Problem Solving through Programming in C
Prof. Anupam Basu
Department of Computer Science and Engineering
Indian institute of Technology, Kharagpur

Lecture-48
Arrays at Strings

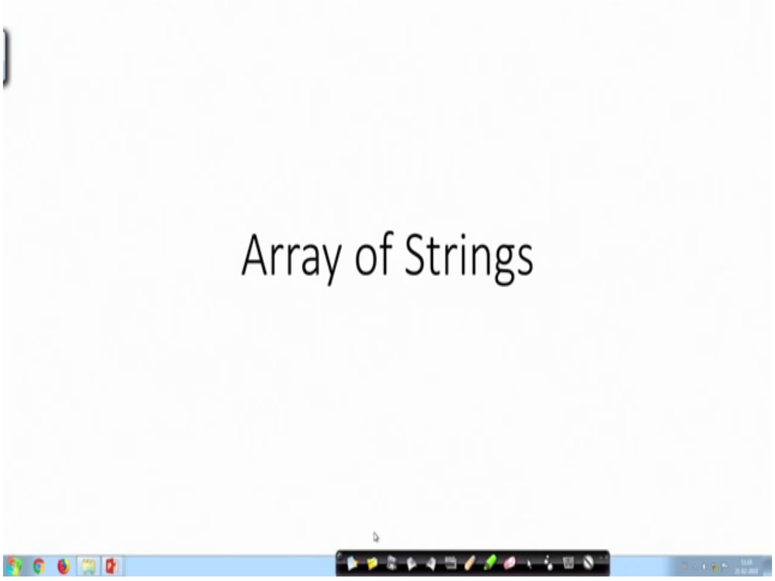
Today, we will be having a look at how the techniques that we have learnt can be applied to different Problem Solving and this will continue from this lecture onwards for a few more lectures. We have learnt quite a few things we have seen the overall structure of the C language.

We have seen the if then else if l structure the iteration structure, like the loops we also know how the variables are defined, we have seen arrays how data can be stored in 1 dimensional array and 2 dimensional arrays, we have also seen functions. And the most important thing about functions is that it enables us to take a complex problem and divide it into different parts and solve each of the parts separately using functions. A very important thing about functions is passing of the parameters and how the function returns of value returns a value to the calling function.

We have also seen that there are two ways in which parameters can be passed from the calling function to a called function, one is called by value another one is called by reference in c language in general always call by value is adopted.

However, in call by value we have to copy the value of the data the parameter value the parameter value to the argument value. However, for the case of arrays whenever I want to pass a particular array at that point an array can be a 100 500 array or 10 by 10 array if it be a 10 by 10 array then I will have to copy hundred values. So, that is not advisable for that we just passed the array name.

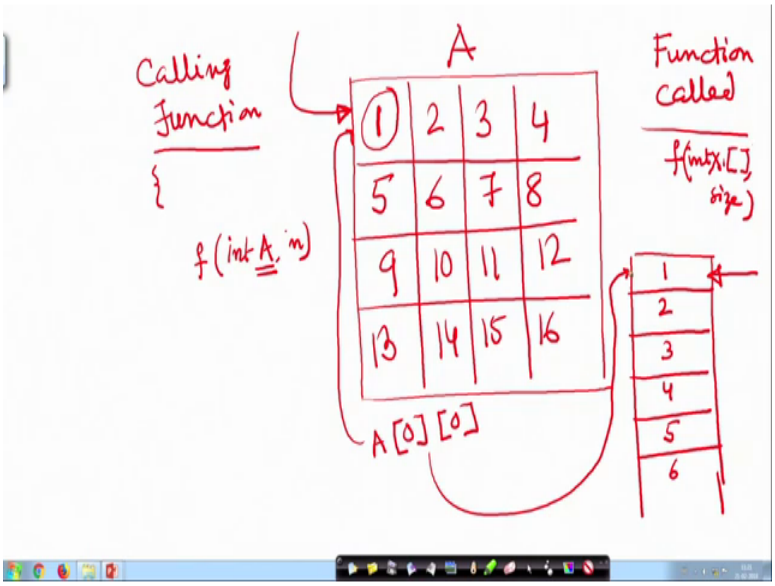
(Refer Slide Time: 02:49)



Array of Strings

And let us quickly have a look that the array name is essentially an address right and array whenever we pass suppose there is an array or 2 dimensional 10 by 10 array all right 2 dimensional array like this.

