

# Functions

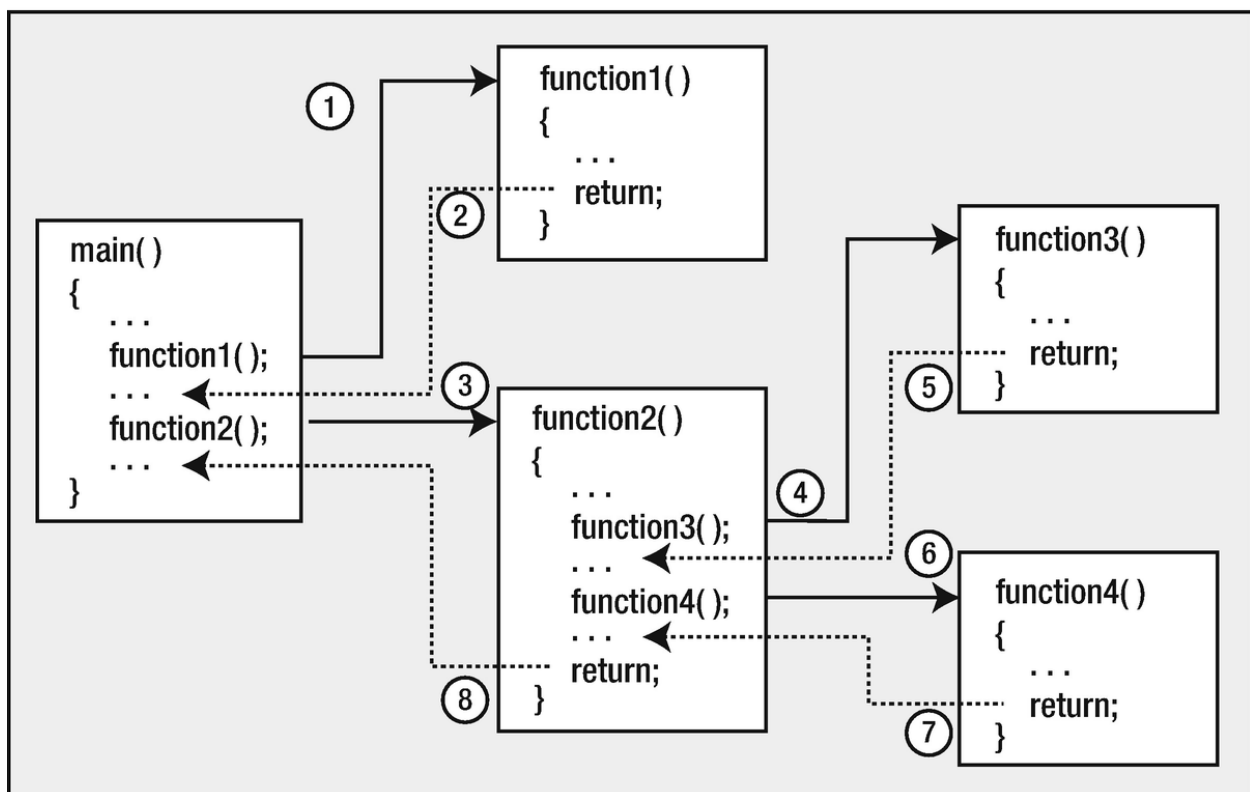
☰ Tags	c functions
🕒 Created time	@September 26, 2024 11:41 AM
# Lecture No.	6
☑ Reviewed	<input type="checkbox"/>

A C program consists of one or more functions, the most important of which is the `main()` where execution starts.

Each function is self-contained and carries out a particular operation.

When a function is called, the code within the function is executed, and when it is complete, it returns to the point where the function was called.

The below figure shows the sequence of execution in a C program.



## Example:

Write a program `gradeSD.c` to calculate the standard deviation (SD) of the input grades.

**Approach:** To calculate SD we need the average. To calculate average, we need a summary of grades. So we will need three functions.

## Functions

A function is a self-contained block of code. The name is a unique sequence of letters and digits, but the first must be a letter (this also includes an `_`)

### Defining Functions

```
Return_type Function_name( Paramters - separated by commas ) {  
    // Statements to be carried out  
}
```

- If the `return_type` is void, the function does not need to return any value.
- If `return_type` is not void, every return statement in the function must return a value of the specified type.

### Example Calculate Sum Function

```
/*  
    Function Name: calculateSum  
    Parameters:  
        array of ints called grades,  
        int length of array  
    Returns: int  
*/
```

```

int calculateSum(int grades[], int length) {
    // local variable, only exist in this function
    int sum = 0;

    for (int i = 0; i < length; i++)
    {
        /* code */
        sum += grades[i];
    }

    return sum;
}

/* In our main function we call this function like this */
int sum = calculateSum(studentGrades, length)

```

## Calling Functions

```

Function_name(List of Arguments - separated by commas)

```

## Template of an Idealised C Program

```

/* short intro about the program, e.g. author, date, description
/* include relevant libraries */
#include<stdio.h>
// etc..

/* function prototypes */
int functionName1(int param1, float param2, char param3);
float functionName2(int param1, float param2, char param3);
void functionName3(int param1, float param2, char param3);
// etc...

```

```

/* main function */
int main(int argc, char * argv[])
{
    /* statments */
    /* the declared functions above can be call here where neces

    return 0;
}

/* actual implementation of the functions */
int functionName1(int param1, float param2, char param3)
{
    //statements
    return an_interger;
}

float functionName2(int param1, float param2, char param3)
{
    //statements
    return a_float;
}

void functionName3(int param1, float param2, char param3)
{
    //statements
}

```