C PROGRAMMING LECTURE

METHODS

Deepak Majeti
M-Tech CSE
IIT-Kanpur
mdeepak@iitk.ac.in

Some Essentials

- **Scope** of a variable:

  - The areas in the code where the variable is visible.

- **Life** of a variable:

  - Areas of the code where the variable exists in memory.

Example

```
for( int i =0; I < 100 ; i++ ){
        \\scope of i
}
can't see i here.
**************************************************
int main(){
        int i = 0;
}// i dies after the main program ends.
```

Method

- Definition: A block of code which performs some well defined computational task.
  - a method may have some input
  - a method may also have some output

Example: *int* add(*int* , *int*)

Can visualize as a mathematical function which takes a value and returns another value.

Why Methods

- Huge codes are difficult to understand and debug.
- Makes the code look simpler, neater and makes your task easier.
- Re-usability of code.

Example:

Imagine a program in which we need to swap two numbers often. Life becomes easy if we could swap in a single step.

printf(),scanf() are functions too. They make our work so easy.

Some terminologies

- Function Declaration/ Prototype:

```
int func(int , int);
```

- Function Definition:

```
int func ( int a, int b){
        printf("Welcome to func");
return (a + b);
}
```

Some more

- The value passed to a method is called its

“argument”

- The variable which receives the arguments is called

“parameter”

- Parameters are declared inside the parenthesis and we must declare the type of the parameter.

- Argument may be an expression but,

type (argument) = *type* (parameter)

Typical Example

```
#include <stdio.h>
void add_print(int , int); //function declaration
int main(){
    int a=4;
    int b=5;
    printf("Entering 'add_print' function\n");
    add_print(a,b);
    printf("Just came from 'add_print' function\n");
return 0;
}
//function definition
void add_print(int val1,int val2){
    int c;
    printf("The two values entered are:%d,%d \n",val1,val2);
    c=val1+val2;
    printf("Sum of numbers entered is:%d \n",c);
}
//parameters, arguments ?????????
```

Contd.

- In the above example,
  - **‘a’**, **‘b’** are arguments to the function.
  - **‘val1’**, **‘val2’** are the parameters of the function.

- Scope and Life
  - **‘a’**, **‘b’**
    - Have scope limited to the main function.
    - Their life is till the main exits.
  - **‘val1’**, **‘val2’**
    - Have scope limited to the function block.
    - Their life is till the function call is over.

Another Example

```
#include <stdio.h>
int mult_return(int , int); //function declaration
int main(){
    int a=4;
    int b=5;
    int c;
    printf("Entering 'mult_return' function\n");
    c = mult_return(a+4,b*2);  // sending an expression
    printf("Just came from 'mult_return' function. The value is %d \n", c);
return 0;
}
//function definition
int mult_return(int val1,int val2){
    int c;              // "c" again????
    printf("The two values entered are:%d,%d \n",val1,val2);
    c=val1*val2;
    return c;
}
```

Another Example

```
#include <stdio.h>
void swap(int , int); //function declaration
int main(){
    int a=4;
    int b=5;
    swap(a,b);
    printf("The value of 'a' is %d and the value of 'b' is %d\n", a,b);
return 0;
} // the values of a and b did not swap ☹.
//function definition
void swap(int val1,int val2){
    int temp;
    printf("The two values entered are:%d,%d \n",val1,val2);
    temp=val1;
    val1=val2;
    val2=temp;
}
```

Some problems

- Till now the functions took the ***values of the arguments*** from the calling function (main).
- What if we need to change the values of variables in the calling function?
- How do we get access to the calling function's data?

Simple!! Send the **address** of the variable

Another Example

```
#include <stdio.h>
int swap(int *, int *); //function declaration
int main(){
    int a=4;
    int b=5;
    swap(&a,&b);
    printf("The value of 'a' is %d and the value of 'b' is %d \n", a,b);
return 0;
} // the values of a and b did swap ☺.
//function definition
int swap(int *val1,int *val2){
    int temp;
    printf("The two values entered are:%d,%d \n",*val1,*val2);
    temp=*val1;
    *val1=*val2;
    *val2=temp;
}
```

Preprocessing

- We can write methods, declare variable in multiple files.
  - Need to link these.
- **# *include<filename>*** includes the file *filename.*
- ***# define ABC(X) X*X***
  - Replaces the occurrences of ABC(z) with z*z.
  - What happens to ABC(z+1)? // try out

Contd

- What if ABC(X) is already defined?
- #ifndef - #endif

```
#ifndef ABC(X)
    #define ABC(X) X*X
#endif
```

End Of Lecture 4

? QUESTIONS ?