(Refer Slide Time: 03:09)



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

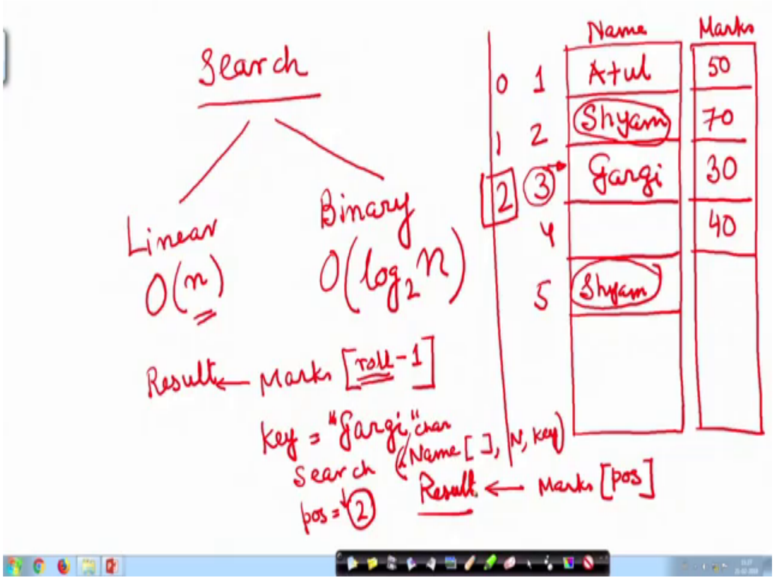
Now, all the elements I am not passing. So, it is a essentially it happened to be a 4 by 4 area. So, that is all the 16 elements I will not be passing. So, here is my called function the function called function and on this side is the calling function, the calling function and the array name is say a. So, the calling function refers to suppose let this function be called f.

So, in from the body of the calling function whenever I am calling f, I am passing the array say if it be an integer array into A. And here I just passed the array A and here in this function body we have got say int A and just saying that the size of the or if it may not be a it can be x also, int x something and the size of that and here int A and maybe n. So, here I am actually passing the name of the array and the name of the array is nothing, but the beginning of the address of the first element of the array. So, here I am actually copying the address.

So, it is a call by reference in the case of an array. And in the last lecture we have also seen that we can also use pointers to sort of establish call by reference. Another major thing that you have to keep in mind is that suppose this array is 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16. Now, these are organized in C in a row major form. So, it is stored in the memory like 1. So, the rows first 2 3 4 then 5 6 like that. So, when I refer to the array A I actually pass on the address of the first location of the array ok. So, A 0 0 in the case of A 2 dimensional array points to this element and that is this element that is how we pass on arrays.

So, for functions and we have also seen some applications of all these techniques that we have learnt in particular we have seen how we can carry out search.

(Refer Slide Time: 06:40)



And we have seen two search techniques one is linear search another is binary search. Now in linear search the complexity the time is proportional to the number of elements whereas, in binary search it is reduced it is faster it is log of the number of elements to the base 2. And search is a very fundamental problem solving technique in technique, because suppose you have got the names of students in the class all right.

Here are the names Atul, Shyam, Gargi in that way the names are there and each of them have got a roll number 1 2 3 4. Now you want to find out how many Shyams are there in the class. So, you have to search this array and find out where Shyam is ok, maybe here and there is another Shyam here. So, that is searching and counting both the things, but you cannot avoid searching this array all right. It may be that along with that there is an array of marks obtained by roll number 1 roll number 2 etcetera.

So, maybe this is 50 this is 70 this is 30 all right. Now here tell me I want to say now they are in 2 arrays this is 1 array called name and this is another array called marks of the corresponding students. Now I want to know how much what is the marks that Gargi has obtained. How do you go about it what would be the first step in solving this problem what is the marks that Gargi has opted, in order to do that I have to first I do not know what is Gargi is roll number.

If I had known what is Gargi's roll number suppose Gargi's roll number is 3, then I could have gone in the marks array and I could have taken marks roll number what if I knew the roll number, roll number minus 1 that marks I could have taken as a result all right, but I do not know the roll number if I had known the rule number 3 then I would have gone to the second element. So, it is marks 2 would have been would have given me the roll number, but suppose I do not know the roll number then what should I do?

First I will have to search this area and what would be my key what is my key will be the screen Gargi, because I want to know the marks obtained by Gargi and what should I do with this key I should carry out a search, it can be a linear search or it can be a binary search let us for the time being assume that it is a linear search. So, we will be searching and I will be finding Gargi here I am assuming for simplicity that there is only one Gargi in the class.

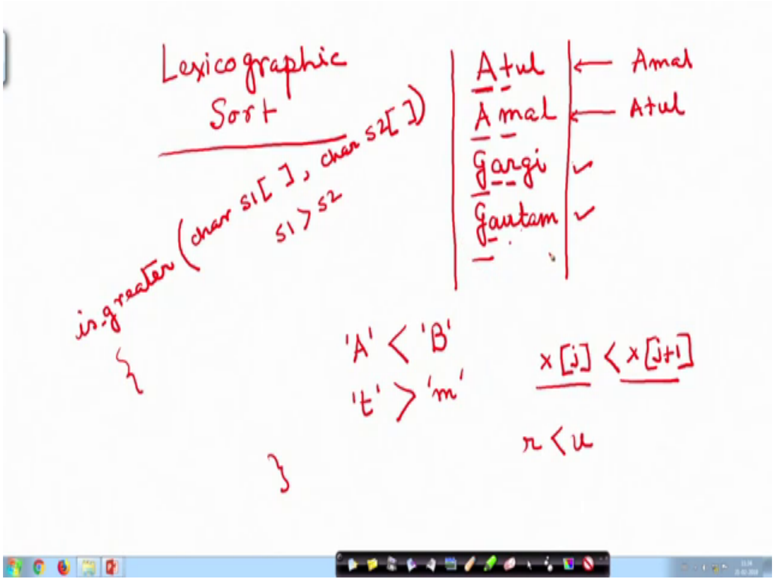
So, I will take this and I find what will the search, if you recall what will the search algorithm return me the search algorithm will return me the index of the position where

Gargi is located. And that position will be actually it will automatically come since this positions will be marked as 0 1 2 I will get this 2 returned. So, I will get. So, I get the key I search the array name this is a character array A care name size whatever size is say n and the key.

