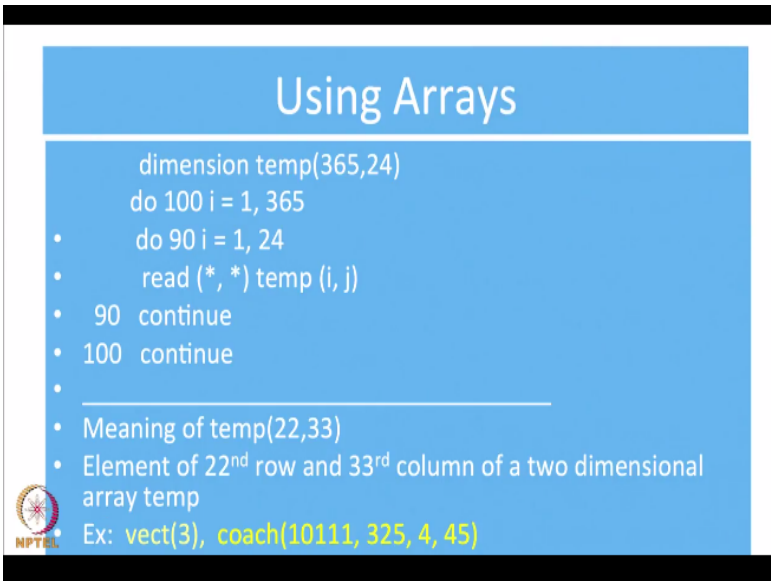


Computational Chemistry & Classical Molecular Dynamics
Prof. B. L. Tembe
Department of Chemistry
Indian Institute of Technology – Bombay

Lecture - 06
Programming Techniques 4. Arrays and Matrices

Hello again. So last time what we have done, we have started the discussion of arrays. So array variables are such that they take on more than one number associated with that name. For example, we use this temp of 365, 24. What does it mean?

(Refer Slide Time: 00:36)



Using Arrays

```
dimension temp(365,24)
do 100 i = 1, 365
•   do 90 j = 1, 24
•   read (*, *) temp (i, j)
•   90 continue
•   100 continue
•
```

- Meaning of temp(22,33)
- Element of 22nd row and 33rd column of a two dimensional array temp

Ex: vect(3), coach(10111, 325, 4, 45)

There are 365 rows and 24 columns. So this allowed us to store the data of 24 temperatures per day for 365 days in one particular variable temp. So this temp is now a subscripted variable or an array variable. So to declare that it is an array variable, we have used this word dimension. There are other ways of doing it other than dimension. We will use it as we go along. So then suppose I have a vector with 3 components, then I will call it look at the last line vect into bracket 3.

It means that it is a vect is an array with maximum 3 components in it. So this has to be declared in the dimension statement. So going back, see in the dimension statement I declare and in the program I give specific values of i and j. Once I give the dimension statement, I do not have to say anywhere else in the program what its range is, it is the system will now that i and j will take on values i going from 1 to 365, j going from 1 to 24.

And once you give the dimensions, these subscripts cannot take more than the value declared in the dimension. The dimension gives you the maximum size. In the last line, I have given you a vector which has 3 components so maximum size is 3. Then, we did talk about railway reservation system. So you know that there are a very large number of trains. So if I want the total data for all the people in the country.

So I should first of all know how many trains are there, how many days are there and how many coaches are there in trains and how many seats are there in each coach. So we know that the train numbers go from 0 to let us say 99999. So there are not more than a lakh trains in India. So the first variable I want to say it in this case I have given a random value because this slide was made a few days before all these train reservation started.

So let us say the first variable tells you the number of the train, second variable tells you the day of the journey, third variable tells you which is the coach number and the last variable tells you the seat number. So this way if my dimensions are this that means it allows 10,111 trains, it allows up to 365 days, it allows up to 4 coaches per train and 45 seats per coach. So how will I modify this to include all the trains?

First thing, I will do I will give a dimension of 99999, so that allows me to take care of all the trains in the country. Then, instead of 325 I will make it 365 okay. So that means it allows me to store the date for 365 days. Then, instead of 4 here, instead of 4 I will give around 30 because there is no train in the country which is more than 30 compartments. So the third will be 30 and the last one I will give 100.

