

Computational Chemistry & Classical Molecular Dynamics
Prof. B. L. Tembe
Department of Chemistry
Indian Institute of Technology - Bombay

Lecture – 12
Practical Session on Programming 2: The Exponential Function

Hello and welcome to today's session on the computational chemistry course. We had 1 practical session a few weeks back. Today, we will have the second practical session. Wherein we will execute a few programs and I urge you that as I am executing the programs here, you also put on your computer, get your Linux operating system on and execute the same programs because unless you practice, all this theoretical knowledge will not be very useful, okay. So now I am setting up this program.

(Refer Slide Time: 00:54)

```
bltembe@bltembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bltembe bltembe 2943 Nov 3 01:29 output1.dat
-rw-rw-r-- 1 bltembe bltembe 36 Oct 20 16:37 outputa.dat
-rw-rw-r-- 1 bltembe bltembe 1701 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bltembe bltembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bltembe bltembe 1184 Jan 11 17:36 progfs.f
-rw-rw-r-- 1 bltembe bltembe 167 Oct 20 16:37 ques1.f
-rw-rw-r-- 1 bltembe bltembe 261 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bltembe bltembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bltembe bltembe 394 Oct 29 16:37 ques.f
-rw-rw-r-- 1 bltembe bltembe 23 Oct 20 16:37 rast.f
-rw-rw-r-- 1 bltembe bltembe 586 Oct 29 16:37 three.f
-rw-rw-r-- 1 bltembe bltembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bltembe bltembe 97 Oct 29 16:37 uj.f
bltembe@bltembe-inspiron-3521:~/prog$ vi bexec
bltembe@bltembe-inspiron-3521:~/prog$ vi expted.f
bltembe@bltembe-inspiron-3521:~/prog$ gfortran expted.f
bltembe@bltembe-inspiron-3521:~/prog$ ./a.out
input value of x whose exp(x) will be calculated
3.
x. exp(x),dex(x)
3.000000000000 20.085536923187001 20.085536923187008
bltembe@bltembe-inspiron-3521:~/prog$
bltembe@bltembe-inspiron-3521:~/prog$
bltembe@bltembe-inspiron-3521:~/prog$
bltembe@bltembe-inspiron-3521:~/prog$ pwd
/home/bltembe/prog
bltembe@bltembe-inspiron-3521:~/prog$ ls
a.out      bl.f          expite.f      inputinv.dat  matrix.f     output1.da
admatrix.f cv.f          frac.f        inputpowr.dat multmatrix.f  outputa.da
ascend.f   dataint      input1.dat    inputreal     ndhl.txt     output.dat
ascending.f del.f         inputiflt.dat inputtw.dat   nn.f         outputd.da
atnu.f     datareal     input3.dat    klv.f         output.dat   progfs.f
bexec     descending.f lnputa.dat    matinv.f     output.dat   ques1.f
bexec     examples.desktop input.dat     matinv-ok.f  outputx.dat  ques2.f
bexec     expitd.f     inputint     matinvsub.f  outmatinvcorr.dat ques4.f
bltembe@bltembe-inspiron-3521:~/prog$ ls -l
```

So you will see on your screen, I am in my working directory. I am in my working directory. So how do I know which directory I am in. I just type in PWD. PWD tells me which is the working directory, where in I am working. So this is bltembe@bltembe inspiron 3521/prog. So I am in the program subdirectory. So I want to know what are all the programs in my subdirectory. So I type ls. Ls gives me a list of all the programs. So what I have here is a list of all the programs in this subdirectory. So this gives me a short list.

(Refer Slide Time: 01:39)

```

Terminal
bitembe@bitembe-inspiron-3521: ~/prog
-rw-rw-r-- 1 bitembe bitembe 320 Nov 3 03:12 inputiflt.dat
-rw-rw-r-- 1 bitembe bitembe 52 Nov 3 02:16 inputold.dat
-rw-rw-r-- 1 bitembe bitembe 56 Nov 3 02:39 input3.dat
-rw-rw-r-- 1 bitembe bitembe 11 Oct 29 16:37 inputa.dat
-rw-rw-r-- 1 bitembe bitembe 20 Oct 29 16:37 input.dat
-rw-rw-r-- 1 bitembe bitembe 143 Jan 11 15:52 inputint
-rw-rw-r-- 1 bitembe bitembe 249 Nov 3 02:34 inputnatinv.dat
-rw-rw-r-- 1 bitembe bitembe 28 Oct 29 16:37 inputpower.dat
-rw-rw-r-- 1 bitembe bitembe 293 Jan 11 15:25 inputreal
-rw-rw-r-- 1 bitembe bitembe 28 Oct 29 16:37 inputw.dat
-rw-rw-r-- 1 bitembe bitembe 120 Oct 29 16:37 kiv.f
-rw-rw-r-- 1 bitembe bitembe 141894 Oct 29 16:37 km.f
-rw-rw-r-- 1 bitembe bitembe 4199 Nov 3 16:40 matinv.f
-rw-rw-r-- 1 bitembe bitembe 3676 Nov 3 19:20 matinv-ok.f
-rw-rw-r-- 1 bitembe bitembe 4213 Nov 3 19:06 matinvsub.f
-rw-rw-r-- 1 bitembe bitembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bitembe bitembe 343 Oct 29 16:37 multimatix.f
-rw-rw-r-- 1 bitembe bitembe 27 Oct 29 16:37 nidht.txt
-rw-rw-r-- 1 bitembe bitembe 6336 Oct 29 16:37 nr.f
-rw-rw-r-- 1 bitembe bitembe 0 Oct 29 16:37 ouput.dat
-rw-rw-r-- 1 bitembe bitembe 1 Oct 29 16:37 ouputd.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 ouputa.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 ouputx.dat
-rw-rw-r-- 1 bitembe bitembe 4237 Nov 3 16:33 outnatinvcorr.dat
-rw-rw-r-- 1 bitembe bitembe 2943 Nov 3 04:29 output.dat
-rw-rw-r-- 1 bitembe bitembe 29 Oct 29 16:37 outputa.dat
-rw-rw-r-- 1 bitembe bitembe 1701 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bitembe bitembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bitembe bitembe 1334 Jan 11 17:32 progfs.f
-rw-rw-r-- 1 bitembe bitembe 157 Oct 29 16:37 ques1.f
-rw-rw-r-- 1 bitembe bitembe 261 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bitembe bitembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bitembe bitembe 144 Oct 29 16:37 ques.f
-rw-rw-r-- 1 bitembe bitembe 23 Oct 29 16:37 rest.f
-rw-rw-r-- 1 bitembe bitembe 506 Oct 29 16:37 three.f
-rw-rw-r-- 1 bitembe bitembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bitembe bitembe 97 Oct 29 16:37 uj.f
bitembe@bitembe-inspiron-3521: ~/prog

```

If I want a long list, I will do `ls -l`, so it will give me a long list which gives all the details of the files. For example, the file size. Where it was created, okay. File name and so on. And the extreme left, there are several characters you will see. You will see the first character will be dash, then `rw-rw-r--`. So these are all the permissions for your Linux file, okay. So this is not so crucial for us.