And this search will return me the value position 2 pos equal to 2, then I will simply take the value of marks I go to the array marks pos, which will be my result right. So, you see we needed the searching. Now, while I was talking of searching this array I said that the searching I am assuming this searching to be a linear search.

Would it be possible to sort this array using binary search, if you just think it over would it be possible to search, it using binary search what does binary search require binary search requires that the array be sorted right array be sorted.

(Refer Slide Time: 13:00)



So, an array like this, Atul, Amal, Gargi and Joy this is an array of 4 elements can I sort it the type of sort if, I want to sort it I will have to organize it in increasing or decreasing order let us assume that I am trying to do it in an increasing order. Now, this sort will be this character array can be sorted in an way which is called the lexicographic sort.

That means I go I sort the character array according to the alphabets or the words as they written; obviously, A comes first. So, therefore, A character A is smaller than character B all right.

So, here between these two elements both of them has got A, but similarly t is greater than m. So, in this sorting I will first check between these say a they are same think of bubble sort I am trying to compare them in normally what we did we found that say array x xj is less than xj plus 1 and then we did something.

If it was if it is the case no issue if it was the other way round I wanted to do a swap, but in this case it will not be. So, simple I can take a string and compare these two strings as to which one is whether they are same or not or whichever is greater, but another way you can write a small function to compare I leave it to you I leave it to you that you now know sorting techniques.

So, you should try to write a function that will sort a string of characters all right. So, e and e are same, if they are same the function what will do it will now look at t and m now m is smaller than t therefore, Amal is smaller than Atul.

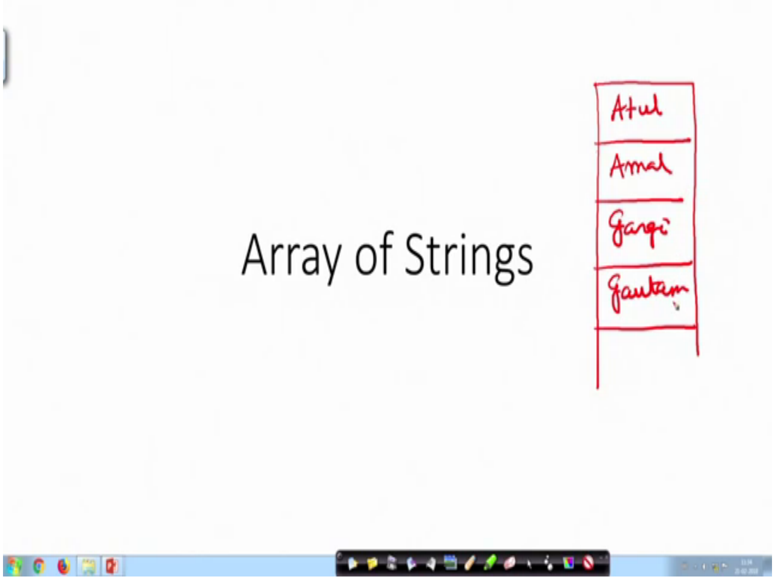
So, Amal will come here and Atul will come here now between Atul and Gargi you can see a is greater than smaller than G. So, Gargi is in a proper space please. Now if say for example, I do not have Joy here I am having another name Gautam here. Now when I am comparing between Gargi and Gautam G and G are the same.

So, they are equal no change next a and a are also same no issue. So, there is no change, but r r is smaller than u, because r appears earlier than u, but in terms of our computer code technology the Ascii code of r is less than the ascii code of you.

So, in both the ways it is meaningful. So, r is less than you therefore, Gargi is smaller than Gautam. So, in that way I can carry out with another small function compare string is greater say is you can write a function is greater 2 strings ok, care S 1 comma care S 2 2 strings will be passed on this function and this function will find out, which character if S 1 is greater than is 2, it will just return if S 1 is greater than is 2 it will return A 1 if S 1 is greater than S 2 try to write such a function ok. Now, that is that I am giving you as a home assignment all of you should try to do that and with a little effort I am sure you will be able to do that.

So, I could have just as I had done in the earlier one, here in this thing I said that I will be doing linear search, I could have done binary search also provided I had provided, I had sorted them provided I had sorted them in a proper way and then with the sorted array I could have done binary search. Now, will be also doing some problem solving within this course will be doing some problem solving in the field of numbers or equations mathematical problem solving, but before that in order to do this today let us have a relook at the strings, we had looked at strings earlier as an array of characters.

