

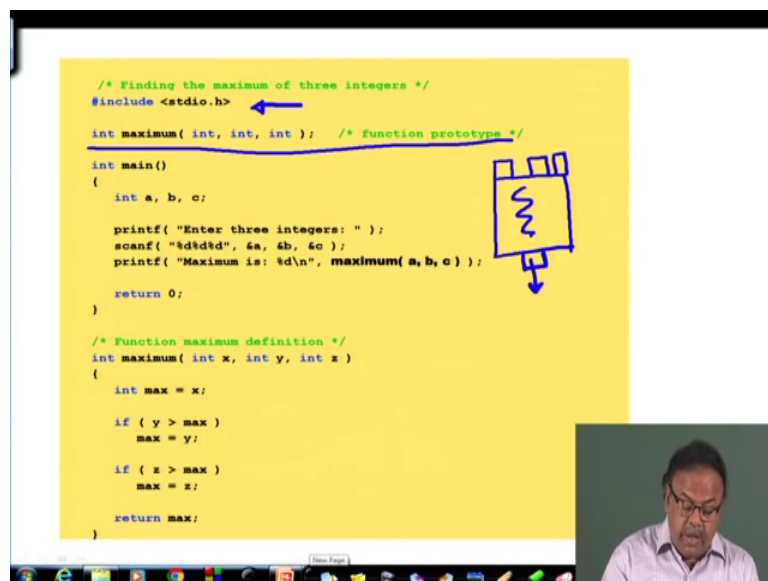
**Problem Solving Through Programming in C**  
**Prof. Anupam Basu**  
**Department of Computer Science & Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 41**  
**Substitution of # include and Macro**

We have discussed about passing arrays as parameters. And for passing arrays as parameters in C to, C functions we take the course to call by reference. Otherwise for all other parameter passing we take the course to call by value. And just to summarize the call by value and call by reference. Call by value we copy the value of the actual parameter to the formal parameter; formal argument and in call by reference we do not copy the value, but we just pass on the address of the pointer and reference to the actual parameter ok.

So, as a result; what happens? That if any change to that variable is resulted in the function itself, in the case of call by value that will not be automatically reflected to the main program, but in the case of call by reference since there is no other separate copy that will automatically be reflected. Just to conclude that part we will see a number of applications of functions just to come to conclude that.

(Refer Slide Time: 01:44)



```
/* Finding the maximum of three integers */
#include <stdio.h>
int maximum( int, int, int ); /* function prototype */
int main()
{
    int a, b, c;

    printf( "Enter three integers: " );
    scanf( "%d%d%d", &a, &b, &c );
    printf( "Maximum is: %d\n", maximum( a, b, c ) );

    return 0;
}

/* Function maximum definition */
int maximum( int x, int y, int z )
{
    int max = x;

    if ( y > max )
        max = y;

    if ( z > max )
        max = z;

    return max;
}
```

Let us have a quick look at 1 function, but there is no idea, but just vision of the prototypes that you can see here, finding the maximum of 3 integers.

So, we have the header file, here include studio dot h. Then here we declared a function prototype. The actual function maximum has been written later. So, here you see that what does this line tell us? This line tells us that maximum is a function and it consists of 3 integer arguments, or 3 integer parameters will come. So, as if this is a function which has got 3 input doors and each door is wide enough to accommodate an integer ok. So, that much is told. And whatever is inside this I am not aware of that, and at this stage I am not aware of that. And the output will also be another integer that is told by this.

Then so, here the salient point to see is that here I have just it was sufficient to just specify the types, no names have been put in. Now comes the main program that is very simple a, b, c are 3 integers.

(Refer Slide Time: 03:19)

```
/* Finding the maximum of three integers */
#include <stdio.h>

int maximum( int, int, int ); /* function prototype */

int main()
{
    int a, b, c;

    printf( "Enter three integers: " );
    scanf( "%d%d%d", &a, &b, &c );
    printf( "Maximum is: %d\n", maximum( a, b, c ) );

    return 0;
}

/* Function maximum definition */
int maximum( int x, int y, int z )
{
    int max = x;

    if ( y > max )
        max = y;

    if ( z > max )
        max = z;

    return max;
}
```

So, they are local to this function, enter the integers abc are interred here. So, you know; what a is? what b is, and what c is, and then you are calling inside c. Here inside the print f I can call maximum with the parameters a, b, c and we come here. And here I have defined here I did not name here I did not name the variables; here I have defined the names of the variables int x, int y, int z. So, this x, y, z again property of this maximum alright.