Because there is no coach which has more than 100 seats but maybe I will just make it about 200 because some of the seating in trains can be up to 106 seats. So if I declare an array of this size, it will be able to save the information of all the people who have reserved trains who has reserved seats in the train. So this data will have information about all the trains, all the days, all the coaches and all the seats in the coaches.

Now what will happen? Suppose 1 day is already over, then I do not need the data of the day. Suppose today is let us say 1st of January, on 2nd January I do not need the data of 1st January. So therefore I can use the data of 1st January of next year in this array. So once the

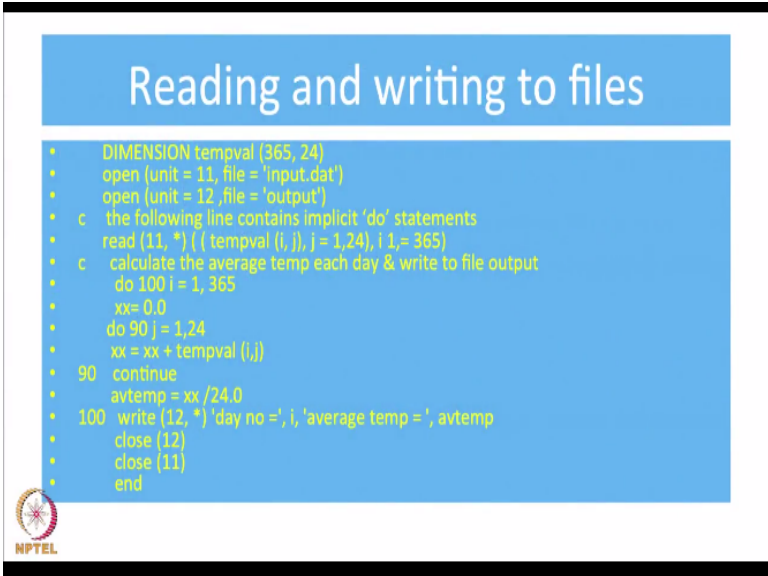
days are over, the previous data I will not read, I will not need but I will still store it for future information.

So this is how you will declare a large array to take care of all the trains and all the coaches. Now let us say your Aadhaar number. The Aadhaar number has 12 digits okay, so that means the information about people up to 1 followed by 12 0s. The information about so many people about can be taken in this Aadhaar system because the numbers are taking that range of 12 digits.

So that is how we will declare dimensions. So right now what we are considering that for our purpose in chemistry all the variables will be either real variables or integer variables. We are not interested in special data sets where the data can have name, birth date. Remember Aadhaar data will have your name, birth date, the finger prints. So those are very complex data sets where the information in your data is very complicated information.

So those are different data structures. For our purpose, our data structures are very simple. We will either take real numbers or integers. We will also have character data as we go along I will come to the character data. So now let us see what we want to do with this temp data that I have created. So I want to use a program which will give me the average temperature for each day and the average temperature for the whole year. That is what I want to calculate.

(Refer Slide Time: 06:36)



The slide is titled "Reading and writing to files" in a blue header. Below the header, on a light blue background, is a list of Fortran code statements. The code includes opening two files, reading data into a 2D array, calculating the average temperature for each day, and writing the results to an output file. The NPTEL logo is in the bottom left corner.

```
• DIMENSION tempval (365, 24)
• open (unit = 11, file = 'input.dat')
• open (unit = 12, file = 'output')
• c the following line contains implicit 'do' statements
• read (11, *) ( ( tempval (i, j), j = 1, 24), i = 1, 365)
• c calculate the average temp each day & write to file output
• do 100 i = 1, 365
•   xx = 0.0
•   do 90 j = 1, 24
•     xx = xx + tempval (i, j)
•   90 continue
•   avtemp = xx / 24.0
• 100 write (12, *) 'day no =', i, 'average temp =', avtemp
• close (12)
• close (11)
• end
```

So now I have changed my situation a little bit. So what I want to do is that all these data, now suppose I want to read from the computer. In the earlier program, I have to read these

365, 24 from the screen that is I will go on feeding this 24 temperatures per day for 365 days it will take me several hours to give this data. So that is not possible because if I make some mistake instead of 20.2 suppose I make it 20.3 the data is wrong.