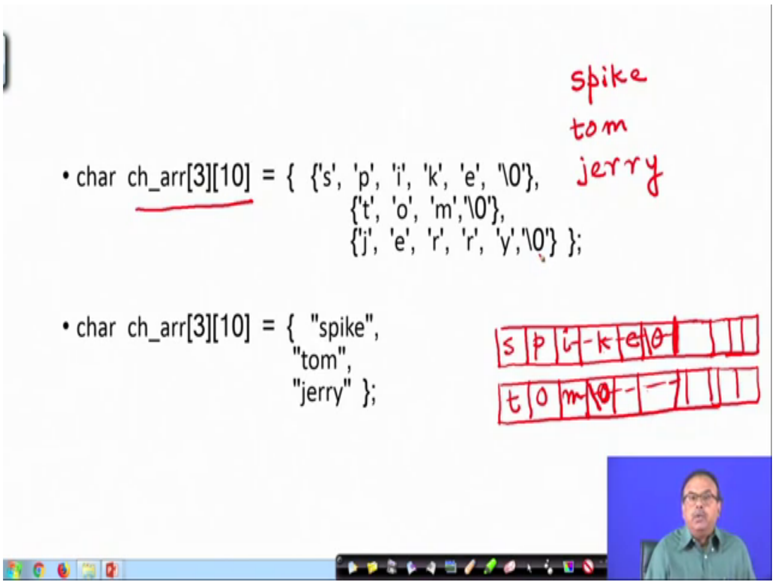
(Refer Slide Time: 18:55)



Array of Strings

So, let us have a relook at the whole thing today how can I represent an array of strings like the one that, I was showing here and array of strings all these names Atul, Amal, Gargi, and Gautam all these are nothing, but this is a string this is a string all these are different strings. So, how can I represent them let us have a look at that.

(Refer Slide Time: 19:30)



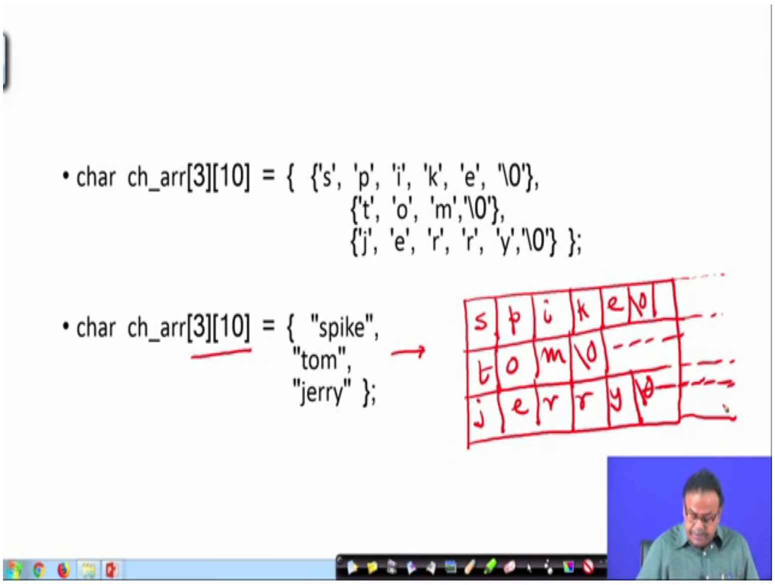
```
• char ch_arr[3][10] = { {'s', 'p', 'i', 'k', 'e', '\0'},
                         {'t', 'o', 'm','\0'},
                         {'j', 'e', 'r', 'r', 'y','\0'} };

• char ch_arr[3][10] = { "spike",
                         "tom",
                         "jerry" };
```

Say I can I have got 3 strings one is spike and there is tom and there is jerry. Now these 3 strings I have to store and I am saying that each string can be at best 10 characters at most 10 characters. And I am looking at 3 such names. So, it is an array 3 rows and each row having 10 characters right. So, the definite that definition would be I named them as char I could have done character array and it is 3 rows and 10 columns now I can initialize that as spike as character S character p.

So, s comes here, p comes here, i comes here, k comes here, e comes here and that is the end. So, it is backslash 0 I also put that and so, 1 2 3 4 5 6 4 places are still vacant 4 places are still vacant. Similarly for the next 1 is tom so, t o m back slash 0 and the rest are rest 6 are all free all right, but my string has ended here. So, 1 way I is that I can represent them in this way where, I specifically talk about the characters in the array and terminate them with backslash 0. The other way I can do it is simply I initialize them.