So, xyz are here. And the algorithm of the logic is simple. Initially I make max to be x; if y is greater than max and y is the max, otherwise z is the max and then I am returning max. I am returning max means I am coming here, and then that max is being printed.

And returns 0 now my main function, where is my main? Main is here, main has got a type int. So, I am returning 0. So, int will if at the end of, the main function a 0 is returned, I will assume that the program has successfully completed ok. The reason I brought up this example is just to illustrate this case that you need not at the prototype level you can escape specifying the parameters.

(Refer Slide Time: 05:06)

```
/* Finding the maximum of three integers */
#include <stdio.h>

int maximum( int, int, int ); /* function prototype */

int main()
{
    int a, b, c;

    printf( "Enter three integers: " );
    scanf( "%d%d%d", &a, &b, &c );
    printf( "Maximum is: %d\n", maximum( a, b, c ) );

    return 0;
}

/* Function maximum definition */
int maximum( int x, int y, int z )
{
    int max = x;

    if ( y > max )
        max = y;

    if ( z > max )
        max = z;

    return max;
}
```

Prototype Declaration

Function Calling

Function Definition

44

So, here is a prototype declaration, function calling, and function definition you know that.

(Refer Slide Time: 05:11)

### Calling Functions: Call by Value and Call by Reference

- Used when invoking functions
- Call by value
  - Copy of argument passed to function
  - Changes in function do not effect original
  - Use when function does not need to modify argument
    - Avoids accidental changes
- Call by reference
  - Passes original argument
  - Changes in function effect original
  - Only used with trusted functions
- For now, we focus on call by value

45

So, this are already repeated I am not going into that.

(Refer Slide Time: 05:18)

### An Example: Random Number Generation

- rand function
  - Prototype defined in <stdlib.h>
  - Returns "random" number between 0 and RAND\_MAX (at least 32767)  
`i = rand();`
  - Pseudorandom
    - Preset sequence of "random" numbers
    - Same sequence for every function call
- Scaling
  - To get a random number between 1 and n  
`1 + ( rand() % n )`
    - `rand % n` returns a number between 0 and n-1
    - Add 1 to make random number between 1 and n  
`1 + ( rand() % 6) // number between 1 and 6`

46

(Refer Slide Time: 05:23)

```
1 /* A programming example
2 Randomizing die-rolling program */
3 #include <stdlib.h>
4 #include <stdio.h>
5
6 int main()
7 {
8     int i;
9     unsigned seed;
10
11     printf( "Enter seed: " );
12     scanf( "%u", &seed );
13     srand( seed );
14
15     for ( i = 1; i <= 10; i++ ) {
16         printf( "%10d ", 1 + ( rand() % 6 ) );
17
18         if ( i % 5 == 0 )
19             printf( "\n" );
20     }
21
22     return 0;
23 }
```

### Algorithm

1. Initialize seed
2. Input value for seed
  - 2.1 Use `srand` to change random sequence
  - 2.2 Define Loop
3. Generate and output random numbers

48

So, instead I will be talking about a new thing here, that is hash include.

(Refer Slide Time: 05:26)

## #include: Revisited

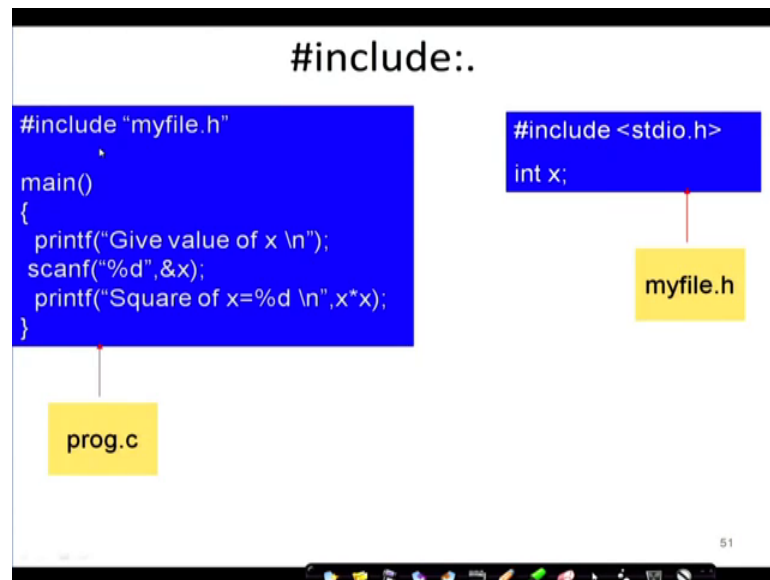
- Preprocessor statement in the following form  
`#include "filename"`
- Filename could be specified with complete path.  
`#include "/usr/home/rajan/myfile.h"`
- The content of the corresponding file will be included in the present file before compilation and the compiler will compile thereafter considering the content as it is.