So I will not be able to enter and read this data. So one way to do it would be to create a file where I do not worry about my program, I just input all these data in one file okay. Then, I want to read that data from that file into the computer again. So what this particular slide is showing how to read and write from files. So this is a slightly different topic than the dimension.

Because I need the dimension for declaring something to read from a file I need some more commands. So now look at this particular program. The first line says dimension tempval 365, 24. So this is now an array. This is the declaration statement. You declare the temp values in array with 365 rows and 24 columns. Now the next line is the crucial new line. It says open (unit=11, file='input.dat').

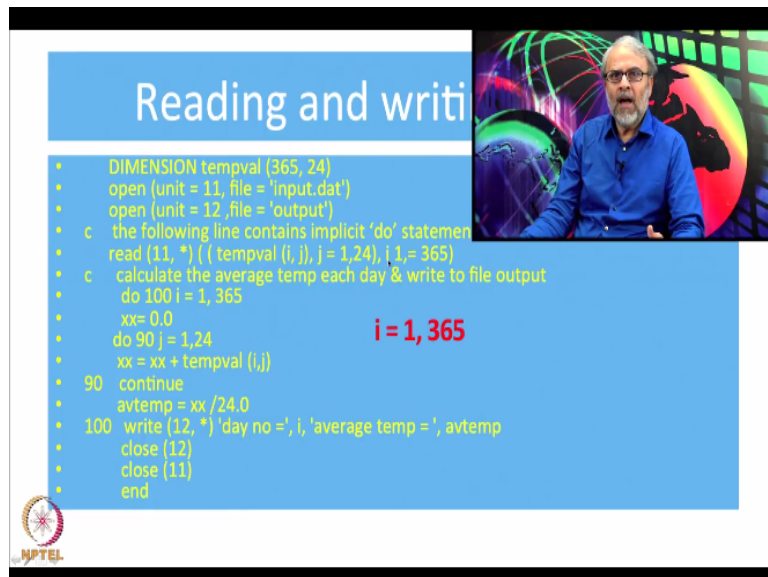
So what this line does, it assigns a number 11 to a file called input.dat. So what you do you open a file input.dat using a vi command. So in that file you enter all those data 365*24 and save that file. So when you want to execute a new program, open unit=11, file=input.dat, this particular line assigns number 11 to that file. So file name has been assigned a number 11. What is the second line?

It says open another file called 12 number 12 that file name is output okay. So what you have done? You have two statements, one for input.dat, one for output. Both these files are assigned to these numbers. So now let us see how I want to use these numbers in a read statement. So in next line, so this is a read statement now, it says read (11, star) okay*((tempval i, j, j going from 1 to 24, i going from 1 to 365.

So the first change you will notice is that earlier statements were read star, star okay. All our earlier statement was read star, star because the first star referred to the screen. Now for the first time instead of a star I have 11 that 11 means I read data from the file input.dat. So this is the first way my read statement directs the program to read from this input.dat because of this 11.

So now I should have given two do loops, remember in the last program I had two loops, one for variable i and one for variable j okay. So what I have done, those both do loops can be written in a single line by this particular procedure. So what I have done ((tempval i, j okay, j going from 1 to 24, it says that the do loop is j going from 1 to 24), then i going from 1 to 365. Actually, there is a slight error here.

(Refer Slide Time: 10:31)



The slide is titled "Reading and writing". It contains a list of Fortran code snippets and a video inset of a man in a blue shirt speaking. The code snippets are:

- DIMENSION tempval (365, 24)
- open (unit = 11, file = 'input.dat')
- open (unit = 12, file = 'output')
- c the following line contains implicit 'do' statement
- read (11, *) { (tempval (i, j), j = 1,24), i = 1,= 365)
- c calculate the average temp each day & write to file output
- do 100 i = 1, 365
- xx= 0.0
- do 90 j = 1,24
- xx = xx + tempval (i,j)
- 90 continue
- avtemp = xx /24.0
- 100 write (12, *) 'day no =', i, 'average temp =', avtemp
- close (12)
- close (11)
- end

A red text overlay on the slide reads "i = 1, 365". The video inset shows a man with glasses and a blue shirt speaking. The NPTEL logo is visible in the bottom left corner.