(Refer Slide Time: 21:35)



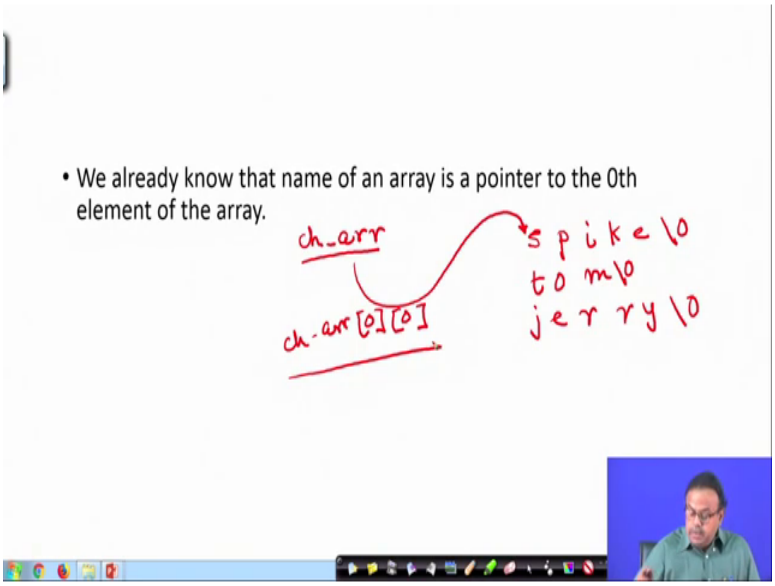
- char ch_arr[3][10] = { {'s', 'p', 'i', 'k', 'e', '\0'},
  {'t', 'o', 'm','\0'},
  {'j', 'e', 'r', 'r', 'y','\0'} };
- char ch_arr[3][10] = { "spike",
  "tom",
  "jerry" };

s	p	i	k	e	\0
t	o	m	\0		
j	e	r	r	y	\0

As 3 10 the same array with 3 strings spike comma, tom comma, jerry. Now internally even if I do that it is essentially becoming the same thing internally it is being stored in the same way with 3 rows and each row having this characters s p i k e backslash 0 and so and so forth, there can be more t o m back slash 0 and so and so forth here.

Also j e r r y and so, on I am sorry backslash 0 and so and so forth it is being stored. So, both of them are equivalent I can do it in either of these ways alright. So, a string can be represented in any of these ways.

(Refer Slide Time: 22:45)

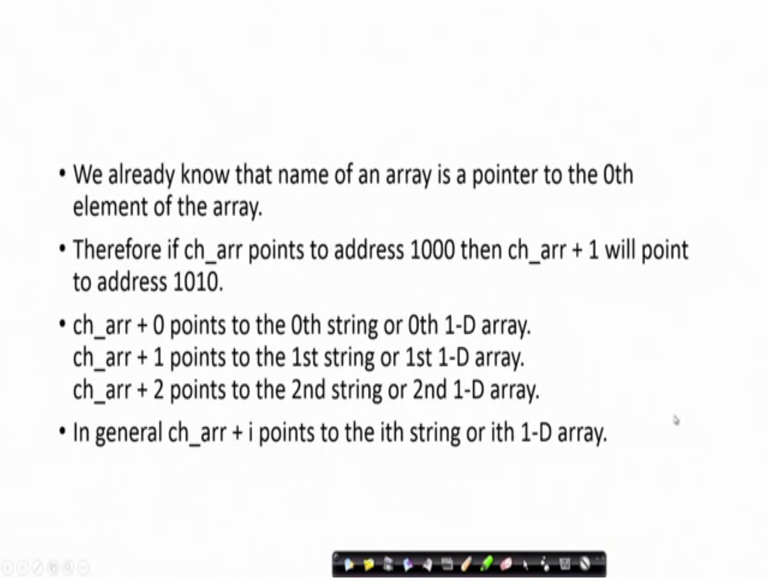


- We already know that name of an array is a pointer to the 0th element of the array.

So, we already know this again the name of the array is appointed to the 0th element of the array. So, if I have this array s p i k e backslash 0, t o m backslash 0, j e r r y backslash 0 when I am calling that charr the name itself is pointing to the first element of this array. So, ch arr 0 0 it is pointing to that.

So, this and ch arr 0 0 are the same they mean the same thing ok. So, this you just like an array this is also true here.

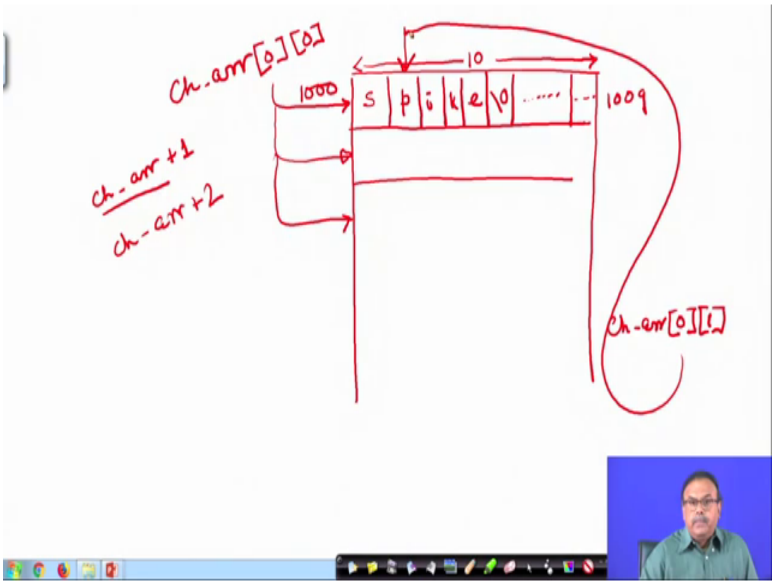
(Refer Slide Time: 23:43)



