9. User defined functions

* Need for user defined functions

We know that every program starts with main(). By only using the main it become large and complex for testing and debugging.

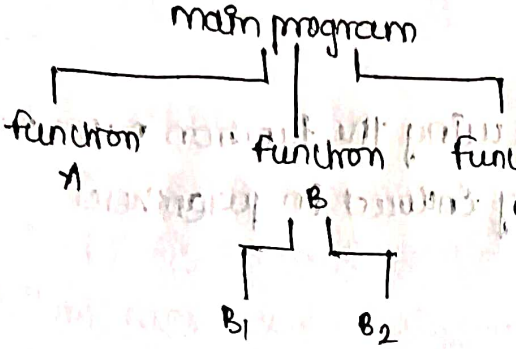

Fig: To make the main function simple

* Modular programming

- Each module should do only one thing
- Communication between modules is allowed only by a calling module
- A module can be called by one and only one higher module
- No communication can take place directly between modules that do not have calling called relationship
- All modules are designed as single-entry, single exit systems using control statements

* Elements of user defined functions

In order to make use of a user-defined function, we need to establish three elements that are related to function

- function ~~declaration~~) definition.
- function call
- function declaration

* Definition of function

A function definition, also known as function implementation shall include the following elements

1. function name,
2. function type
3. list of parameters

} function header

4. local variable declarations
5. function statements
6. a return statement

} function body

* return values and their types

return;
(or)
return expression;

* function calls

→ A function can be called by simply using the function name followed by a list of actual parameters, if any enclosed in parentheses.

Example: main()
{
int y;
y = mul(10,5); ← function call
printf("%d\n", y);
}

* function declaration

all functions in a C program must be declared, before they are invoked.

→ A function declaration consists of four parts

- function type
- function name
- parameter list
- Terminating semicolon :

function type function_name(parameter list);

Eg: int mul(int m, int n);

* category of functions

1. Functions with no arguments and no return values.
2. Functions with arguments and no return values
3. function with arguments and one return value
4. functions with no arguments but return a value
5. Functions that return multiple values

1. Sum of two numbers.

Start
↓
define the function sum(a,b)
that returns a+b
↓
declare the two number in
main (num1, num2)
↓
and read the inputs
↓
call sum(num1, num2) → display the result → stop

2. Factorial of a number.

Start → define the function factorial(n) → declare the inputs and read it
n * factorial(n-1);

call the factorial function → display the result. → stop.

3. Power of a number

Start → define the power(base, exp) → declare the inputs base and exponent
that returns base^exp

call function → display result → stop

4. check Even/odd

Start → define even function by → declare & read the inputs → call function
if n%==0

if function returns 0 → print even → stop
else → print odd

5. Largest Element in an array

Start → define largest function
that returns max elem
by using for loop to
Each element in array
→ declare the inputs → call function

display largest Element → stop

6. Sum of all elements in an array

Start → define the sum function that returns the sum by using for loop Sum = Sum + arr[i]; → declare & read the input in main → call function

Stop. ← display Sum ←

7. count Even and odd elements

1. Start
2. define count function that returns the no. of Even and odd elements by using the loop → if arr[i] % 2 == 0 then Even++ else odd++
3. declare the input array size and elements
4. call function to find the Even & odd
5. display count → stop.

8. find length of a string

1. Start
2. define Stringlength returns length by checking the condition str[i] != '\0'
3. Input string
4. call function
5. display length
6. Stop.

9. count vowels in a string

1. Start
2. define countvowels that returns no. of vowels.
3. Input string in main function
4. display count
5. Stop.

10. Convert string to upper case.

Start → define upper case (ch) → declare the string → call function → display → stop

11. calculate area of triangle
Start → define the function that returns 0.5 * base * height → declare the inputs → call function → display area → stop.

12. convert fahrenheit to celsius
Start → define the function that returns (f-32)* 5/9 ; → declare the inputs → call function → display → stop

13. minimum of three numbers
Start → define the function → declare the inputs → call function → display → stop
check the min value by if

14. calculate simple interest
Start → define the function (p*r*t)/100 → declare the inputs → call function → display → stop

15. swap of two numbers
Start → define the function by temp → declare the inputs → call function → display → stop

16. print pattern (Tri)
Start → define print(n).
By using two for loops → declare the inputs → call function → stop

17. Reverse a number
Start → define function
rev = rev * 10 + n%10;
n = n/10; → declare inputs → call function → stop

18. Sum of digits
Start → define function
sum = sum + n%10;
n = n/10; → declare inputs → call function → stop

Q1. difference of two numbers.

```
#include <stdio.h>
int difference (int a, int b) {
    return a - b;
}
int main() {
    int num1, num2;
    printf("Enter two numbers:");
    scanf("%d %d", &num1, &num2);
    printf("difference = %d\n", difference(num1, num2));
    return 0;
}
```

Q2 check prime

```
#include <stdio.h>
int isPrime(int n) {
    if (n<2) return 0;
    for(int i=2; i*i<=n; i++)
        if(n%i==0) return 0;
    return 1;
}
int main() {
    int num;
    printf("Enter a number:");
    scanf("%d", &num);
    if(isPrime(num)) printf("prime\n");
    else printf("Not prime\n");
    return 0;
}
```

Q3. find GCD.

```
#include <stdio.h>
int gcd(int a, int b) {
    while(b!=0) {
        int temp = b;
        b = a%b;
        a = temp;
    }
    return a;
}
int main() {
    int x,y; printf("Enter two numbers");
    scanf("%d %d", &x, &y);
    printf("GCD = %d\n", gcd(x,y));
    return 0;
}
```

24. convert string to lowercase

```
#include <stdio.h>
void toLowerCase(char str[]) {
    for (int i=0; str[i] != '\0'; i++) {
        if (str[i] >= 'A' && str[i] <= 'Z')
            str[i] += 32;
    }
}
int main() {
    char str[100];
    printf("Enter a string"); scanf("%s", str);
    toLowerCase(str);
    printf("Lowercase : %s\n", str);
    return 0;
}
```

25. Sum of N natural numbers

```
#include <stdio.h>
int sumNatural(int n) {
    int sum=0;
    for (int i=1; i<=n; i++)
        sum += i;
    return sum;
}
int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    printf("Sum = %d\n", sumNatural(n));
    return 0;
}
```

26. print multiplication table

```
#include <stdio.h>
void multiplicationTable(int n) {
    for (int i=1; i<=10; i++)
        printf("%d x %d = %d\n", n, i, n*i);
}
int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    multiplicationTable(n);
    return 0;
}
```