50

We have talked about hash include, we have used hash include. And we mentioned that hash include is nothing but a preprocessor statement. A preprocessor statement how? Say hash includes filename. We have seen hash include studio dot h, but it can be for that matter any file name, hash include file name.

Say for example I can specify the full path, say slash usr in a Linux environment slash home rajan my file dot h. I want to include this file; I want to include this file in mind for serving. This file in my program, all right; the content of the corresponding file will be included in the present file, before compilation is done before compilation is done. And the compiler will compile there after considering the content as it is. So, it is a preprocessor statement. That is, I put that inside that hash include that file is included and then the compilation is done ok.

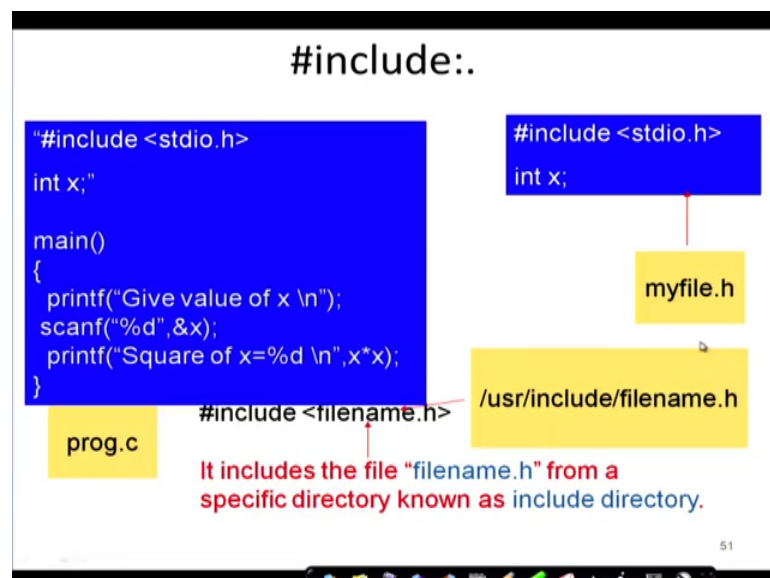
(Refer Slide Time: 07:01)



Now, let us look at this hash include example. Look at this program segment which is my source program, which is known as prog dot c. So, this is my source program. This source program has got hash include no stdio dot h, but it has just said that in hash include my file dot h, and then a main program follows ok. Now, this my file dot h is nothing but another file, which is which has got hash include stdio dot h and index.

So, when I include my file dot h, then this particular file will be included this particular file that is this segment will be included here.

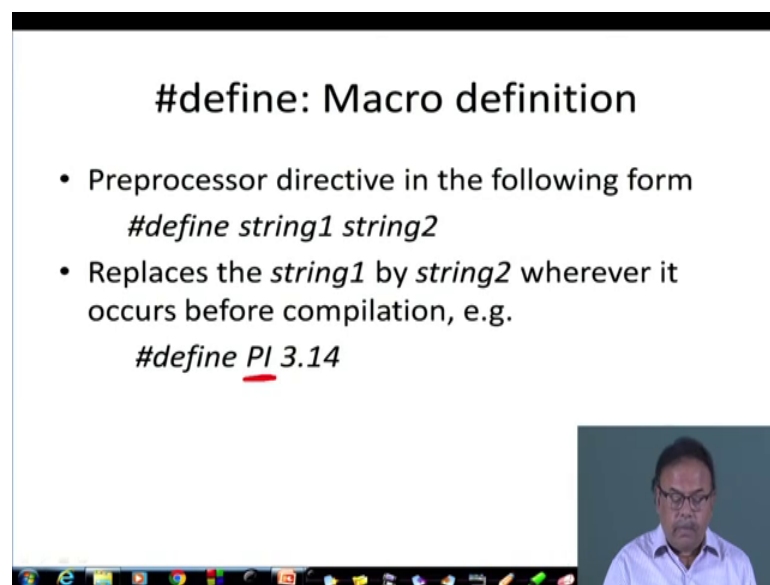
(Refer Slide Time: 07:59)