- We already know that name of an array is a pointer to the 0th element of the array.
- Therefore if ch_arr points to address 1000 then ch_arr + 1 will point to address 1010.
- ch_arr + 0 points to the 0th string or 0th 1-D array.
  ch_arr + 1 points to the 1st string or 1st 1-D array.
  ch_arr + 2 points to the 2nd string or 2nd 1-D array.
- In general ch_arr + i points to the ith string or ith 1-D array.

Therefore, if ch_arr points to the address 1000 ch_arr plus 1 will point to address 1010, why the reason is again I have got this array.

(Refer Slide Time: 24:02)

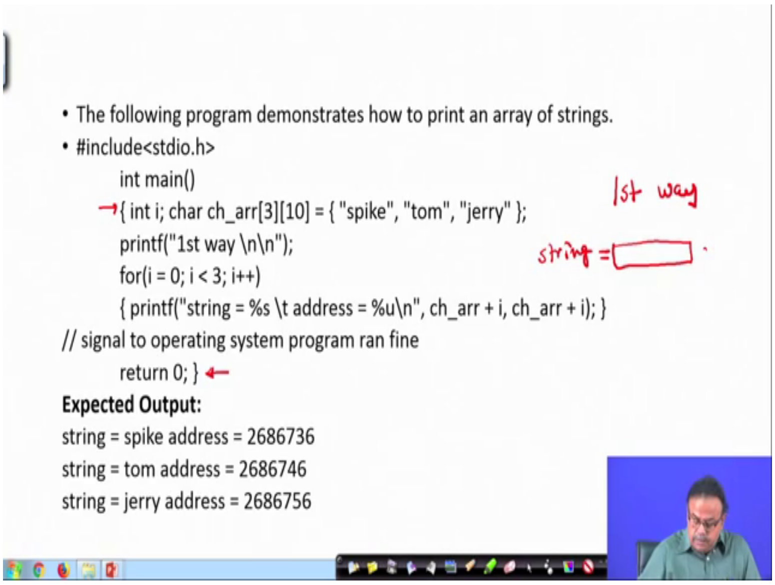


And ch_arr is pointing ch-ar arr is pointing to the 0 element of the first row, if this is 1000. Then 1002, 1009 this array this location is 1009, because the size is 10 size is 10. So, although s p i k e backslash 0 etcetera is there, then when I go to the next one ch underscore arr plus 1; that means, whatever this pointer arrays the entire thing is taken 0 to 10. So, I am taking here it is taking all these 10 elements as 1 array it will be pointing to the next one; that means, it will be 1010 ok.

Again if I make it ch-arr plus 2 it will come to this 1 all right that is how the pointer works, but how can I get this character this will be ch-arr 0 1 that is pointing to this element ok. Now this thing must be clearly understood. So, ch-arr points 0 th string or the 0 th 1 dimensional array ch-arr 1 points to the first string or the first 1 dimensional array I mean number 1 0 at first ch-arr 2 points the second 1 dimensional array.

So, each of them are 1 dimensional arrays and ch ar is pointing to each of these 1 dimensional arrays ch-arr is pointing to 1 one dimensional array. And as I increment the pointer this pointer is going on increasing alright in general ch-arr I points to the I th string of the yet 1 dimensional array clear.

(Refer Slide Time: 26:42)



- The following program demonstrates how to print an array of strings.
- #include<stdio.h>

```
        int main()
        { int i; char ch_arr[3][10] = { "spike", "tom", "jerry" };
        printf("1st way \n\n");
        for(i = 0; i < 3; i++)
        { printf("string = %s \t address = %u\n", ch_arr + i, ch_arr + i); }
// signal to operating system program ran fine
        return 0; }
```

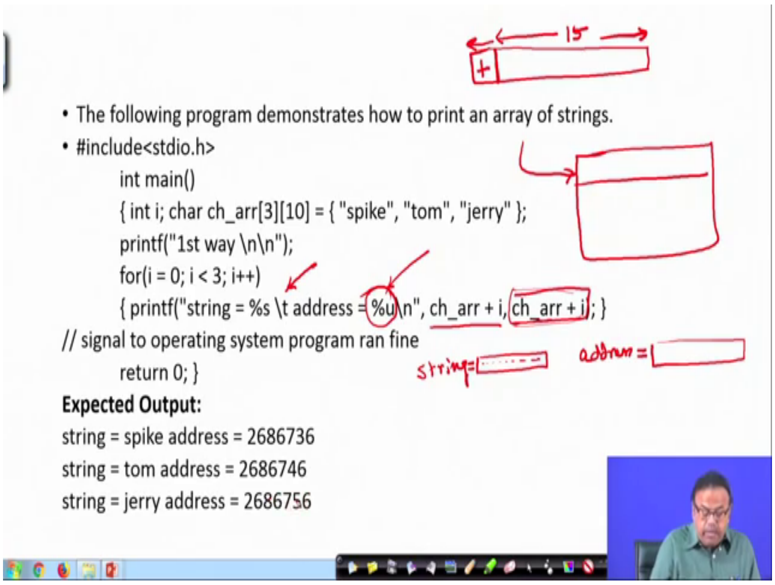
Expected Output:

```
string = spike address = 2686736
string = tom address = 2686746
string = jerry address = 2686756
```