(Refer Slide Time: 02:12)

```

Terminal
bitembe@bitembe-inspiron-3521: ~/prog
ab.f          bt.f          expte.f       inputnatinv.dat  matrix.f       output1.dat
adefmatrix.f cost.f       fff.f         inputpower.dat  multimatix.f   output2.dat
a.out        cv.f         input1.dat     inputold.dat     nidht.txt       output.dat
ascend.f     datant       inputiflt.dat inputw.dat       nr.f            progfs.f
ascendf.f   datreal     input3.dat     kiv.f            ouput.dat       ques1.f
atnu.f      descending.f inputa.dat     matinv.f         ouputa.dat      ques2.f
bexec       examples.desktop input.dat      matinv-ok.f     ouputx.dat      ques4.f
bexecd      expdep.f    inputint      matinvsub.f     outnatinvcorr.dat ques.f
bitembe@bitembe-inspiron-3521: ~/prog
total 404
-rw-rw-r-- 1 bitembe bitembe 137 Oct 29 16:37 ab.f
-rw-rw-r-- 1 bitembe bitembe 372 Oct 29 16:37 adnatix.f
-rwxrwxr-x 1 bitembe bitembe 7967 Jan 11 17:52 a.out
-rw-rw-r-- 1 bitembe bitembe 54 Oct 29 16:37 ascend.f
-rw-rw-r-- 1 bitembe bitembe 487 Oct 29 16:37 ascendi.f
-rw-rw-r-- 1 bitembe bitembe 320 Jan 11 00:00 ascending.f
-rw-rw-r-- 1 bitembe bitembe 35 Oct 29 16:37 atnu.f
-rwxrwxr-x 1 bitembe bitembe 7967 Jan 11 15:57 bexec
-rwxrwxr-x 1 bitembe bitembe 7967 Jan 11 16:06 bexecd
-rw-rw-r-- 1 bitembe bitembe 147 Oct 29 16:37 bt.f
-rw-rw-r-- 1 bitembe bitembe 1341 Oct 30 01:23 cost.f
-rw-rw-r-- 1 bitembe bitembe 34 Oct 29 16:37 cv.f
-rw-rw-r-- 1 bitembe bitembe 143 Jan 11 15:53 datant
-rw-rw-r-- 1 bitembe bitembe 243 Jan 11 15:50 datreal
-rw-rw-r-- 1 bitembe bitembe 84 Jan 11 00:00 dd.f
-rw-rw-r-- 1 bitembe bitembe 536 Oct 29 16:37 descending.f
-rw-rw-r-- 1 bitembe bitembe 8980 Oct 29 16:37 examples.desktop
-rw-rw-r-- 1 bitembe bitembe 494 Jan 11 17:51 expdep.f
-rw-rw-r-- 1 bitembe bitembe 370 Jan 11 00:20 expte.f
-rw-rw-r-- 1 bitembe bitembe 191 Oct 29 16:37 fff.f
-rw-rw-r-- 1 bitembe bitembe 7104 Nov 3 18:30 input1.dat
-rw-rw-r-- 1 bitembe bitembe 320 Nov 3 03:12 inputiflt.dat
-rw-rw-r-- 1 bitembe bitembe 52 Nov 3 02:16 inputold.dat
-rw-rw-r-- 1 bitembe bitembe 56 Nov 3 02:39 input3.dat
-rw-rw-r-- 1 bitembe bitembe 11 Oct 29 16:37 inputa.dat
-rw-rw-r-- 1 bitembe bitembe 20 Oct 29 16:37 input.dat
-rw-rw-r-- 1 bitembe bitembe 143 Jan 11 15:52 inputint

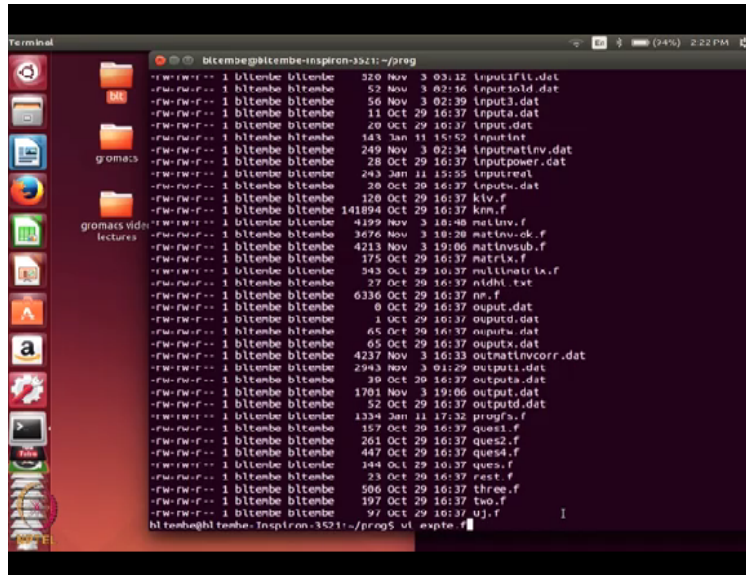
```

But you will just see that for some files, like this `a.out`. You see this file `a.out`. For that file, there is an `x` in the last column, `rw-rwxr-x`. So this `x` means it is an executable file. So `a.out` we have done in the past. It is an executable file. There are 2 more executable files. So whenever you compile a Fortran program, the compiled program will be an executable file. So `a.out` is 1

example. So we will now consider other examples as well today.

So if you remember last time we were executing a program to calculate the exponential of a function. The number we got was not quite what was obtained from the Fortran functions.

(Refer Slide Time: 02:58)

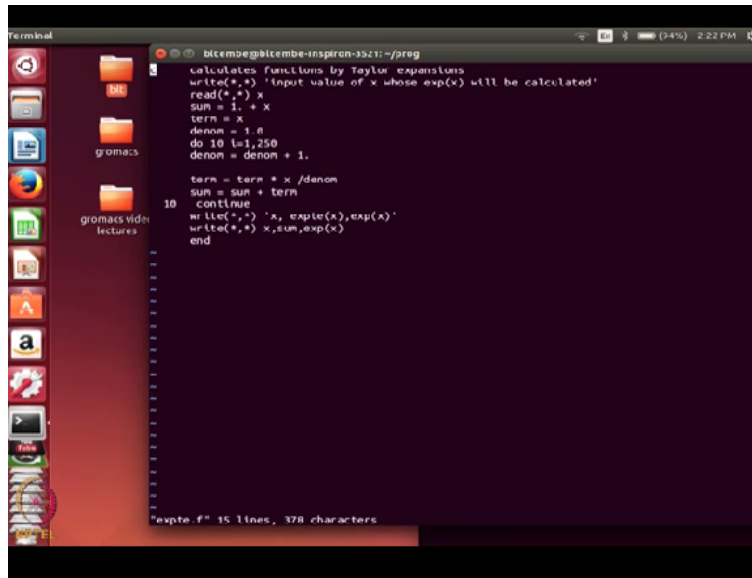


```
Terminal
bitempeg@bitembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bitembe bitembe 320 Nov 3 03:42 input1fit.dat
-rw-rw-r-- 1 bitembe bitembe 52 Nov 3 02:16 inputold.dat
-rw-rw-r-- 1 bitembe bitembe 56 Nov 3 02:39 input3.dat
-rw-rw-r-- 1 bitembe bitembe 11 Oct 29 16:37 inputa.dat
-rw-rw-r-- 1 bitembe bitembe 20 Oct 29 16:37 input.dat
-rw-rw-r-- 1 bitembe bitembe 143 Jan 11 15:52 inputint
-rw-rw-r-- 1 bitembe bitembe 249 Nov 3 02:34 inputnativ.dat
-rw-rw-r-- 1 bitembe bitembe 28 Oct 29 16:37 inputpower.dat
-rw-rw-r-- 1 bitembe bitembe 243 Jan 11 15:55 inputres1
-rw-rw-r-- 1 bitembe bitembe 20 Oct 29 16:37 inputv.dat
-rw-rw-r-- 1 bitembe bitembe 120 Oct 29 16:37 kiv.f
-rw-rw-r-- 1 bitembe bitembe 141894 Oct 29 16:37 km.f
-rw-rw-r-- 1 bitembe bitembe 4999 Nov 3 18:40 matinv.f
-rw-rw-r-- 1 bitembe bitembe 3676 Nov 3 18:28 matinv.ok.f
-rw-rw-r-- 1 bitembe bitembe 4213 Nov 3 19:06 matinvsub.f
-rw-rw-r-- 1 bitembe bitembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bitembe bitembe 343 Oct 29 16:37 multmatrix.f
-rw-rw-r-- 1 bitembe bitembe 27 Oct 29 16:37 nldht.txt
-rw-rw-r-- 1 bitembe bitembe 6336 Oct 29 16:37 np.f
-rw-rw-r-- 1 bitembe bitembe 0 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bitembe bitembe 1 Oct 29 16:37 output1.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 output2.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 output3.dat
-rw-rw-r-- 1 bitembe bitembe 4237 Nov 3 16:33 outnativcorr.dat
-rw-rw-r-- 1 bitembe bitembe 2943 Nov 3 04:29 output1.dat
-rw-rw-r-- 1 bitembe bitembe 30 Oct 29 16:37 outputa.dat
-rw-rw-r-- 1 bitembe bitembe 1701 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bitembe bitembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bitembe bitembe 1334 Jan 11 17:32 profg3.f
-rw-rw-r-- 1 bitembe bitembe 157 Oct 29 16:37 ques1.f
-rw-rw-r-- 1 bitembe bitembe 261 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bitembe bitembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bitembe bitembe 348 Oct 29 16:37 ques5.f
-rw-rw-r-- 1 bitembe bitembe 23 Oct 29 16:37 rest.f
-rw-rw-r-- 1 bitembe bitembe 506 Oct 29 16:37 three.f
-rw-rw-r-- 1 bitembe bitembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bitembe bitembe 97 Oct 29 16:37 u1.f
bitempeg@bitembe-inspiron-3521:~/prog$ ul -expte
```

So now let us execute that program again. So I have called that program, expte, vi expte.f. The name of the program is expte. Why I have called it expte? Because it is a program to calculate the exponential of a function and how do I calculate it? I calculate it using a Taylor expansion. So it is always good to give names of files which will remind you of what the things are going to be done in that program because otherwise there will be so many programs, so many names.