It should be i going from 1, 365. This equal to sign should be after i. See here j going from 1 to 24, i=1, 365 okay. So this way, I read all the information from file which is labeled 11 and both do loops are exactly in one particular line. So you will see that long ways of writing do loops, line numbers everything is all taken care of by just this one particular line. It is a very compact line where in this one line I have executed both do loops and read the information from file 11.

So once I have this data, now I want to calculate the average temperature of each day and write the file to the output. Now look at this, this is a comment card. I have already read the data, now I want to calculate the average temperature of each day and write it into some file. So let us see how to do it. So to do that I start a loop do 100 i going from 1 to 365. What it says that I have a do loop which goes up to line number 100 okay.

And it will repeat all these commands up to line number 100, that is the meaning of a do loop. Now this do loop has line 100 as the last line and i as the variable, the variable goes from 1 to 365. Now you will see that it is obviously the case that i going from 1 to 365 is a

loop variable for the days in the year okay. So I want to do a calculation for each day, so next I set a variable $xx=0$.

So this is a dummy variable but first now I have set that $=0$, then the next line is do 90 j going from 1 to 24. This is a second loop, that loop extends up to line 90 and which and the loop goes from 1 to 24 that is it will the j will take on values from 1 to 24. First time it is 1, second time it is 2, third time once j is 24, it will complete this part then go to the next line and complete that first loop.

So let us go to the inner loop, so now these are called nested loops. Nest means there is one outer loop and one inner loop and always ensure that the inner loop completes before the outer loop. So now I have started the outer loop. The inner loop $x=0$, it goes through the second line. So the next line here $xx=xx+\text{temp value } i, j$. Remember, I am in the inner loop, in the inner loop, only j changes.

So what was the first value of i, i was $=1$ so when $j=1$, so the first time this is executed, xx will be $0+\text{the first value of temperature, first day, first temperature}$. So I go to 90 continue go back to 90. Now j has become 2, when $j=2$, I come to the second line, $xx=\text{the first temperature}+\text{the second temperature of the day}$. First day again, $i=1, j=2$. I go to 90 continue, come back $j=3$.

Now I add the third temperature of the day, go here go back, so when I execute this loop my xx has added all the 24 temperature of the first day because i has not changed, only j has changed. So once I add that my xx will be the sum of all the temperatures for a given day. So the next line is $\text{average temperature}=xx/24$. I have added all the 24 temperature in xx . When I divide by 24, what this will give is the average temperature for that day.

So, so far I have calculated the average temperature of the first day. Then, I go to the next line, it says write 12, star) Now see instead of 11, I have 12 that means whatever I write it will be written in a file whose unique number is 12. So 12 was output. So what will it write in that file number 12? It will write 'day number, so that means it will write day number, then i, so it will write i was 1, so it will write day number.

Remember whatever is in the quotation marks that is written on the screen. So it will write day number=, i which was 1 then again, again in quotation average temperature=this will write again on the screen, avtemp, avtemp was calculated in the previous line. So through this write statement, I will write the first time I come to line 100 that is when i is=1. I write day number=1, average temperature=the value of the temperature.

Once this is done, I go back to the first loop that is the outer loop, do 100. So i=1 I have already run the calculation, now i becomes 2, when i becomes 2 I go to the next line, the xx is assigned value 0 again. It is assigned 0 again, then it goes to the second loop. What does the second loop do? It takes the value of xx for the second day. All the values now i=2, j goes from 1 to 24.

So for the second day all the temperatures are added in this variable xx, the loop is completed. Then, I go here calculate the average temperature of the second day, go to the write statement, so the second line of the write statement will be day number=2 average temperature=average temperature of the second day. Go back, now i becomes 3 third day, again reset xx=0.

