LECTURE NOTE 15

ACTUAL ARGUMENTS AND FORMAL ARGUMENTS

Actual Arguments:

1. Arguments which are mentioned in the function in the function call are known as calling function.
2. These are the values which are actual arguments called to the function.

It can be written as constant , function expression on any function call which return a value .

ex: funct (6,9) , funct (a,b)

Formal Arguments:

1. Arguments which are mentioned in function definition are called dummy or formal argument.
2. These arguments are used to just hold the value that is sent by calling function.
3. Formal arguments are like other local variables of the function which are created when function call starts and destroyed when end function.

Basic difference between formal and local argument are:

a) Formal arguments are declared within the ( ) where as local variables are declared at beginning.

b) Formal arguments are automatically initialized when a value of actual argument is passed.

c) Where other local variables are assigned variable through the statement inside the function body.

Note: Order, number and type of actual argument in the function call should be matched with the order , number and type of formal arguments in the function definition .

PARAMETER PASSING TECHNIQUES:

1. call by value
2. call by reference

Call by value:

Here value of actual arguments is passed to the formal arguments and operation is done in the formal argument.

Since formal arguments are photo copy of actual argument, any change of the formal arguments does not affect the actual arguments

Changes made to the formal argument t are local to block of called function, so when control back to calling function changes made vanish.

Example:

```
void swap (int a , int b)        /* called function */
    {
    int t;
    t = a;
    a=b;
    b = t;
  }
main( )
 {
    int k = 50,m= 25;
    swap( k, m) ;    / * calling function */ print
    (k, m);      / * calling function */
    }
```

Output:

50, 25

Explanation:

```
    int k= 50, m=25 ;
```

Means first two memory space are created k and m , store the values 50 and 25 respectively.

swap (k,m);

When this function is calling the control goes to the called function.

void swap (int a , int b),
k and m values are assigned to the ‘a’ and ‘b’.
then a= 50 and b= 25 ,

After that control enters into the function a temporary memory space ‘t’ is created when int t is executed.

t=a; Means the value of a is assigned to the t , then t= 50.

a=b; Here value of b is assigned to the a , then a= 25;

b=t; Again t value is assigned to the b , then b= 50;

after this control again enters into the main function and execute the print function print (k,m). it returns the value 50 , 25.

NOTE:

Whatever change made in called function not affects the values in calling function.

Call by reference:

Here instead of passing value address or reference are passed. Function operators or address rather than values .Here formal arguments are the pointers to the actual arguments.

Example:

```
#include<stdio.h>
void add(int *n);
int main()
    {
    int num=2;
    printf("\n The value of num before calling the function=%d", num);
    add(&num);
    printf("\n The value of num after calling the function = %d", num);
    return 0;
    }
    void add(int *n)
    {
```

```
   *n=*n+10;
printf("\n The value of num in the called function = %d", n);
}
```

Output:

The value of num before calling the function=2

The value of num in the called function=20 The

value of num after calling the function=20

NOTE:

In call by address mechanism whatever change made in called function affect the values in calling function.

EXAMPLES:

1: Write a function to return larger number between two numbers:

```
int fun(int p, int q)
{
int large;
if(p>q)
 {
large = p;
 }
 else
  {
large = q;
}
return large;
}
```

2: Write a program using function to find factorial of a number.

```
#include <stdio.h> int
factorial (int n)
{
   int i, p;
   p = 1;
  for (i=n; i>1; i=i-1)
```

```
    {
        p = p * i;
    }

    return (p);
}

void main()
{
    int a, result;
    printf ("Enter an integer number: ");
    scanf ("%d", &a);
    result = factorial (a);
    printf ("The factorial of %d is %d.\n", a, result);
}
```

EXERCISE:

1. What do you mean by function?
2. Why function is used in a program?
3. What do you mean by call by value and call by address?
4. What is the difference between actual arguments and formal arguments?
5. How many types of functions are available in C?
6. How many arguments can be used in a function?
7. Add two numbers using
   a) with argument with return type
   b) with argument without return type
   c) without argument without return type
   d) without argument with return type
8. Write a program using function to calculate the factorial of a number entered through the keyboard.
9. Write a function power(n,m), to calculate the value of n raised to m.
10. A year is entered by the user through keyboard. Write a function to determine whether the year is a leap year or not.
11. Write a function that inputs a number and prints the multiplication table of that number.
12. Write a program to obtain prime factors of a number. For Example: prime factors of 24 are 2,2,2 and 3 whereas prime factors of 35 are 5 and 7.
13. Write a function which receives a float and a int from main(), finds the product of these two and returns the product which is printed through main().

14. Write a function that receives % integers and returns the sum, average and standard deviation of these numbers. Call this function from main() and print the result in main().
15. Write a function that receives marks obtained by a student in 3 subjects and returns the average and percentage of these marks. Call this function from main() and print the result in main().
16. Write function to calculate the sum of digits of a number entered by the user.
17. Write a program using function to calculate binary equivalent of a number.
18. Write a C function to evaluate the series
    $\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \ldots\ldots\ldots\ldots$ to five significant digit.
19. If three sides of a triangle are p, q and r respectively, then area of triangle is given by
    area = $(S(S-p)(S-q)(S-r))^{1/2}$, where $S = (p+q+r)/2$.
    Write a program using function to find the area of a triangle using the above formula.
20. Write a function to find GCD and LCM of two numbers.
21. Write a function that takes a number as input and returns product of digits of that number.
22. Write a single function to print both amicable pairs and perfect numbers.(Two different numbers are said to be amicable if the sum of proper divisors of each is equal to the other. 284 and 220 are amicable numbers.)
23. Write a function to find whether a character is alphanumeric.