So suppose you call it program1, program2, program3; you will have 100 programs. All names are alike. You will not know the difference between 1 and the other. Whereas if the name itself tells you something like this name is exp, exp may tell you that it is something to do with an exponential. And te is a Taylor expansion. So this program is to calculate the exponential of a function using a Taylor expansion. So vi is an editor.

(Refer Slide Time: 04:04)



```
Terminal
bitembeg@bitembe-inspiren-1521: ~/prog
calculates functions by Taylor expansions
write(*,*) 'input value of x whose exp(x) will be calculated'
read(*,*) x
sum = 1. + x
term = x
denom = 1. #
do 10 l=1,250
denom = denom + 1.

term = term * x /denom
sum = sum + term
continue
10
write(*,*) 'x, exp(x),exp(x)'
write(*,*) x,sum,exp(x)
end

"expte.f" 15 lines, 378 characters
```

So when I type this vi and hit my enter button, I see the entire program on my screen. So let us recapitulate what all the statements in this program are. So the first line is a comment card. It says that it calculates functions by Taylor expansion. So that is my goal of this program. Calculate the function using a Taylor expansion. So now the next line is an executable line, right *,* input value of x whose exponential will be calculated.

So the moment this line is executed, on your screen will appear input the value of x whose exponential will be calculated. So now you know that you will have to input something. So then the next line is read *,*x. What is the meaning of this read *,*? That means it will read from the screen the first * refers to the screen, the second * refers to the format. Format we have discussed last time.

So usually it is best to give an unformatted statement so that you do not have to bother. * means it will read in whichever format you will input the line. So the next line says it will read this value of x. So then the next one, now you want to calculate the exponential of that x using a Taylor expansion. What is the Taylor expansion of the exponential? It is $1+x$, $1+x+x^2/2$ factorial + $x^3/3$ factorial and so on and so on.

It is an infinite series. So each term, each of the new term will be 1 power higher than the preceding term and the denominator will be 1 factorial higher than the previous term. So each

term is x to the n/n factorial. So how do I go about this? I will say that $sum=1+x$. Now why do I say $sum=1+x$? I know that the first term is x and sorry, the first term is 1, the second term is x , okay.

Then I have called $term=x$, okay. So now $denominator=1$. So this is my initial set of definitions. Now I started do loop. Do 10 i going from 1 to 250. So what this means. I have a do loop which extends from the present line of the do loop up to this 10 continue. So this is my entire do loop and this goes from $i=1$ to $i=250$. That means it will calculate 250 terms, okay. Now the next question is, how do I know 250 is enough?

So that is something which we will consider later. If 250 is not enough, you will go to 300. So that option you have. But I have given 250 because I know that it is a large enough number so which will be expected to give me reasonable number as far as the function is concerned. So do 10 i going from 1 to 250. Next line is $denominator$ is $denominator+1$. So you see that the earlier $denominator$ was 1.

So inside the loop, the $denominator$ will be 2 now. Because I have incremented it by 1 and the next, now I want to calculate the next term. The first 2 terms are already known, 1 and x . The next term will be $term*x/denominator$, okay. So this term was already x earlier. Now this new value of $term=x*x$, that is x square, $/denominator$. Now this $denom$ is, the earlier value was 1. Now it is $1+1$.

So the new value of the term will be x square/2. You know that 2 factorial is a same as 2. So I have my new value for the term. So now what is my new value of the sum? $Sum=sum + term$. The first value of sum was $1+x$. The new value of sum is the old value of sum, which is $1+x+x$ square/2 factorial. Now 10 continue. 10 continue means I will go back to this line which is the do statement.

So remember that in Fortran whenever you have an = sign, what that = sign does is to calculate whatever is on the right side and replace the value of term with the value what is calculated on the right side. So second time when I execute this, now i will be 2. When i will be 2, this

denominator is $\text{denom}+1$. The earlier value of denom was 2. Now this denom is 3 because $2+1$ is 3.

So the next value of the term will be old value of the term which was already $x^2/2$ and it multiplies by x . Now the numerator becomes x^3 and you are dividing by 3. So already there was 2 in the denominator. Now the denominator is multiplied by 3. So the new value of the term will be $x^3/3$ factorial. Now what do I do the sum? The sum will be again incremented, old value of the sum + this new value.

So what will be the sum at the second stage? It will be $1+x+x^2/2$ factorial + $x^3/3$ factorial. Then I continue again. It goes back. Now i will be 3. When i is 3, denominator becomes 4. The new term becomes, there is already x^3 earlier. x^3*x is x^4 . I divide by 4. So it is $x^4/4$ factorial. So fourth term is added here. $x^4/4$ factorial. I go on and on, until I do up to 250, $i=250$.

When $i=250$, I will be dividing by 251 factorial. So as the number of terms increases, each value of the term becomes smaller and smaller. So the last term will be so small that you will not change the sum anymore. So I calculate all this. Now what I want to do? I want to write on the screen whatever I have done. So what my program has done? My program has calculated the value of sum.

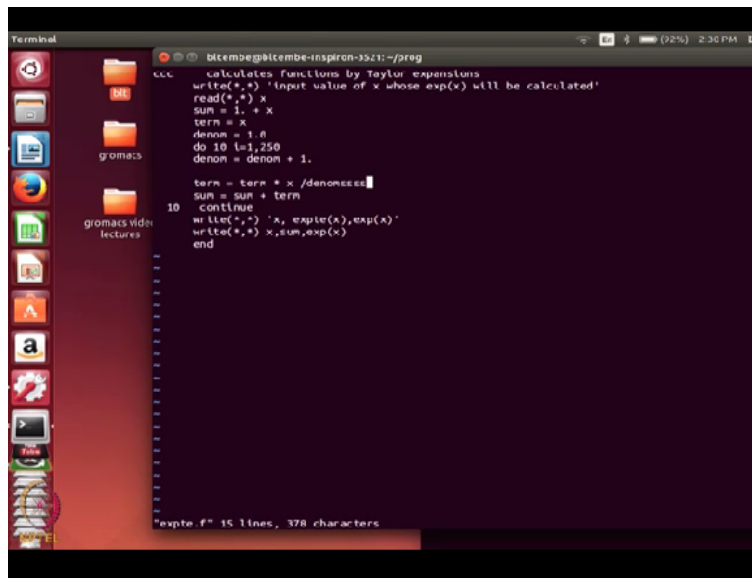
So on the screen I will write, write `*,*` the first quotation and the last quotation, these are 2 quotations which are such that whatever is written in between those 2 quotation marks, it will be printed on the screen. So this line will write to you `xtex`, what is the meaning of `xtex`? It is the exponential calculated using Taylor expansion. It will write `xtex` then `expx`. What is this `expx`? This `expx` is the exponential calculated by your Fortran function.

Remember your Fortran compiler has functions like `cos`, `sine`, all these are already built into the Fortran compiler. So what I want to do with this program is to compare the value of the exponential calculated by Taylor expansion that is the sum and the value which is calculated by the Fortran compiler. So both these will be written on the screen and then the program ends. So

since I have done vi here, this is my edit mode.

If I want to insert something, I have to type i. So if I type i, it allows me to insert things. So I have typed i now.