This resetting is always important because if you do not reset=0, it will add all the temperatures of all the days, you do not want that right now. You want only the average temperature of each day. So every time you want to calculate the average you reset xx=0, add the temperature for that day, write now we have up to i=3, day number=3, average temperature=avtemp.

So like that when you repeat this for 365 times, there will be 365 lines written and each line is the average temperature of that day okay. Then, once you finish this loop close 12, remember there is an open statement, open means you are opening that file, it is always a good idea to close that file which is like a cupboard whichever cupboard is there you open, use all the information close it, close 12, close 11 then end okay.

So once the program ends you can edit your file, output is the name of the file. When you now all these lines that are written they will not be written on the screen because it is write 12, star not write star, star. If for to the star, star it will write on the screen. So since it is 12,

star the computer will execute everything and you will see nothing on the screen and you will get the cursor back.

So once you get the cursor back, do vi output and you will see that in that particular file, all these 365 lines are written and each line has an average temperature for that particular day. Now let me post the question. Suppose I want the average temperature of the whole year; how will you do that? The way to do would be I have to add the average temperature for all the 365 days and divide by 365 okay.

So that is how I will do that is whenever I calculate this average temperature okay, let me define a new variable called yy okay, new variable yy. The value of yy I will set=0 before this do loop okay. So that yy is a new variable which is set=0 here and every time I calculate the average temperature after this line, I will say $yy=yy+\text{average temperature}$. So what this will do?

It will take the average temperature of the first day, add to yy, second day add to the first, third day add to the first and second like that I would have added yy for all the days and finally when I come outside this line I will say average temperature for the whole year, let me call it say w, w will be $yy/365$. So that is my average temperature for the whole year. Then, just as I have written 12, these 365 lines I will have one more statement which will write me the average temperature for the whole year.

I will try to ask you to do this as an exercise. Do this as an exercise, so that you will know how to improve on whatever is given. There was one error here, remember it was i space 1,= this was not correct, it should be $i=1, 365$ okay. So in fact it is a good thing once in a while to have a wrong statement, so that then the computer will not run, it will give you an error. So you go back to that statement, by vi editor correct that and again compile and execute.

So this completes our use of the dimension statement, use of an array variable. So in the next program, I will use array variables to arrange numbers in an ascending order okay. Arranging objects in an ascending order is very important. It is a very important problem arranging things okay. So the next program tells me how to arrange okay.

(Refer Slide Time: 20:19)

THE MATRIX MULTIPLICATION PROGRAM

A program to multiply two matrices is given below.

We shall consider only square matrices.

The (i, j) th element of the product c of two $n \times n$ a and b matrices is

$$c(i,j) = \sum_{k=1}^n a(i,k) * b(k,j), \quad k = 1, n$$



So this is I will come to this is a matrix multiplication, I will come later okay. So alright let us do this matrix multiplication rather than arranging numbers in an ascending order. This is also, this is a fairly easy program compared to arranging numbers in an ascending order. So I want now to do a matrix multiplication. So what is matrix multiplication? You are given two matrices, let us say first matrix is a and second matrix is b.

And you want the product of these two matrices okay. I will call the product matrix c, so the elements of c are represented by a i, j; c i, j and the elements of matrix a and b are a i, k and b k, j. So what is the product c i, j? You know from a matrix multiplication program is that c i, j = sum of a i, k * b k, j k goes from 1 to n. This formula is the sum, so that is the i, jth element of the product matrix is you take the ith row of matrix a, jth column of matrix b and multiply ith row by jth column okay.

So that is what this is, so you see that this is the ith row of matrix a and jth column. Here j is not changing, so it is the jth column, i is not changing which is the ith column and k is changing, so it will be like a i, 1 b 1, j to that you add a i, 2 b 2 j, a i, 3 b 3, j. So you are taking the ith row and multiplying by the corresponding element of the jth column of matrix b. So this is the formula, so whatever next program we will do it will read matrix a, read matrix b and calculate all the values of matrix c.

This is a conceptually easy program but as we saw earlier you have to be careful when you initiate values in calculating. If you do not initiate properly you can get very, very funny and strange results okay. So next will be our program.