So, consequently what will happen is when I say in general hash include file name dot h it includes a file name file, file name dot h from a specific directory which is known as the include directory. When I say hash include it takes from the include directory, and in the include directory I have kept my file dot h, and consequently when that is included the ultimately the thing that looks like is, it will be it will look like this.

Because that hash include I am sorry, there should not be an inverted comma here, it should be just hash include stdio dot h followed by intex. So, this whole thing whole thing has been has replaced my file dot h. And where was my file dot h? My file dot h was under user usr include file name dot h. So, that is how the include I can have other files included in my program. Now another point of critical discussion a is macro definition. We have seen the macros like hash define.

(Refer Slide Time: 09:06)



**#define: Macro definition**

- Preprocessor directive in the following form  
*#define string1 string2*
- Replaces the *string1* by *string2* wherever it occurs before compilation, e.g.  
*#define PI 3.14*

The slide also features a small video inset of a man in the bottom right corner and a Windows taskbar at the bottom.

So, let us so, that is again a preprocessor directive. We have seen that hash include is a preprocessor command; that means, even before the compilation is d1; that is, that is copied there on the other hand, you see the preprocessor directive of hash defined string 1 string 2; that means, string 1 should be replaced by string 2. So, how does it happen? It replaces string 1 by string to whenever it occurs, where ever it occurs, where ever it occurs before compilation.

So, for example, hash defined pi to be 3.14. So, wherever it will find this pi, wherever it will find this pi, it will replace that with 3.14.

(Refer Slide Time: 10:19)

### #define: Macro definition

```
#include <stdio.h>
#define PI 3.14
main()
{
    float r=4.0,area;
    area=PI*r*r;
}
```

```
#include <stdio.h>
main()
{
    float r=4.0,area;
    area=3.14*r*r;
}
```

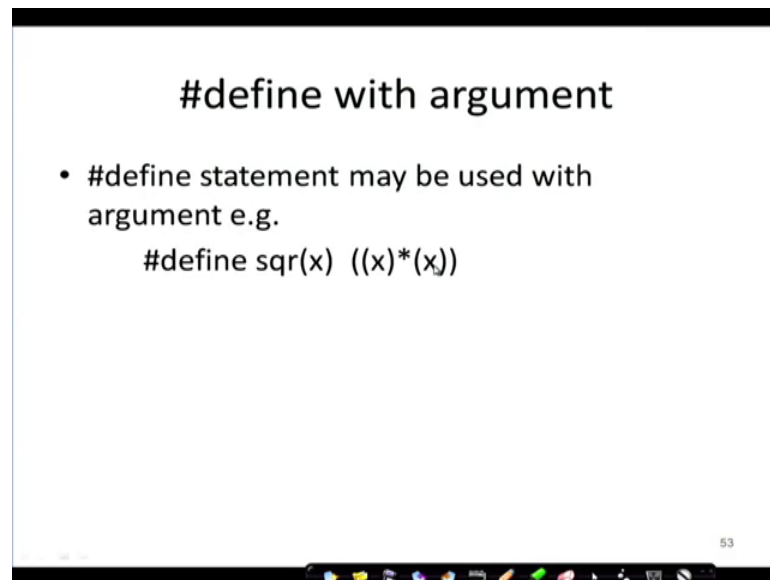
52

So, here for example, you see, we have got hash include stdio dot h, then we define pi to be 3.14 as the hash define preprocessor command. Now in my main program I have got float r 4-time 4; r is 4, and area is another variable both are float. Now area is pi times r times r, r is already initialized. So, what will happen is this will be translated to this pi will be replaced. So, the body will be float r assigned 4 and area.

And area is so, r is 4, and area will be 3.14 times r times r. So, before come even before compilation, this pi is being replaced by 3 of the value 3.14 as has been stated in the hash defined command.