(Refer Slide Time: 11:25)

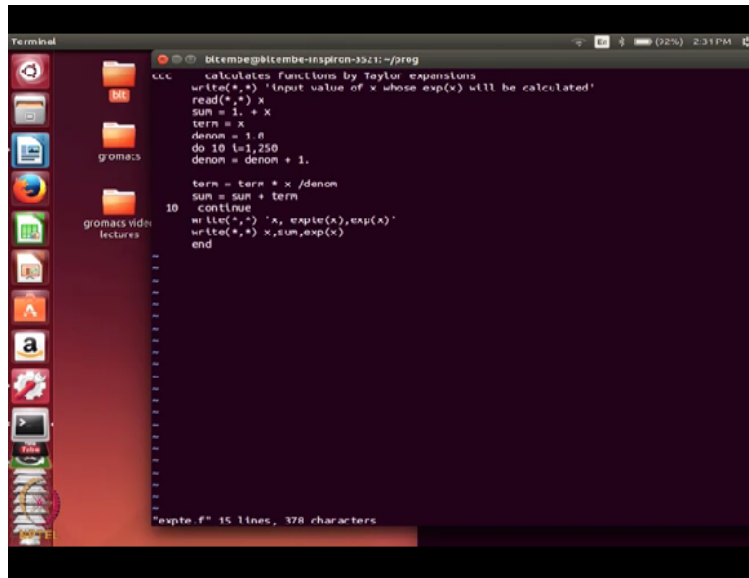


```
Terminal
bitembeg@bitembe-inspiron-3521:~/prog
ccc calculates functions by Taylor expansions
write(*,*) 'input value of x whose exp(x) will be calculated'
read(*,*) x
sum = 1. + x
term = x
denom = 1.0
do 10 i=1,250
denom = denom + 1.
term = term * x /denom
sum = sum + term
continue
write(*,*) 'x, exp(x),exp(x)'
write(*,*) x,sum_exp(x)
end
"expte.f" 15 lines, 378 characters
```

See whenever I type i, it inserts. So whenever I want to come out of the insert mode, I type escape. Escape allows me to come out of the insert mode. Once I have an escape, I can use the arrows to go wherever I want in the program. So remember whenever I am using a vi editor, there are 2 modes, one is an insert mode and one is an edit mode. Edit mode you go from insert mode, type escape, you will go to edit mode.

Now if I want to type something, I am at the character m. Instead of typing i, I can type a. When I type a, my cursor goes to 1 space after m and I can insert whatever I want. See I have typed a, I have written all this.

(Refer Slide Time: 12:15)



```
Terminal
bcc @ bicembe@bicembe-inspiron-3521: ~/prog
ccc calculates functions by Taylor expansions
write(*,*) 'input value of x whose exp(x) will be calculated'
read(*,*) x
sum = 1. + x
term = x
denom = 1.0
do 10 i=1,250
denom = denom + 1.

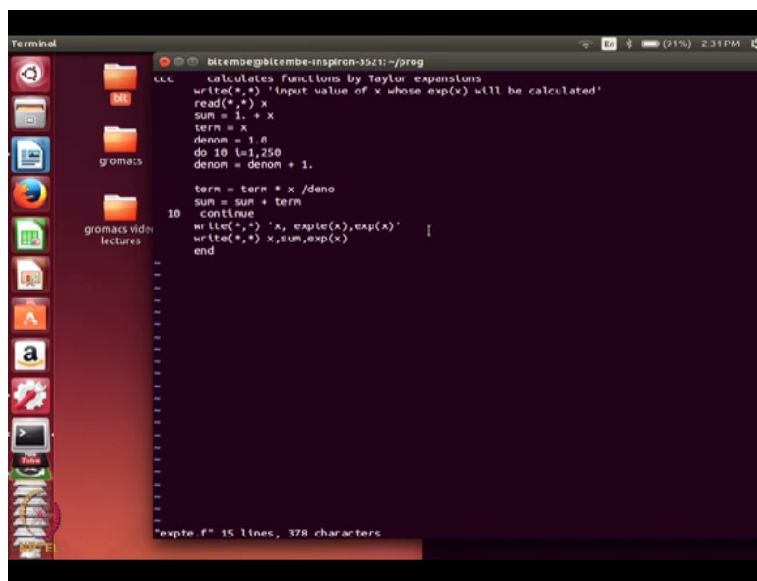
term = term * x /denom
sum = sum + term
continue
write(*,*) 'x, exp(x),exp(x)'
write(*,*) x,sum,exp(x)
end

"expte.f" 15 lines, 378 characters
```

Now if I want to remove all those extra things, I type escape again, then type 1x, that 1s is gone. Next x, the second s is gone. Type once more x, x means it is deleting that character. Delete once more, so all the wrong things that I have written are gone. So there are 2 ways of inserting, one is just typing i. When you type i, it will insert wherever your character is. Now suppose I type i now and I start inserting. So it has inserted to the left side of m.

Now again I want to delete them. Type escape again, then again x, x, x, x. Now I have just deno remaining.

(Refer Slide Time: 12:59)



```
Terminal
bcc @ bicembe@bicembe-inspiron-3521: ~/prog
ccc calculates functions by Taylor expansions
write(*,*) 'input value of x whose exp(x) will be calculated'
read(*,*) x
sum = 1. + x
term = x
denom = 1.0
do 10 i=1,250
denom = denom + 1.

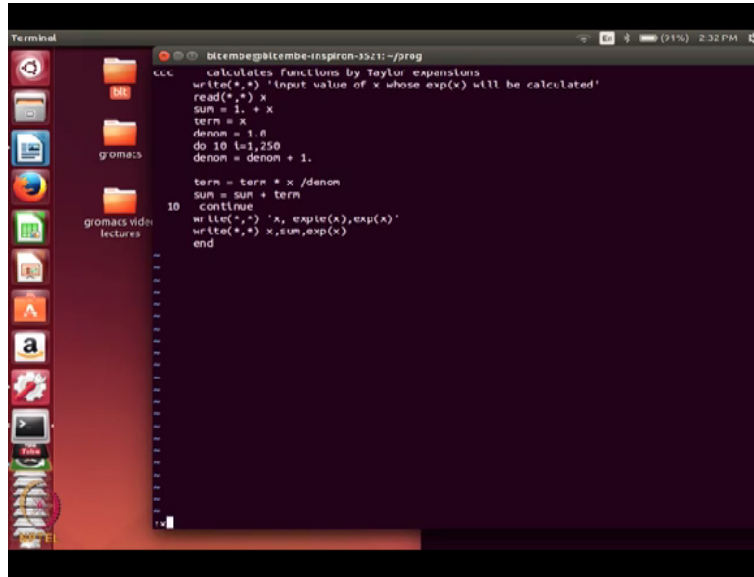
term = term * x /deno
sum = sum + term
continue
write(*,*) 'x, exp(x),exp(x)'
write(*,*) x,sum,exp(x)
end

"expte.f" 15 lines, 378 characters
```

Now I want to insert m after o. So in that case I will type a. When I type a, I go to the right of o

and I type m. I type escape again. So this is how you will use the editor to add or subtract things on the screen. So now my program is okay. So I want to come out of this editor. To come out of this editor, first you are in the edit mode, you shift :

(Refer Slide Time: 13:22)

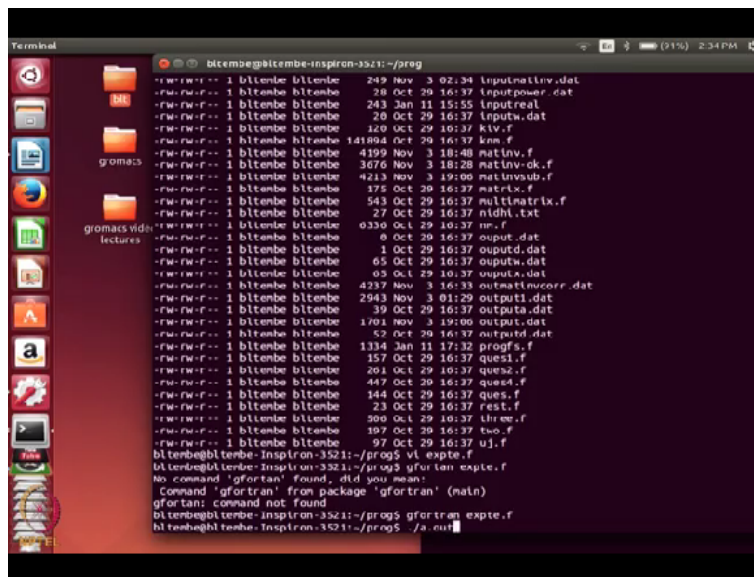


```
Terminal
bltenbe@bltenbe-inspiron-3521:~/prog
ccc calculates functions by Taylor expansions
write(1,*) 'input value of x whose exp(x) will be calculated'
read(1,*) x
sum = 1. + x
term = x
denom = 1. * x
do 10 i=1,250
denom = denom + 1.

term = term + x /denom
sum = sum + term
10 continue
write(1,*) 'x, exp(x),exp(x)'
write(1,*) x,sum,exp(x)
end
```

When I type shift : simultaneously, you see this colon coming at the bottom, so I am now ready to enter. So when I type x here, that means I can exit from my program. So I have typed x there, then enter. So what it has done, I went into this program xte.f, did whatever corrections I wanted, then came out. Now I am in a position to execute my program. To execute, what I have to do? I have to compile this. So earlier what did we do?

