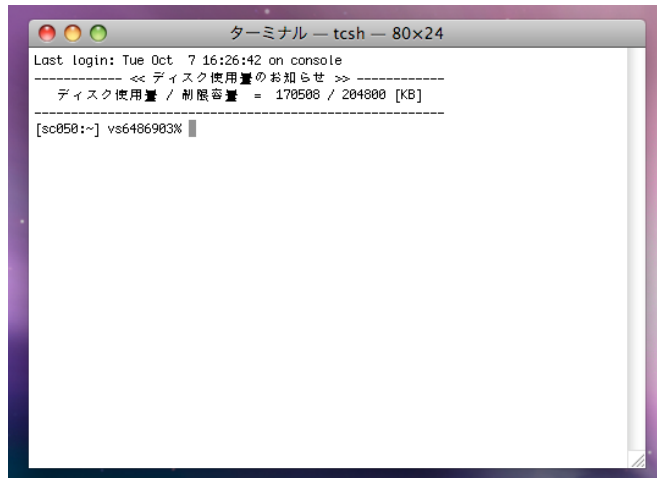 - Example: C, FORTRAN, COBOL, Java, etc …

Interpreter language: Python I

- Interpreter based languages: You can type the text, and the interpreter immediately executes your command.
- Example language: Python
- On your Mac, open X-windows terminal:

Interpreter language: Python II

- X-Windows automatically opens a so-called "Terminal"

Interpreter language: Python III

- In your terminal, type:

  python <enter>

- You will see:

Interpreter language: Python IV

- In your terminal, type:
  >>>3+10<enter>
- You will see:

13
   >>>x=2; y=3; x**2+y**2

13
   >>>s="Hello, world"
   >>>s
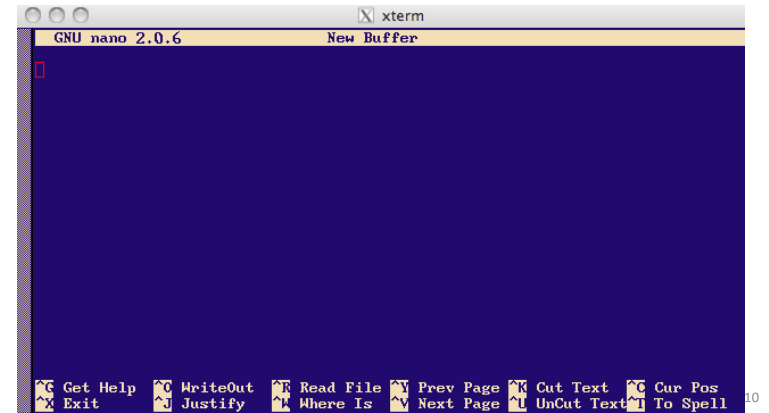   'Hello, world"
   >>>quit()

Now you are back at your terminal command line.

9

Compiler languages: Text editor I

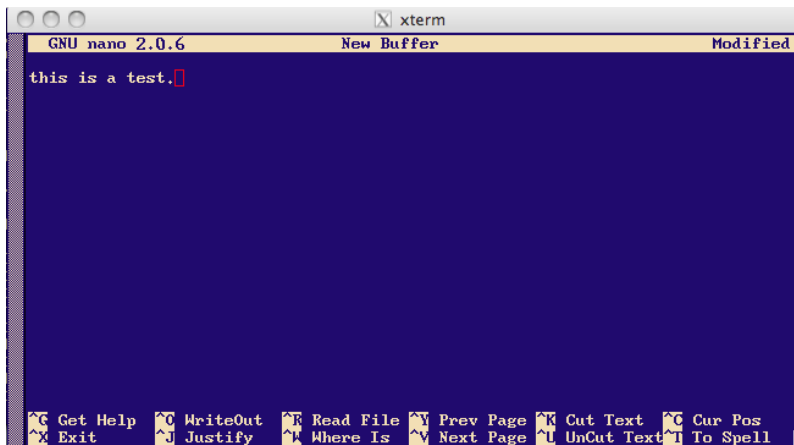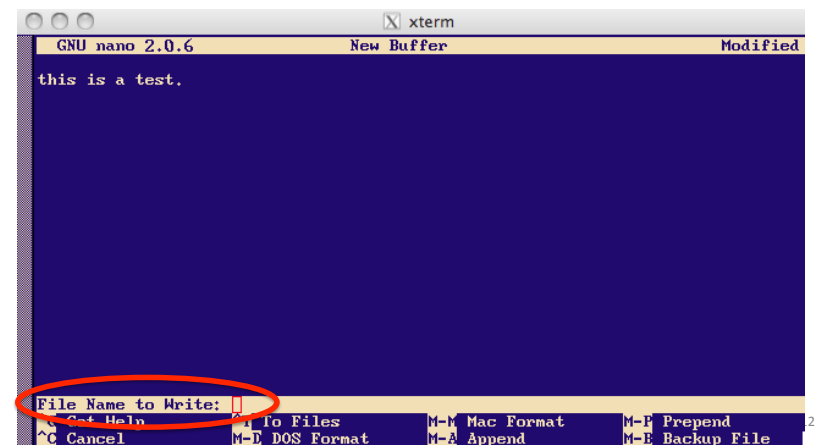- In your terminal, type:

  pico<Enter>

10

Compiler languages: Text editor II

- In pico, you can enter any text you like, for example:

Compiler languages: Text editor III
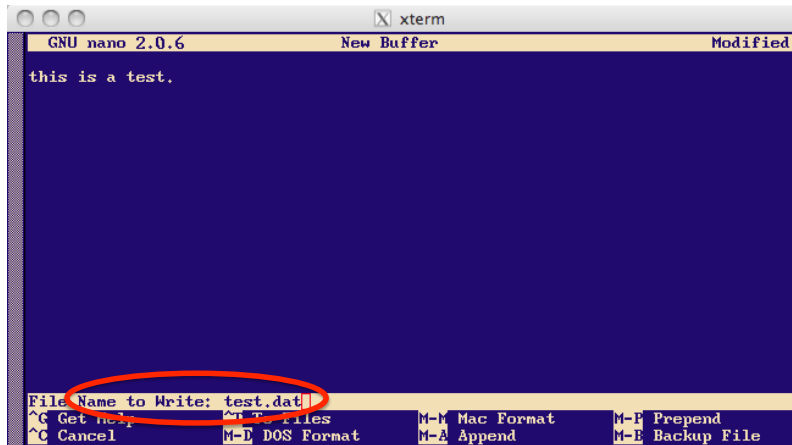
- To save the file, you type <Ctrl>O = ^O
- You are asked a file name:

Compiler languages: Text editor IV
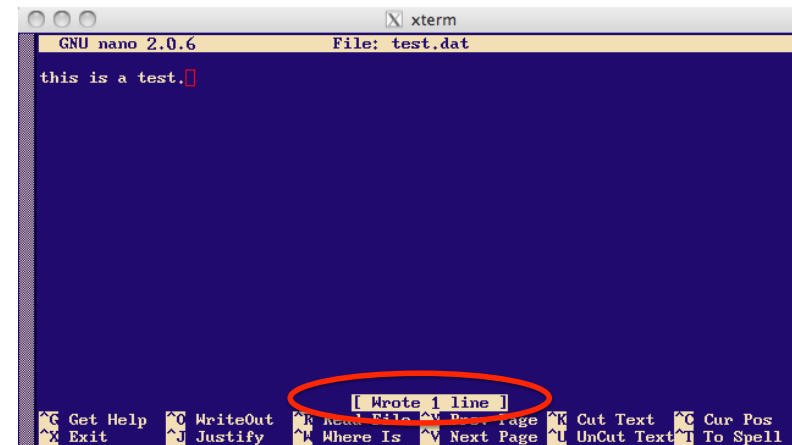
- Give a file name, for example test.dat<enter>

```
GNU nano 2.0.6            New Buffer                    Modified

this is a test.




File Name to Write: test.dat
^G Get Help          ^T To Files      M-M Mac Format    M-P Prepend
^C Cancel            M-D DOS Format   M-A Append        M-B Backup File
```

13

Compiler languages: Text editor V

- Result looks like this:

```
GNU nano 2.0.6            File: test.dat

this is a test.




                              [ Wrote 1 line ]
^G Get Help  ^O WriteOut       ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify        ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell
```

14

Compiler languages: Text editor V

- Exit text editor using ^X
- You are back to the command line, where you can type:
- ls<enter>
- You should see:
- test.dat

15

Example programs: C, FORTRAN

- Using pico, write a C language program "hello.c"

```
/* hello.c: display a message on the screen */
#include <stdio.h>
main()
{
    printf("hello, world\n");
}
```

Note: On Japanese keyboards, the "\" is under the ¥ key"

Comments
Opening statements
Program or subroutine name statements

- Compile program: cc hello.c
- Run program: ./a.out

16

Example programs: C, FORTRAN

- Writing "Hello world" in FORTRAN.

```
c
c   hello.f: display a message on the screen
c
    Program Hello
    implicit none

    write(*,10)
10  format('Hello, world.')
    END
```

10: Label!

Comments
Opening statements
Program or subroutine name
statements

- Compile program: gfortran hello.f
- Run program: ./a.out

C, FORTRAN program structure

- Labels do NOT exist in C
- Labels are sometimes used in BASIC and FORTRAN, especially with "GOTO" statements
- Example: infinity loop:

```
10  write (*,20)
20  format ('Hello, world')
    goto 10
    end
```