(Refer Slide Time: 11:36)



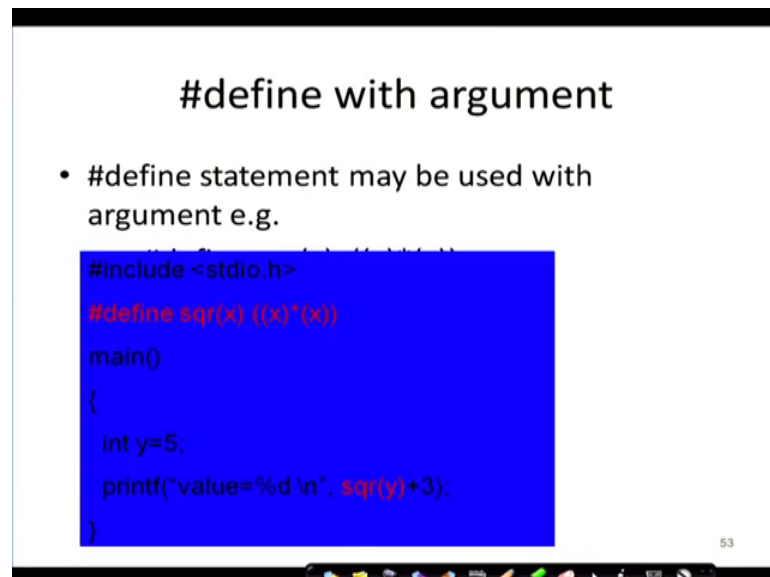
**#define with argument**

- #define statement may be used with argument e.g.  
`#define sqr(x) ((x)*(x))`

53

We can also give hash define with an argument. Till now we have seen hash define pi as a constant hash define some constant k to be 5 or whatever. Here we can also define some functions. Like you see hash define square x is x times x. So, wherever square x will appear, that will be replaced by x times x ok.

(Refer Slide Time: 12:17)



**#define with argument**

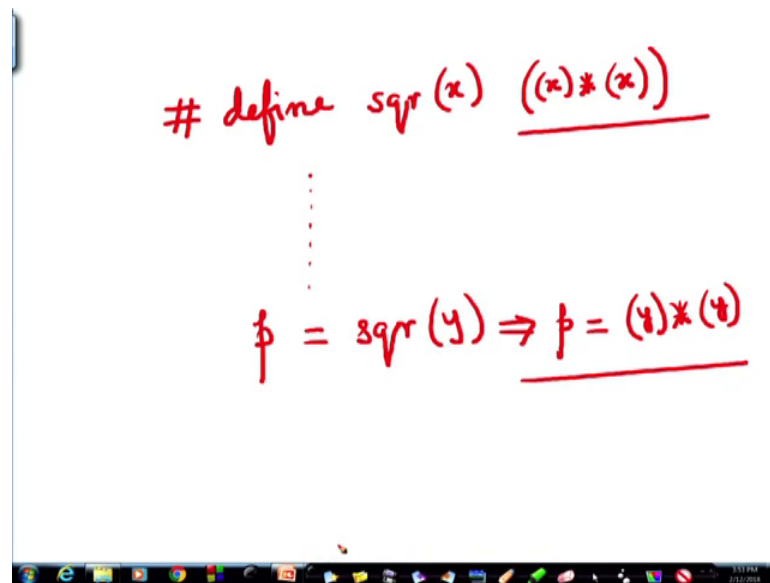
- #define statement may be used with argument e.g.

```
#include <stdio.h>
#define sqr(x) ((x)*(x))
main()
{
    int y=5;
    printf("value=%d \n", sqr(y)+3);
}
```

53

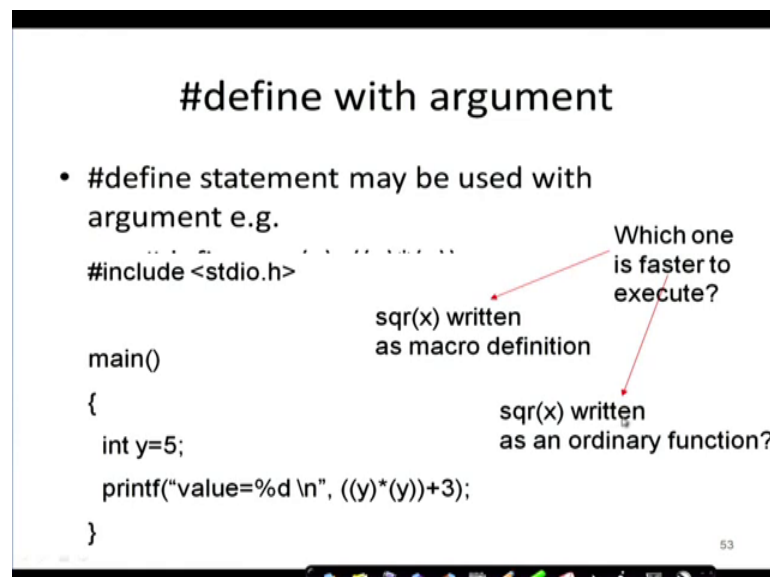
Say for example, here let us look at this program again. I think you are not being able to read this. So, let me go back say let me have if I have, declared say, hash define square x is x times x.