(Refer Slide Time: 13:58)



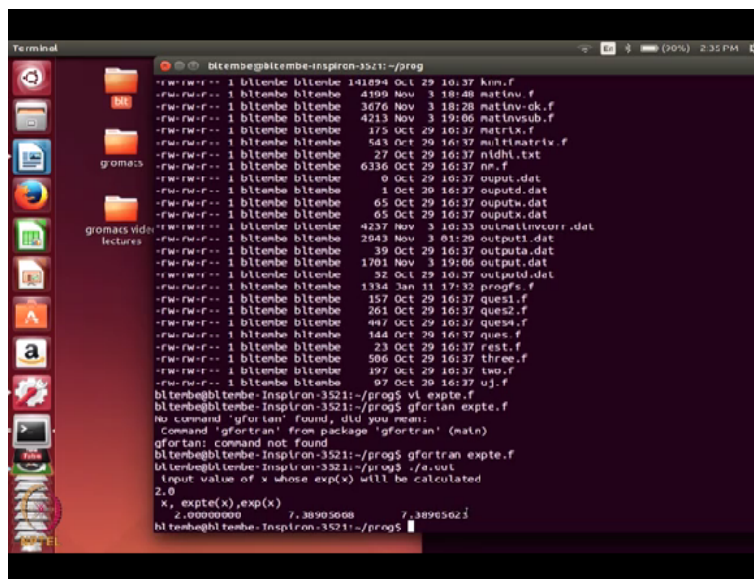
```
Terminal
bltenbe@bltenbe-inspiron-3521:~/prog
-rw-rw-r-- 1 bltenbe bltenbe 249 Nov 3 02:34 inputmatinv.dat
-rw-rw-r-- 1 bltenbe bltenbe 28 Oct 29 16:37 inputpower.dat
-rw-rw-r-- 1 bltenbe bltenbe 243 Jan 11 15:55 inputreal
-rw-rw-r-- 1 bltenbe bltenbe 20 Oct 29 16:37 inputw.dat
-rw-rw-r-- 1 bltenbe bltenbe 120 Oct 29 16:37 klv.f
-rw-rw-r-- 1 bltenbe bltenbe 141884 Oct 29 16:37 km.f
-rw-rw-r-- 1 bltenbe bltenbe 4199 Nov 3 18:48 matinv.f
-rw-rw-r-- 1 bltenbe bltenbe 3676 Nov 3 18:28 matinv-ok.f
-rw-rw-r-- 1 bltenbe bltenbe 4213 Nov 3 19:00 matinvsub.f
-rw-rw-r-- 1 bltenbe bltenbe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bltenbe bltenbe 543 Oct 29 16:37 multmatrix.f
-rw-rw-r-- 1 bltenbe bltenbe 27 Oct 29 16:37 nidhi.txt
-rw-rw-r-- 1 bltenbe bltenbe 0330 Oct 29 16:37 nr.f
-rw-rw-r-- 1 bltenbe bltenbe 0 Oct 29 16:37 ouput.dat
-rw-rw-r-- 1 bltenbe bltenbe 1 Oct 29 16:37 ouputd.dat
-rw-rw-r-- 1 bltenbe bltenbe 65 Oct 29 16:37 ouputw.dat
-rw-rw-r-- 1 bltenbe bltenbe 65 Oct 29 16:37 ouputx.dat
-rw-rw-r-- 1 bltenbe bltenbe 4237 Nov 3 16:33 outmatincorr.dat
-rw-rw-r-- 1 bltenbe bltenbe 2943 Nov 3 01:29 output1.dat
-rw-rw-r-- 1 bltenbe bltenbe 39 Oct 29 16:37 outputa.dat
-rw-rw-r-- 1 bltenbe bltenbe 1701 Nov 3 19:00 output.dat
-rw-rw-r-- 1 bltenbe bltenbe 52 Oct 29 16:37 outputf.dat
-rw-rw-r-- 1 bltenbe bltenbe 1334 Jan 11 17:32 prog5.f
-rw-rw-r-- 1 bltenbe bltenbe 157 Oct 29 16:37 ques1.f
-rw-rw-r-- 1 bltenbe bltenbe 201 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bltenbe bltenbe 487 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bltenbe bltenbe 144 Oct 29 16:37 ques.f
-rw-rw-r-- 1 bltenbe bltenbe 23 Oct 29 16:37 rest.f
-rw-rw-r-- 1 bltenbe bltenbe 300 Oct 29 16:37 tlrme.f
-rw-rw-r-- 1 bltenbe bltenbe 197 Oct 29 16:37 too.f
-rw-rw-r-- 1 bltenbe bltenbe 97 Oct 29 16:37 uj.f
bltenbe@bltenbe-inspiron-3521:~/prog$ vi expte.f
bltenbe@bltenbe-inspiron-3521:~/prog$ gfortran expte.f
No command 'gfortran' found, did you mean:
Command 'gfortran' from package 'gfortran' (main)
gfortran: command not found
bltenbe@bltenbe-inspiron-3521:~/prog$ gfortran expte.f
bltenbe@bltenbe-inspiron-3521:~/prog$ ./a.out
```

To compile, we wrote gfortran, so I have made a mistake here. So gfortr, gfortran, xte.f, expte.f. So it will come, and I hit enter. So when I say enter, it says gfortran found. Now it is giving some error, okay. gfortran command not found. Now can you identify what was the mistake? So instead of gfortran, I wrote gfortan. So it does not recognize this line. So I have to write I have to do it again gfortran exte.f.

So this time it has compiled. So I hope you have also put your computers on. Type all these things and you practice as I execute things here. Because that is the best way so that if there are any errors, you will be able to detect. And remember that in this program even a single comma here and there, single character in the wrong place, it will not compile properly. So you should always remember that everything should be in the right place.

So I have already compiled it. Now I execute it by saying ./a.out. Any execution in our programs is done by typing ./a.out. What does this do now? I have already compiled using gfortran, the compiled file is saved as a.out. So now when I say ./a.out, it will execute the program. I enter now.

(Refer Slide Time: 15:47)



```
bltembe@bltembe-Inspiron-3521:~/prog$ gfortran xte.f
bltembe@bltembe-Inspiron-3521:~/prog$ gfortran expte.f
bltembe@bltembe-Inspiron-3521:~/prog$ ./a.out
Input value of x whose exp(e) will be calculated
2.0
x, exp(x), exp(x)
2.00000000 7.38905608 7.38905608
bltembe@bltembe-Inspiron-3521:~/prog$
```

When I exit, when I enter, now it is asking me input value of x whose exponential will be calculated. So now all I have to do is to give some value. So let me give you like a value 2.0. This is the value I am giving and I enter. So when I enter it, it writes, remember in our program,

we asked it to write x, xtex as well as x specs. So it writes x which was 2, it calculated xtex. See what is the value now?

7.38905668, this is the value of the sum which our program calculated but the computer's function that is the gfortran function gives me 7.38905621, okay. So the only difference is our last 2 digits are 68 and the computer's last 2 digits are 21. So then the next question would be, how do I know which one is better? Because we took the gfortran compiler, it gave us some numbers.

We do not know how the program calculates it. Because we do not know the algorithm of gfortran. Our algorithm we know. So now next what I want to do, I want to use an algorithm which uses more digits than these 8 digits. Remember in this particular calculation, there were 8 digits after the decimal points. Count them again. 38905668, there are 8 digits after the decimal point.

So this is called single precision in Fortran. Single precision or it uses 32 bits of memory for these 8 digits. Now double precision is the next higher level of precision. It uses 16 digits after the decimal point. So 16 digits is certainly much more accurate than 8 digits. So how do I do with 16 digits that is called a double precision calculation. So for this double precision calculation, you have to make certain alternatives or changes in the main program that you will, I will illustrate by edit in the next program. So what is my next program?