The following program let us look at the program and the following program demonstrates how to print an array of strings? Please take 1 minute to read this array a program yourself and try to understand. It you can see here that I am starting with the main function and although it is written in a 1 in a very cryptic way. So, my program starts from this point and ends at this point, i is an integer ch-arr is again an array of 3 strings and each can be of maximum length 10. So, spike tom and jerry are there.

So, I can print I just print first way next line I go to next line and to next line. So, I just put first way and then what do I do. So, I will print first way and then 2 lines have been given blank, then for i 0 2 less than 3 i plus plus printf string, I am printing the string. So, what will be printed string equals a placeholder for the string and then there will be some tab 1 space and then.

(Refer Slide Time: 28:34)



- The following program demonstrates how to print an array of strings.
- #include<stdio.h>

```
        int main()
        { int i; char ch_arr[3][10] = { "spike", "tom", "jerry" };
        printf("1st way \n\n");
        for(i = 0; i < 3; i++)
        { printf("string = %s \t address = %u\n", ch_arr + i, ch_arr + i); }
// signal to operating system program ran fine
        return 0; }
```

Expected Output:

```
string = spike address = 2686736
string = tom address = 2686746
string = jerry address = 2686756
```

So, the first will be string here some space and then tab this is tab.

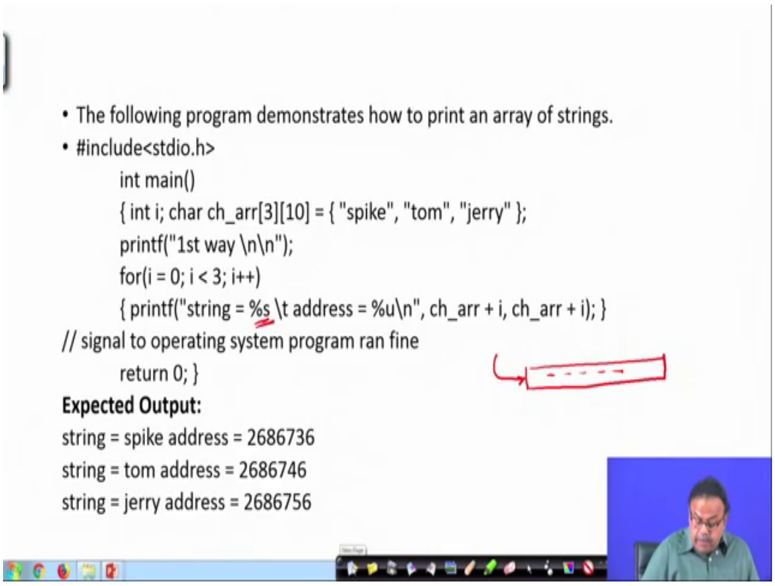
So, I give some space and write address and then sorry string equal to string equal to address equal to and there should be some space for the address and that format is unsigned integer. So, it will become longer alright.

Because unsigned and signed integer a whenever I am putting suppose I have got 16 bits for representing an integer. If it be assigned if I distinguish between plus 5 and minus 5 and keep 1 bit kept for the plus or minus then for the 16 bit I will be actually having 15 bits for this size, but if I make it unsigned then I can have all these 16 bits. So, I can represent more numbers all right more numbers.

So, that apart so, string and address and then what am I printing here care I care plus i; that means, 0 care 0 the first string and then I am again putting the care this, now what is the difference between this care this and this is being printed in percentage arrays; that means, the string will be printed and here in an unsigned integer what I am printing here is the address of that string.

So, this string spike was here when I print this just the name of the array; that means, I am in this format if I do I am printing the address of it therefore, the output will be somehow like this ok. So, this is something to be understood very clearly that this array when I am actually looking at this I am taking the content of the array otherwise I am taking the array itself.