(Refer Slide Time: 12:46)



Then in my program, wherever I will get say, p is assigned square y. So, this will be translated as p assigned y times y ok. So, this; what should be d1 that has already been told here that square x is this. So, this will simply replace it ok. So, that is the purpose of hash define here, we can see that and so, let us look at this example again.

(Refer Slide Time: 14:01)



Main y is 5, print f value is y times y; that means, this is being repeat this is what has been generated after the square x has been square here it was square x plus 3. So, that has become y times y plus 3. Now, which 1 is faster to execute? You can think of I can if I

write square x written as a macro definition, then I am pasting as if I am pasting the body of the code inside this in the main program replacing that. So, I do not need to call any function. I can write it as a macro definition and I as a result even before compilation.

Even before compilation, I can get this look. And so, this can be straightaway applied without requiring to call a function. Now if square x is written as an ordinary function, not like a macro definition, in that case I have to call a function, and calling a function has got some overheads which I am not discussing here. So, if it be a very simple thing like this it is better to make it a macro. So, that it automatically gets pasted and it becomes faster.

(Refer Slide Time: 15:40)

**#define with arguments: A Caution**

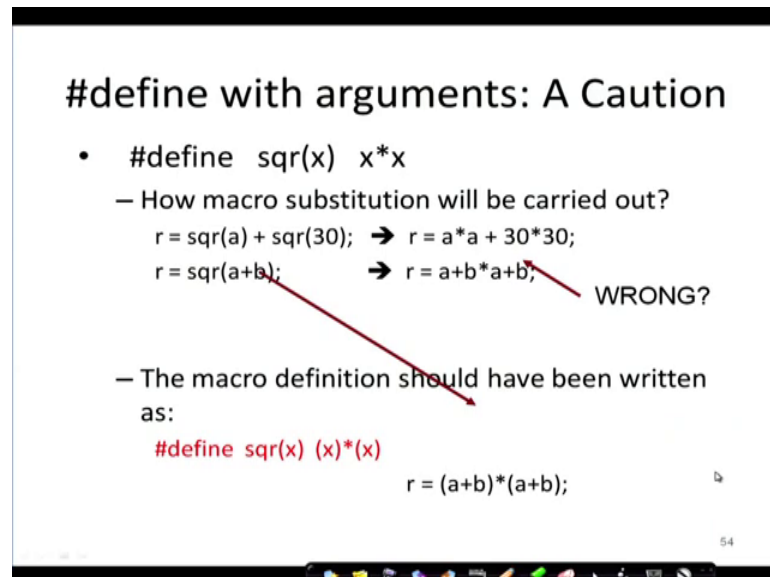
- `#define sqr(x) (x)*(x)`
  - How macro substitution will be carried out?
    - `r = sqr(a) + sqr(30);` → `r = a*a + 30*30;`
    - `r = sqr(a+b);` → `r = (a+b)*(a+b);`

However, there is a word of caution for example, if I had defined square x, now it is now when I am defining as a macro it is I have to ensure that it is correct. Therefore, if I define square x as x times x. Now, how the macro definition substitution will be carried out? Let us look at this. If there be something like r assigned square of a plus square of 30.

So, it will be a times a plus 30 times 30. That is how it will happen ok, now what about r square of a plus b? It will be simply pasted as a plus b times a plus b. Consequently, the result will be wrong, because in the during execution b times a will be d1 first, and then that will be added with a and with b. So, this is not what I intended. Therefore, if I had d1 this, define this as like this, and like this, this problem would not have occurred.

So, unless I do that this a plus b should will not be I need to take if I define it like this then it becomes this which is the correct 1. But unless I do that it will lead to a wrong result.