(Refer Slide Time: 17:59)

```

Terminal
bitembeg@bitembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bitembe bitembe 141894 Oct 29 10:37 km.f
-rw-rw-r-- 1 bitembe bitembe 4399 Nov 3 18:48 matrix.f
-rw-rw-r-- 1 bitembe bitembe 3676 Nov 3 18:28 matrix-ok.f
-rw-rw-r-- 1 bitembe bitembe 4213 Nov 3 19:06 matrixsub.f
-rw-rw-r-- 1 bitembe bitembe 175 Oct 29 10:37 matrix.f
-rw-rw-r-- 1 bitembe bitembe 543 Oct 29 16:37 multimatrx.f
-rw-rw-r-- 1 bitembe bitembe 27 Oct 29 16:37 nidht.txt
-rw-rw-r-- 1 bitembe bitembe 6336 Oct 29 16:37 nn.f
-rw-rw-r-- 1 bitembe bitembe 9 Oct 29 10:37 ouput.dat
-rw-rw-r-- 1 bitembe bitembe 3 Oct 29 16:37 ouputd.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 ouputw.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 ouputx.dat
-rw-rw-r-- 1 bitembe bitembe 4237 Nov 3 10:33 outmatinvcorr.dat
-rw-rw-r-- 1 bitembe bitembe 2043 Nov 3 08:20 output1.dat
-rw-rw-r-- 1 bitembe bitembe 39 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bitembe bitembe 1701 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bitembe bitembe 32 Oct 29 10:37 outputd.dat
-rw-rw-r-- 1 bitembe bitembe 1334 Jan 15 17:32 prog5.f
-rw-rw-r-- 1 bitembe bitembe 157 Oct 29 16:37 quest.f
-rw-rw-r-- 1 bitembe bitembe 261 Oct 29 16:37 quest.f
-rw-rw-r-- 1 bitembe bitembe 447 Oct 29 16:37 quest.f
-rw-rw-r-- 1 bitembe bitembe 144 Oct 29 16:37 quest.f
-rw-rw-r-- 1 bitembe bitembe 23 Oct 29 16:37 rest.f
-rw-rw-r-- 1 bitembe bitembe 506 Oct 29 16:37 three.f
-rw-rw-r-- 1 bitembe bitembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bitembe bitembe 87 Oct 29 16:37 uj.f
bitembeg@bitembe-inspiron-3521:~/prog$ vi expte.f
bitembeg@bitembe-inspiron-3521:~/prog$ gfortran expte.f
No command 'gfortran' found, did you mean:
Command 'gfortran' from package 'gfortran' (main)
gfortran command not found
bitembeg@bitembe-inspiron-3521:~/prog$ gfortran expte.f
bitembeg@bitembe-inspiron-3521:~/prog$ ./a.out
input value of x whose exp(x) will be calculated
2.0
x, exp(x),exp(x)
2.00000000 7.38905608 7.38905621
bitembeg@bitembe-inspiron-3521:~/prog$ vi exptedp.f

```

I have called it vi expte, that is Taylor expansion exponential calculation. I will say dp, that is my other file. So what is this file? Similar to the old one, I have called it exptedp, that dp is for double precision. So let us see what this program is, okay.

(Refer Slide Time: 18:19)

```

Terminal
bitembeg@bitembe-inspiron-3521:~/prog
calculates functions by Taylor expansions
c
c real*8 (a-h,o-z)
c double precision x,sum,term,denom,y
c implicit real*8 (a-h,o-z)
c write(*,*) 'input value of x whose exp(x) will be calculated'
c read(*,*) x
c input value of x whose exp(x) will be calculated
2.0
sum = 1. + x
term = x
denom = 1.0
do 10 l=1,250
denom = denom + 1.
term = term + x /denom
sum = sum + term
10 continue
write(*,*) 'x, exp(x),deap(x)'
y = deap(x)
write(*,*) x,sum,y
end
"exptedp.f" 20 lines, 494 characters

```

Now look at this, the program is mostly the same but the differences are, look at the 1, 2, 3, 4. Look at the fifth line. It says implicit real*8 (a-h, o-z). So what this means is that all the variables that begin with a up to h and o up to z, remember all are the real variables, they begin with a to h and o to z. So implicit real *8 means, all the variables that I use in this program will be in double precision.

What is double precision. 16 places after the decimal point or it is 64 bit calculation. So before I gave this implicit real*8 a-h o-z, the first time I just wrote real*8 a-h o-z, this is what I had typed before. So when I execute it using real*8, it gave me an error, okay. Then next time I realized that real*8 may not be the correct way to do this particular a-h and o.z. Next time I did double precision x sum term denom y.

Remember this x sum term denom y, these are all the variables in my program. So if I declare all of them as double precision, then the program works properly. So double precision and real*8, they mean the same thing. The double precision x sum term denom y, what this line does? It makes all those 5 variables as double precision variables. Whereas if I give implicit real*8 (a-h, o-z), when I give implicit real*8, all variables beginning with a to h and o to z will be double precision.

So whenever you do very accurate calculations in chemistry, something like ab initio calculations or molecular dynamics calculations, the programs will do multiplications and additions millions of times. So when millions of times you are doing calculations, the errors can accumulate to very large numbers. So if you have double precision, it makes your calculation accurate at least up to 9 or 10 digits.

Because you may ignore digits starting from say 11, 12, 13, 14. So double precision is a very good method to use whenever you want very accurate calculations and when the program involves a large number of operations. So then what are the other differences, let us note here. So most of the programming lines are the similar except you will see that the rest are the program denominator do program everything is the same except when I say $y = dx x$.

Remember when you normally calculate in single precision, it is just $\exp x$. So whenever you do a double precision calculation, $dx x$ is the Fortran compilers version of exponential which is calculated to double precision. So our calculations are done up to 250 terms each in double precision. The Fortran compiler I have also used double precision. Now I want to see what results I get when I execute this, okay.

(Refer Slide Time: 21:51)

```
bltembe@bltembe-inspiron-3521:~/prog
calculates functions by Taylor expansions
C
C
C      real*8 (a-h,o-z)
C      double precision x,sum,term,denom,y
C      implicit real*8 (a-h,o-z)
C      write(*,*) 'input value of x whose exp(x) will be calculated'
C      read(*,*) x
C      sum = 1. + x
C      term = x
C      denom = 1.d
C      do 10 l=1,250
C        denom = denom + 1.
C
C        term = term * x /denom
C        sum = sum + term
10   continue
C      write(*,*) 'x, exp(x),exp(x)'
C      y = exp(x)
C      write(*,*) x,sum,y
C      end
```

So how will I come out of this? Escape shift :, when I type shift :, see that colon appears at the bottom of the screen. Then I type W, W or X is better. X means you exit and save.

(Refer Slide Time: 22:00)