(Refer Slide Time: 30:36)



- The following program demonstrates how to print an array of strings.
- #include<stdio.h>

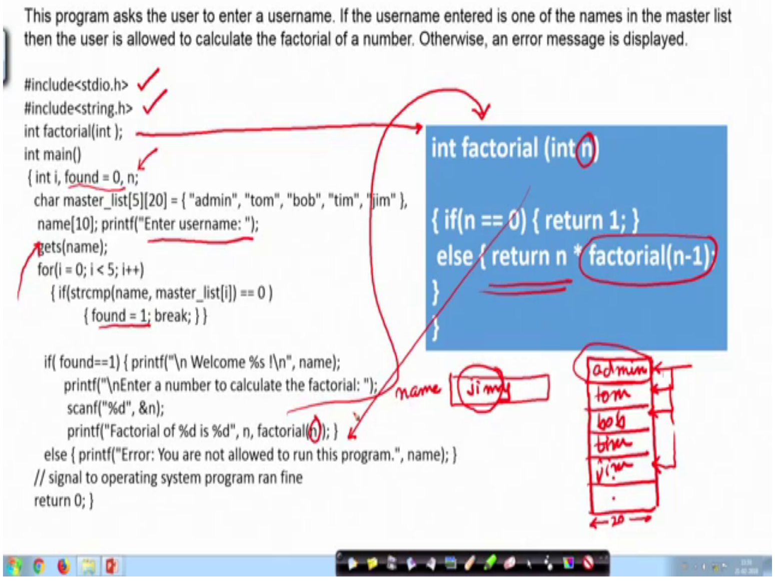
```
        int main()
        { int i; char ch_arr[3][10] = { "spike", "tom", "jerry" };
        printf("1st way \n\n");
        for(i = 0; i < 3; i++)
        { printf("string = %s \t address = %u\n", ch_arr + i, ch_arr + i); }
// signal to operating system program ran fine
        return 0; }
```

Expected Output:

```
string = spike address = 2686736
string = tom address = 2686746
string = jerry address = 2686756
```

So, here is a program.

(Refer Slide Time: 30:53)



This program asks the user to enter a username. If the username entered is one of the names in the master list then the user is allowed to calculate the factorial of a number. Otherwise, an error message is displayed.

```
#include<stdio.h>
#include<string.h>
int factorial(int );
int main()
{ int i, found = 0, n;
  char master_list[5][20] = { "admin", "tom", "bob", "tim", "jim" },
   name[10]; printf("Enter username: ");
   gets(name);
   for(i = 0; i < 5; i++)
     { if(strcmp(name, master_list[i]) == 0 )
          { found = 1; break; } }

    if( found==1) { printf("\n Welcome %s !\n", name);
       printf("\nEnter a number to calculate the factorial: ");
       scanf("%d", &n);
       printf("Factorial of %d is %d", n, factorial(n)); }
    else { printf("Error: You are not allowed to run this program.", name); }
  // signal to operating system program ran fine
  return 0; }
```

```
int factorial (int n)

{ if(n == 0) { return 1; }
 else { return n * factorial(n-1);
}
}
```

This program asks the user to enter a username if the username entered is one of the names in the master list then the user is allowed to calculate the factorial of a number otherwise an error message is displayed.

So, we have got sorry a master list of names now if you are a valid user your name is there in the list. So, first what is doing it is including stdio dot h it is including string dot h; that means, some string functions are also included I am defining the factorial prototype the factorial function has been written later. Suppose this part this part is later than the main function ok.

Now, the main is here and what is the main doing it has got some integer I and found is 0 initially the name is not found here and some n number for which I have to compute the factorial, master list 5 by 20; that means, each of these each of these can be 20 character long and there are 5 search. And I put the valid names as admin tom, bob, tim, and jim.

Now I am asking a name is so, I am and I am asking from the user enter user name now the user gives the name. So, get s here you are getting a new function being introduced here that is get s what is get s get care get ch, you know and here the get s means get the string. So, in 1 short it is getting the string as the name.

So, name I am getting from the user some name say it is jim suppose the user enters jim. Now for I equals to 0 to 5 why 5 there are all the 5 names are here if string compare name and master list I, if this character this string and this string matches then found is 1, otherwise I am again going in the fall loop.

So, at every stage of the loop I am trying to compare this name with this string here there is no match. So, I come here no match no match in that way I come and find jim. So, here ultimately I will get found and then I will break. Now when I come out suppose it was not jim, it was say jimmy alright then I do not get a match here. So, I can come out with found equal to 0 or found equal to 1, he found equal to 1 then welcome then welcome exclamation and then the name it is printing welcome jim.

And asking please enter the number is entering the number for the factorial and then here you see in this printf I have called the factorial, that factorial of the number is this n is this so, I call factorial and this part you know. Now that this n and this n and may not be the same I mean they are two different locations, but they same it could have been x also in that case internally had it had to be x here.

So, this is being passed over here and the factorial is being computed all right. Once the factorial is computed, now how the factorial is computed I will be discussing that later, because here there is some trick forget about that trick now I will be discussing that later that is recursion, which you have not yet discussed and after this I am coming back here and I am printing.

So, the purpose of this program was to show how we can handle strings. So, these are very common very common thing I mean you try to login in a machine right. And you type in your login name and your password, if they do not match the system says login failure now how does it do it internally.

Internally it has got a list of just like this it has got a list of all the users and it takes your username and checks it with all these. That is a problem that is being solved a sub problem that is being solved in the operating system and that is the basic principle is depicted in this program alright we will continue with other problem solving approaches in the future lectures.