(Refer Slide Time: 17:16)



**#define with arguments: A Caution**

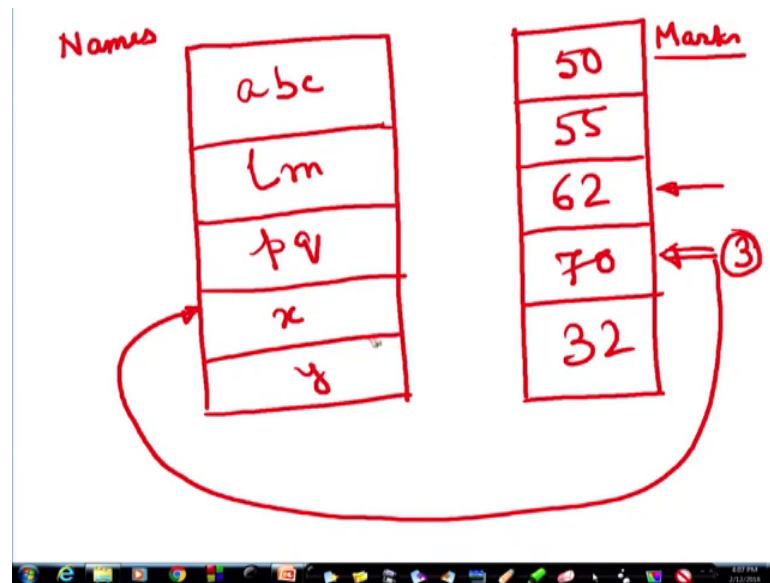
- `#define sqr(x) x*x`
  - How macro substitution will be carried out?  
`r = sqr(a) + sqr(30);` → `r = a*a + 30*30;`  
`r = sqr(a+b);` → `r = a+b*a+b;` **WRONG?**
  - The macro definition should have been written as:  
`#define sqr(x) (x)*(x)`  
`r = (a+b)*(a+b);`

So, you must be very careful about it, this is wrong. And the macro definition should have been written as I have shown square x is x times x, in that case we would have got the correct result. Now before moving to anything else, we will now look at some applications of what we have learned. We are now at a position where we have learnt all the fundamental tools, without the sophistications, that are required to write a program ok.

So, suppose I want to solve a problem like this. That, I have got 5 students in the class; and the 5 students have got 5 names of course, and each of them have got some marks, total marks, right. Now how do I represent that? And I want to find out which student I want to print the name of the student who has got the highest marks, all right? Is the problem clear let me repeat the problem, I have got 5 students in a class? And I want to find out which student has got the highest marks. And I want to know the name of that student ok.

How can I solve this problem? This is a simple problem. Here what I will need is, first I will need an array of the names of the students. So, first I need an array of the names of the students.

(Refer Slide Time: 19:13)




So, here 5 names will be stored very, very disproportionate array. And let me call this array to be name or names. And on this side, I have got another array, which holds the marks order of the different students. Corresponding to the roll number maybe 1, 2, 3, 4, 5 the marks are stored here. Somebody got 50, somebody got 55, somebody got 62, somebody 70, somebody 32. Say, and their names are a, b, c, l m p q x and y; suppose, these are names, these are the different names. Now why do I need to add is, these are this; the array called marks. Why do I need 2? 2 arrays, first of all the arrays should be holding same type of data.

So, what is the type of data that the array names are holding? It is a array of character or character string. Now how can I represent these names? I can represent them as an array of an array of strings, or each of these names is what each of these names is a character array. Therefore, if I just look at names, names will look like a 2-dimensional array. Now what would be my algorithm first of all come to the representation.

Later, what would my; be my algorithm? First of all, I will have to go to this mark array which is simpler it is an array of integers, assuming all integer marks are given. Then I will have to carry out find out the maximum of this. And you can write a function to find the maximum of an array, right. So, let us try to write the function first so, I have got an array marks; which has got 5 elements, all right.

(Refer Slide Time: 22:22)

```
int maximum(int M[]),
{
    int i = 0; maxindex,
    int max = M[0];
    for (i = 1; i <= 4; i++)
    {
        if (M[i] > max
            max = M[i];
            maxindex = i;
        }
    return (maxindex);
}
```



So, I can call a function max, what let us first of all think of this maximum value what will it return me? Say, it we will find out say if I come to this, it will find out which 1 is the maximum, and will return me an index, right or this is not the maximum sorry this 1 is the maximum.