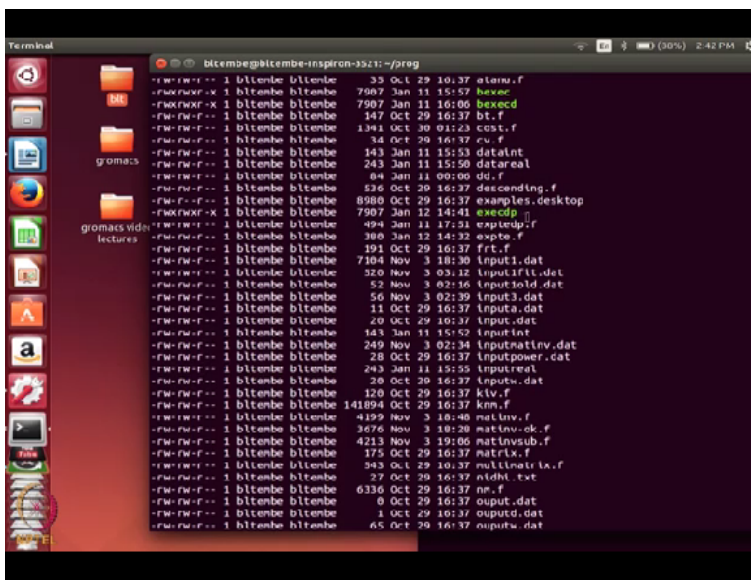
```
bltembe@bltembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bltembe bltembe 3070 Nov 3 10:20 natuvuk.f
-rw-rw-r-- 1 bltembe bltembe 4213 Nov 3 19:06 matnub.f
-rw-rw-r-- 1 bltembe bltembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bltembe bltembe 543 Oct 29 16:37 multmatrix.f
-rw-rw-r-- 1 bltembe bltembe 27 Oct 29 10:37 nihl.txt
-rw-rw-r-- 1 bltembe bltembe 6336 Oct 29 16:37 na.f
-rw-rw-r-- 1 bltembe bltembe 0 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 1 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bltembe bltembe 65 Oct 29 10:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 65 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 4237 Nov 3 16:33 outmatnubcorr.dat
-rw-rw-r-- 1 bltembe bltembe 2943 Nov 3 01:29 output1.dat
-rw-rw-r-- 1 bltembe bltembe 39 Oct 29 10:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 1761 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bltembe bltembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bltembe bltembe 1334 Jan 11 17:32 progfa.f
-rw-rw-r-- 1 bltembe bltembe 197 Oct 29 10:37 ques1.f
-rw-rw-r-- 1 bltembe bltembe 261 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bltembe bltembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bltembe bltembe 144 Oct 29 16:37 ques.f
-rw-rw-r-- 1 bltembe bltembe 23 Oct 29 10:37 rest.f
-rw-rw-r-- 1 bltembe bltembe 586 Oct 29 16:37 thrae.f
-rw-rw-r-- 1 bltembe bltembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bltembe bltembe 97 Oct 29 16:37 uj.f
bltembe@bltembe-inspiron-3521:~/prog$ vi expte.f
bltembe@bltembe-inspiron-3521:~/prog$ gfortran expte.f
No command 'gfortran' found, did you mean:
  Command 'gfortran' from package 'gfortran' (main)
bltembe@bltembe-inspiron-3521:~/prog$ gfortran expte.f
bltembe@bltembe-inspiron-3521:~/prog$ ./a.out
input value of x whose exp(x) will be calculated
2.0
x, exp(x),exp(x)
2.0000000  7.3890568  7.38905621
bltembe@bltembe-inspiron-3521:~/prog$ vi exptedp.f
bltembe@bltembe-inspiron-3521:~/prog$ gfortran exptedp.f -c excedp
bltembe@bltembe-inspiron-3521:~/prog$ ls -l
```

So I have entered, okay. So now I will compile the double precision program. How do I compile? Gfortran exptedp, so this is double precision program, .f, so I want to compile this program. Now when I, if I enter now, it will compile and save my compiled version in a file called a.out. But now you know every time I compile, I am getting a.out. So suppose I want to give the name for this executable file not a.out, but let us say excedp, this is a double precision program.

So I want to give the name for that executable file as expdp. So what do I do for that? After I

type gfortran exptedp.f, then -o. -o allows me to give a different name for my executable file. -o space execd. So I want to call the executable program execd. I want this new name now. Not a.out but execd. So I enter, so now it has executed. So whenever you get the next line without any comments or errors, so that means it has executed. So what did I call my new name? Execd. Now let us see my directory. So how do I see my directory? ls -l.

(Refer Slide Time: 23:41)



```
bltembe@bltembe-inspiron-9521:~/grog
-rw-rw-r-- 1 bltembe bltembe 53 Oct 29 10:37 alemn.f
-rwxrwxr-x 1 bltembe bltembe 7907 Jan 11 15:57 hexec
-rwxrwxr-x 1 bltembe bltembe 7907 Jan 11 16:06 bexecd
-rw-rw-r-- 1 bltembe bltembe 147 Oct 29 16:37 bt.f
-rw-rw-r-- 1 bltembe bltembe 1341 Oct 30 01:23 cost.f
-rw-rw-r-- 1 bltembe bltembe 34 Oct 29 16:37 cv.f
-rw-rw-r-- 1 bltembe bltembe 143 Jan 11 15:53 dataint
-rw-rw-r-- 1 bltembe bltembe 243 Jan 11 15:50 datareal
-rw-rw-r-- 1 bltembe bltembe 68 Jan 11 00:00 dg.f
-rw-rw-r-- 1 bltembe bltembe 536 Oct 29 16:37 descending.f
-rw-rw-r-- 1 bltembe bltembe 8980 Oct 29 16:37 examples.desktop
-rwxrwxr-x 1 bltembe bltembe 7907 Jan 12 14:41 execd
-rw-rw-r-- 1 bltembe bltembe 494 Jan 11 17:51 exptedp.f
-rw-rw-r-- 1 bltembe bltembe 380 Jan 12 14:35 exptg.f
-rw-rw-r-- 1 bltembe bltembe 191 Oct 29 16:37 frt.f
-rw-rw-r-- 1 bltembe bltembe 7104 Nov 3 18:30 input1.dat
-rw-rw-r-- 1 bltembe bltembe 520 Nov 3 03:12 input1fl.dat
-rw-rw-r-- 1 bltembe bltembe 52 Nov 3 02:16 inputold.dat
-rw-rw-r-- 1 bltembe bltembe 56 Nov 3 02:39 input3.dat
-rw-rw-r-- 1 bltembe bltembe 11 Oct 29 16:37 inputa.dat
-rw-rw-r-- 1 bltembe bltembe 20 Oct 29 16:37 input.dat
-rw-rw-r-- 1 bltembe bltembe 163 Jan 11 15:52 inputat
-rw-rw-r-- 1 bltembe bltembe 249 Nov 3 02:34 inputmatlv.dat
-rw-rw-r-- 1 bltembe bltembe 28 Oct 29 16:37 inputpower.cat
-rw-rw-r-- 1 bltembe bltembe 243 Jan 11 15:55 inputreal
-rw-rw-r-- 1 bltembe bltembe 29 Oct 29 16:37 inputw.dat
-rw-rw-r-- 1 bltembe bltembe 120 Oct 29 16:37 klv.f
-rw-rw-r-- 1 bltembe bltembe 141894 Oct 29 16:37 km.f
-rw-rw-r-- 1 bltembe bltembe 4199 Nov 3 10:40 matlv.f
-rw-rw-r-- 1 bltembe bltembe 3676 Nov 3 18:28 matlvok.f
-rw-rw-r-- 1 bltembe bltembe 4213 Nov 3 19:06 matlvsub.f
-rw-rw-r-- 1 bltembe bltembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bltembe bltembe 545 Oct 29 10:37 multimat1a.f
-rw-rw-r-- 1 bltembe bltembe 27 Oct 29 16:37 nldm.txt
-rw-rw-r-- 1 bltembe bltembe 6336 Oct 29 16:37 ne.f
-rw-rw-r-- 1 bltembe bltembe 0 Oct 29 16:37 ouput.dat
-rw-rw-r-- 1 bltembe bltembe 1 Oct 29 16:37 ouputd.dat
-rw-rw-r-- 1 bltembe bltembe 65 Oct 29 16:37 ouputw.dat
```

When I enter, so I got this directory. So now let us see what has happened. You see that there is an execd, this file has been created. Now this has been created just at this time, it is 2:41 now, so it is calculating, it has compiled this at this time. So when you do that, it gives the time in which it got this file created and you see that x here on the left side, so that means it is an executable file. So you have created a new file execd. So now I want to execute it.

