Structure of a C Program

- A program in C language is a collection of functions and declarations. It also have C preprocessor (cpp) directives.
- A function named 'main()' is mandatory.

C Function

- A function definition in C language has
  - a name,
  - a list of formal parameters (optional),
  - type of the value it returns (if any),
  - local variable declarations (if any), and
  - a sequence of statements[a].

[a] There may be comments for better human understanding.

An Example with Three Functions

```
#include <stdio.h> // cpp directive
int gcd(int, int); // Func Interface
int sum(int small, int large) // Def
{
    int i, total = 0;
    if(small > large){ int temp=small;
       small=large; large=temp;
    }
    for(i=small;i<=large;++i) total += i;
    return total;
} // Def ends
```

```
int main() { // main function
    int large, small;
    printf("Enter two non-ve integers\n");
    scanf("%d%d", &large, &small);
    printf("%d + ... + %d = %d\n",
            small,large,sum(large,small));
    printf("GCD(%d, %d) = %d\n",
            large,small,gcd(large,small));
    return 0;
}
```

```
int gcd(int large, int small)
{
    if(small == 0) return large;
    return gcd(small, large%small);
} // sample.c
```

File Name

File name of a C program ends with a .c, in this case it is sample.c.

Discussion

```
#include <stdio.h>
```

A line starting with a sharp sign '#' is a C preprocessor directive. This directive tells the preprocessor to include the header file for the standard I/O functions from the header files directory (often /usr/include).

```
int gcd(int, int);
```

- The function `gcd()` is called (invoked) in `main()` before its definition.
- The compiler while translating the code sequentially, encounters the invocation of `gcd()` before its definition. It does not have any clue regarding the number and the types of parameters, and the type of the return value of `gcd()`.

```
int gcd(int, int);
```

This type of statement provides inform regarding the number and types of the parameters, and the type of the return value of a function yet to be defined. It is known as the prototype or interface of the function[a].

[a] The header file `stdio.h` provides the prototypes of `printf()` and `scanf()`.

```
int sum(int small, int large) // Def
{
    int i, total = 0;
    if(small > large){ int temp=small;
       small=large; large=temp;
    }
    for(i=small;i<=large;++i) total += i;
    return total;
} // Def ends
```

The actual function definition specify name, parameters, computation and return value.

Parameters and Return Value

```
int sum(int small, int large)
```

- The function sum: `int` $\times$ `int` $\rightarrow$ `int`. This may be viewed as a mathematical function.
- Two named parameter are `small` and `large`.
- It computes and returns a value of type `int`.

Variable Declaration

```
int i, total = 0;
```

- `i` and `total` are variables local to the function `sum()`.
- Currently `i` does not have any value (contains garbage), but `total` is initialized to zero (0).

Statements

```
small=large;
```

```
if(small > large){ ··· }
```

```
for(i=small;i<=large;++i) ···;
```

```
return total;
```

Operators

Assignment: =

Relational: <, <=

Special assignment: ++, + =

Library Functions

Reads from Keyboard: `scanf()`

Writes on the VDU: `printf()`

Library functions are supplied along with the compiler.

Compile and Run

```
$ cc -Wall sample.c
$ ./a.out
Enter two non-ve integers
12 18
18 + ...  + 12 = 105
GCD(12, 18) = 6
```

Note: Replace `cc` by `gcc` in the laboratory.

Macro Definition: #define

```
#include <stdio.h>
#define MAX 100
int main() {
    int n=MAX;

    n = n+MAX;
    printf("n + MAX is: %d\n", n + MAX);
    return 0;
} // preProc1.c
```

Macro Definition

#define *identifier tokens*

After Substitution

```
int main() {
    int n=100;

    n = n+100;
    printf("n + MAX is: %d\n", n + 100);
    return 0;
}
```

Macro Definition with Parametera

```
#include <stdio.h>
#define EXCH(X,Y,T) ((T)=(X),(X)=(Y),(Y)=(T))
int main() {
    int m, n, temp;

    scanf("%d%d", &m, &n);
    printf("m: %d, n: %d\n", m, n);
    EXCH(m,n,temp);
    printf("m: %d, n: %d\n", m, n);
    return 0;
} // preProc2.c
```

After Substitution

```
int main() {
    int m, n, temp;

    scanf("%d%d", &m, &n);
    printf("m: %d, n: %d\n", m, n);
    ((temp)=(m), (m)=(n), (n)=(temp));
    printf("m: %d, n: %d\n", m, n);
    return 0;
}
```

Note

Use of parenthesis around the parameters is safer. Otherwise there may be semantic error.

```
#define MULT(X,Y) X*Y
    ........
printf("2*(m+n): %d\n", MULT(2,m+n));
```

After substitution:

```
printf("2*(m+n): %d\n", 2*m+n);
```

Example - II

```
/* second.c */
#include <stdio.h>
int main() {
    int n ;

    printf("Enter a non Negative integer: ") ;
    scanf("%d", &n) ; /* Reads an integer */
    printf("\nSum of (0 + ... + %d) = %d\n",
                    n, n*(n + 1)/2 ) ;
    return 0 ;
}
```

Example - III

```
/* Area of a Circle: third.c */
#include <stdio.h>
#include <math.h>
int main() {
    float radius, area ;

    printf("Enter the radius :") ; scanf("%f", &radius) ;
    area = 4.0 * atan(1.0) * radius * radius ;
    printf("\n Circle-Area = %f for Radius = %f\n",
                              area, radius) ; return 0 ;
}
```

Compile with Mathematical Library

It may be necessary to compile with mathematical library.

```
$ cc -Wall -lm third.c
Where is the mathematical library?
$ cd /lib
$ ls -l libm*.*
-rwxr-xr-x 1 root root 177315 Dec 20
2004 libm-2.3.4.so
lrwxrwxrwx 1 root root 13 Jul 12 2006
libm.so.6 -> libm-2.3.4.so
```

Example - IV

```
// Convert Fahrenheit to Celsius: fourth.c
#include <stdio.h>
int main(){
    float cel, fah ;
    printf("Enter temp. in F: ") ;
    scanf("%f", &fah) ;
    cel = 5.0*(fah-32.0)/9.0 ;
    printf("%6.2f F = %6.2f C\n", fah, cel) ;
    return 0 ;
}
```

Example - V

```
#include <stdio.h>
int main() {    // **** fifth.c
    int first, second, max ;

    printf("Enter two integers :") ;
    scanf("%d%d", &first, &second) ;
    if(first > second) max = first ;
    else max = second ;
    printf("\nMax(%d,%d) = %d\n",
           first, second, max) ; return 0 ;
}
```

Example - VI

```
#include <stdio.h>
int main() {          // **** sixth.c
    int n, i, sum = 0 ;

    printf("Enter a non-negative integer:") ;
    scanf("%d", &n) ;
    for(i = 0; i <= n; ++i) sum += i ;
    printf("\nSum of (0 + ... + %d) = %d\n",
                                  n, sum ) ;
    return 0 ;
}
```

Example - VII

```
int sum(int n) {  // *** seventh.c
    if(n==0) return 0;
    else return n+sum(n-1);
}
#include <stdio.h>
int main() {
    int n ;
    printf("Enter a non-negative integer:") ;
    scanf("%d", &n) ;
    printf("\nSum of (0+...+%d)=%d\n",n,sum(n));
    return 0 ;
}
```