(Refer Slide Time: 22:40)

```
c      program matrix multiplication
      dimension a(100,100), b(100,100), c(100,100)
      open (unit=11, file='mata.dat')
      open (unit=12, file='matb.dat')
      open (unit=13, file='matc.dat')
      write (*,*) 'value of n of the n x n matrix is = '
      read (*,*) n
      read (11, *) ( ( a(i,j), j = 1,n ), i = 1,n)
      read (12, *) ( ( b(i,j), j = 1,n ), i = 1 ,n)
1      continue
      do 10 i = 1, n
      do 10 j = 1, n
      sum = 0.0
      do 5 k = 1, n
5      sum = sum + a(i,k) * b(k,j)
      c(i,j) = sum
10     write (13, *) ' i, j, c ( i, j) = ', i, j, c(i,j)
      close(13)
      close(12)
```

This is a program to multiply matrices, so from what we have seen earlier since these are array variables, so I have to have a dimension statement. This particular program I want to write for matrices of the size 100/100. The maximum value I want to consider is 100/100. So my first line will be dimension, a 100, 100, b 100, 100, c 100, 100. So what this does? It declares 3 arrays, one called a of 100/100, second array b 100/100, third array c 100/100.

So these are the matrices a, b and c. So since there are 3 matrices, I want to read it from 3 files. Now you will ask me why do I want to read from 3 files, 100/100 is 10,000 data points. You cannot possibly enter 10,000 on a screen without making a mistake, so there is no point in giving data on the screen, it is best to write all the data when you are relaxed and at your convenience you can create the matrix a and create the matrix b at your convenience.

So what are my units now? 11, 12 and 13. These are the 3 units to associate files mata.dat, matb.dat and matc.dat. My file names are mata.dat, matb.dat, matc.dat okay. So through this units you have to link these numbers to these files okay. So this is again the statement links these numbers to the files which are given in the bracket. All the file name should be given in single quotation marks.

So once you have assigned these numbers, the next thing what it will write on the screen, write star, star. So now you see, I am using writing on the screen here. It will write on the screen this particular line; it says 'value of n of the n x n matrix is='. It will write on the

screen value of n , so you will know when you execute the program that it will write this line on the screen.

So you will know that you have to give the value of n to your program as an input. So once you see this line on the screen, read star, star n . So you are giving n which is a size of the matrix. Once you give the size of the matrix, your first job is to read matrix a and b . Remember, last time I have given a read statement where the read line itself has an implicit do loop in it.

Why do I call it an implicit do loop? There is no do written here but it is understood the program understands that the variables take on these values. So now let me read the first matrix read 11, star, 11 refers to the first matrix mata.dat, read from that file okay ((a i, j, j going from 1 to n i going from 1 to n okay. So I am reading the full matrix a by this procedure.

Now again look at this, the inner variables are executed first and then the outer variables. So when I write a i, j; j goes from 1 to n okay. So that needs when I have a i, j always i and j are equal to 1 the first time. So I will read the first column and then the second time $i=2$ then I will read the second column, then third column and so on. So the way this is written, columns are read first and then the rows.

So this way I read matrix a , then next I will read matrix b , all elements of matrix b . So now I ask you the question how many elements are there? n I have read from the screen, so if n is 10 there will be 10/10 for matrix a and 10/10 for matrix b . So now you know how to read information from the files into a matrix, so I will stop here. You try to think of how to calculate the product.

So this is already given in the slide, do not look at the slide, just see how you can write a program to calculate the product. The product is matrix c , you have to calculate all the elements c 1, 1 to c 10, 10. I will conclude here. Think about it, in the next class we will revise this assigning units to files, assigning dimensions and complete the execution of this matrix multiplication and also we will see a program how to arrange numbers in an ascending order okay.

So this is what we have covered in the last few lectures, lot of items in theory. From next class, we will actually see all these things on the screen. So that whatever you have read you will understand and execute and remember in computing whatever you see in a lecture that has meaning only when you execute it on a computer, only when you execute and get the result that is your final learning, not just listening to this lecture. So I will conclude here. Thank you.