(Refer Slide Time: 24:24)

```

Terminal
bitempe@bitembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bitembe bitembe 143 Jan 11 15:32 input.txt
-rw-rw-r-- 1 bitembe bitembe 249 Nov 3 02:34 inputmativ.dat
-rw-rw-r-- 1 bitembe bitembe 28 Oct 29 16:37 inputpower.dat
-rw-rw-r-- 1 bitembe bitembe 243 Jan 11 15:55 inputreal
-rw-rw-r-- 1 bitembe bitembe 20 Oct 29 10:37 inputw.dat
-rw-rw-r-- 1 bitembe bitembe 120 Oct 29 16:37 kv.f
-rw-rw-r-- 1 bitembe bitembe 141894 Oct 29 16:37 km.f
-rw-rw-r-- 1 bitembe bitembe 4399 Nov 3 18:48 matinv.f
-rw-rw-r-- 1 bitembe bitembe 3070 Nov 3 18:20 matinv-ck.f
-rw-rw-r-- 1 bitembe bitembe 4213 Nov 3 19:06 matinvsub.f
-rw-rw-r-- 1 bitembe bitembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bitembe bitembe 543 Oct 29 16:37 multmatrix.f
-rw-rw-r-- 1 bitembe bitembe 27 Oct 29 10:37 nldh.txt
-rw-rw-r-- 1 bitembe bitembe 6336 Oct 29 16:37 nn.f
-rw-rw-r-- 1 bitembe bitembe 0 Oct 29 16:37 outut.dat
-rw-rw-r-- 1 bitembe bitembe 1 Oct 29 16:37 oututd.dat
-rw-rw-r-- 1 bitembe bitembe 05 Oct 29 10:37 output.dat
-rw-rw-r-- 1 bitembe bitembe 65 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bitembe bitembe 4237 Nov 3 10:33 outmatinvcorr.dat
-rw-rw-r-- 1 bitembe bitembe 2943 Nov 3 01:29 output1.dat
-rw-rw-r-- 1 bitembe bitembe 39 Oct 29 10:37 output.dat
-rw-rw-r-- 1 bitembe bitembe 1701 Nov 3 19:06 output.dat
-rw-rw-r-- 1 bitembe bitembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bitembe bitembe 1334 Jan 11 17:32 progfs.f
-rw-rw-r-- 1 bitembe bitembe 157 Oct 29 10:37 ques1.f
-rw-rw-r-- 1 bitembe bitembe 261 Oct 29 16:27 que2.f
-rw-rw-r-- 1 bitembe bitembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bitembe bitembe 144 Oct 29 16:37 ques.f
-rw-rw-r-- 1 bitembe bitembe 23 Oct 29 10:37 rest.f
-rw-rw-r-- 1 bitembe bitembe 566 Oct 29 16:27 three.f
-rw-rw-r-- 1 bitembe bitembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bitembe bitembe 97 Oct 29 16:37 uj.f
bitembe@bitembe-inspiron-3521:~/prog$ ./excdp
input value of x whose exp(x) will be calculated
2.0
x, exp(x), dexp(x)
2.0000000000000000 7.38905609989306486 7.38905609989306504
bitembe@bitembe-inspiron-3521:~/prog$ ul ./excdp.f

```

How do I execute? Now I will not type a.out but I will type ./excdp. What this is doing? It will execute my program in double precision. So I enter. Now again I had a similar line, input the value of x whose exponential will be calculated, that is what I want to do. So let me give this number 2.0. So I have given 2.0 whose exponential I want to calculate in double precision. So I enter.

Now you see that I have this 2.000, this is my initial value. Now there are 16 places after the decimal point. Your exponential is also calculated. Now what are the values? 7.38905609989306486. You will see that on the other side, this one is the sum that we calculated and the second one is what we calculated using the Fortran compiler, okay. So where do they differ now?

Last 4 digits are 6486 whereas for the Fortran compiler, it is 6504. The first few are the same, 3890, look at this 3890, 5609, 5609, 8930, 8930, 6504, I have 504 here and 486 here. So the difference between the 2 calculations is about 18 digits different in the 15th and the 16th place. Now how do I know which is right? So one way to find out would be, you go back to your program and instead of 250 terms, let me use 300 terms, okay. So I will edit, vi exp Taylor expansion, dp.f, okay.

(Refer Slide Time: 26:16)


```

Terminal
bltembe@bltembe-inspiron-3521:~/prog
calculates functions by Taylor expansions
C
C
C
C
real*8 (a-h,o-z)
double precision x,sum,term,denom,y
implicit real*8 (a-h,o-z)
write(*,*) 'input value of x whose exp(x) will be calculated'
read(*,*) x
sum = 1. + x
term = x
denom = 1. d
do 10 l=1,350
denom = denom + 1.

term = term * x /denom
sum = sum + term /denom
continue
10
write(*,*) 'x, exp(x),decp(x)'
y = decp(x)
write(*,*) x,sum,y
end

```

So now what I want to do? I will change that calculation. I was doing 250 terms. Now I will do, so let us be, so instead of 250, let me do 350 terms, okay. So how do I change this 2 to 3? Type, I am in the edit mode, type r and just type 3. So it had changed 2 to 3, okay. So I have made 250 into 350. It is already in the escape mode. Shift :x, okay.

(Refer Slide Time: 26:49)

```

Terminal
bltembe@bltembe-inspiron-3521:~/prog
-rw-rw-r-- 1 bltembe bltembe 4199 Nov 3 18:48 matinv.f
-rw-rw-r-- 1 bltembe bltembe 3676 Nov 3 18:28 matinvok.f
-rw-rw-r-- 1 bltembe bltembe 4213 Nov 3 19:00 matinvsub.f
-rw-rw-r-- 1 bltembe bltembe 175 Oct 29 16:37 matrix.f
-rw-rw-r-- 1 bltembe bltembe 593 Oct 29 16:37 multmatrix.f
-rw-rw-r-- 1 bltembe bltembe 27 Oct 20 16:37 nidkl.txt
-rw-rw-r-- 1 bltembe bltembe 6336 Oct 29 16:37 ne.f
-rw-rw-r-- 1 bltembe bltembe 0 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 1 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 65 Oct 20 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 65 Oct 29 16:37 output.dat
-rw-rw-r-- 1 bltembe bltembe 4237 Nov 3 16:33 outmatinvcorr.dat
-rw-rw-r-- 1 bltembe bltembe 2993 Nov 3 01:29 output.dat
-rw-rw-r-- 1 bltembe bltembe 30 Oct 20 16:37 outputa.dat
-rw-rw-r-- 1 bltembe bltembe 1701 Nov 3 19:00 output.dat
-rw-rw-r-- 1 bltembe bltembe 52 Oct 29 16:37 outputd.dat
-rw-rw-r-- 1 bltembe bltembe 1339 Jan 11 17:32 prog3.f
-rw-rw-r-- 1 bltembe bltembe 157 Oct 20 16:37 ques1.f
-rw-rw-r-- 1 bltembe bltembe 261 Oct 29 16:37 ques2.f
-rw-rw-r-- 1 bltembe bltembe 447 Oct 29 16:37 ques4.f
-rw-rw-r-- 1 bltembe bltembe 194 Oct 29 16:37 ques5.f
-rw-rw-r-- 1 bltembe bltembe 23 Oct 20 16:37 rest.f
-rw-rw-r-- 1 bltembe bltembe 506 Oct 29 16:37 three.f
-rw-rw-r-- 1 bltembe bltembe 197 Oct 29 16:37 two.f
-rw-rw-r-- 1 bltembe bltembe 97 Oct 29 16:37 uj.f
bltembe@bltembe-inspiron-3521:~/prog$ ./execdp
input value of x whose exp(x) will be calculated
2.0
x, exp(x),decp(x)
2.0000000000000000 7.3890560989366486 7.3890560989366504
bltembe@bltembe-inspiron-3521:~/prog$ vi exptedp.f
bltembe@bltembe-inspiron-3521:~/prog$ gfortran exptedp.f -c execd
bltembe@bltembe-inspiron-3521:~/prog$ ./execdp
input value of x whose exp(x) will be calculated
2.0
x, exp(x),decp(x)
2.0000000000000000 7.3890560989366486 7.3890560989366504
bltembe@bltembe-inspiron-3521:~/prog$

```

So what I did? Now I changed 250 to 350, compiled it, sorry saved it. Now I will again compile, gfortran, okay expted.f, okay. I will compile it again, okay. Now I will execute. How do I execute? ./execdp enter. So now I have to input the value. I will put 2.0, enter. Now see what values we got? So I again got 486 in the last digits and the Fortran compiler is 504. So what do you conclude using this?

That my value has not changed which means whether I calculate up to 250 terms or 350 terms, I am getting the same result. So I know that this has converged. So I would like to trust my calculations much more than the Fortran compiler's calculations because I know that no matter whether I take 250 terms in the sum or 350 terms, I am getting the same result.

So therefore, this is another example which illustrates that when you want to know whether a result is converged, you calculate for more and more terms and if your value does not change, you know that it is a converged result. So what we have done today? We have compiled the program which calculates an exponential function using a Taylor series and we got the results.

And to know whether my result is accurate or not, I used double precision, double precision means I take 16 digits after the decimal point. Then I calculated using 250 terms, I got a result. Then I used 350 terms, I got the same result. So now I know that it is a converged result. So this is how you can know how to get a converged result by taking more and more terms in your sum. So we will conclude this particular lecture today.

And in the next lecture, I will tell you how to use these same programs in functions and subroutines. Remember, last time we have used functions and subroutines. So I want to use several functions and several subroutines and execute in my next session. So I will conclude here. Thank you.