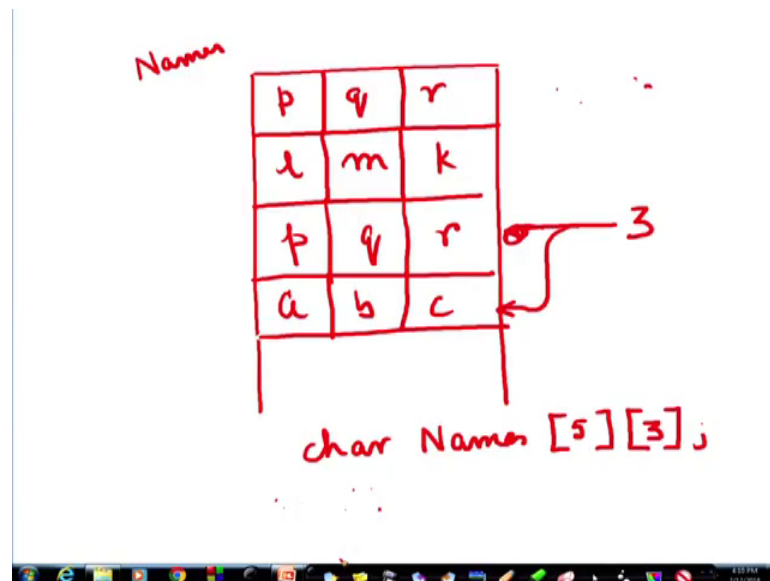
So, it will return me an index 0 1 2 3 so, it will return me an integer. Therefore, I can start with like this that int maximum of I can call an array, integer array int suppose that is m, this and the main function will call it with marks. And inside this what will happen? You know, initially let me call a value int this is a local max assigned m 0. So, I initialize a variable max with m 0. Now for I assign some index i.

So, I have to declare I also here. I assigned 1 i less than equal to 5 less than equal to 4, sorry, less than equal to 4, i plus

. Here if  $m_i$  is greater than max, then if I then max is  $m_i$ . And that will go on. Now ultimately what shall I turned from this? Should I return the max, no, I am returning the value of i. Because, where I found the maximum now? So, for that now this I is changing. So, here I will also have to make say. So, let me just make this I to be 0, let me make this i to be initially i is 0. So, I could have d1 away with this. And wherever I get this max I am changing this i, whenever I am getting this max I can say that there is another max index. And I am making max index equals i. So, I am remembering where I found the max and I will be returning the max index, all right.

So, my maximum function in this way, what it will do is it will sorry it will find out the maximum so, it will return 3. So, the first thing is find max. Now I have got an array names, how does that represent it? The representation can be as strings or it can be as an array of characters. If I consider this to be an array of characters, then remember that it is a 2-dimensional array ok. And I will in that case I will take. So, this 3 and I will come to this 3 this, and I will print out this particular element from that array ok. So now, what about these names let us see. Names is an address a p, q, r or something.

(Refer Slide Time: 27:59)



So, I can have this as an array of characters. So, p q r l m whatever it is, and this part is blank. So, what is my declaration of names? Names will be there are 5 names, 5 rows and each row is a character.

So, it is a character array char names, let me write it here. Char names 5 rows, and each row may have 3 columns, right. At best like this lm p lm k pq r like that say. Now I have suppose come with that maximum index was this rows so, that is 3. Now how do I print this pq r name here, if it is kept is a string I could have straight with percentage s format. But here how would I do with this here I can, I will have I will print which row I will print. I am not writing the entire code ok. Print f which row would I print I had print the names 3, but there are 3 characters. So, names 3 in a loop I have to print right. So, print f so, how would the printing be d1? How do I print? 1 rows of an array a 2 dimensional so,

here I come. So, I will print in a loop names 3 0 3 1 3 2 ok. Actually, this is 3 should be somewhere here, it is confusing 3 should be somewhere here say a, b, c.

So, 3 is actually this row. So, names I will be printing 3 0 3 1 3 2 and that I can print in a for loop. So, I leave it to you to try to print this in the character format. You can also try it in the string format. So, what we have d1 here is, we have seen the how the macros are replace replaced, and we will see further examples of functions all through in the future lectures, where we will be applying functions and arrays consequently.