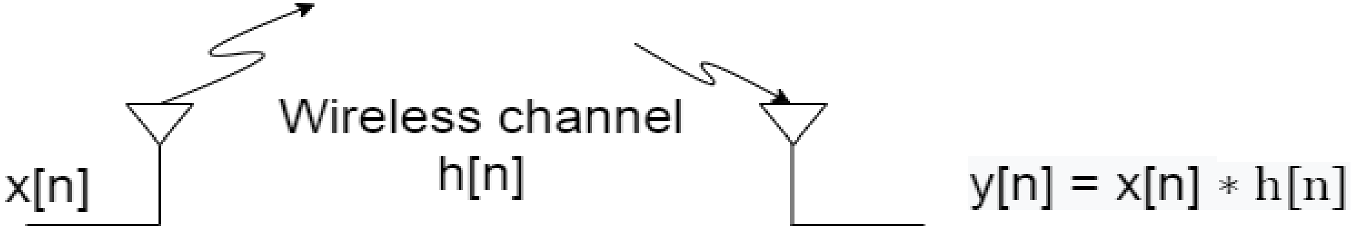
Convolution

Convolution of two sequence x[n] and h[n] is given as

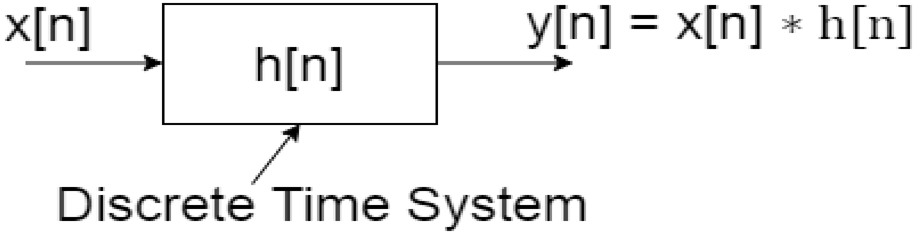
$$y[n] = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k)$$

Application :

1. Wireless communication



2. Discrete time systems



Correlation

Correlation of two sequence x[n] and y[n] is given as

$$\boldsymbol{R_{xy}[k] = \sum_{n=0}^{\infty} x(n)\, y(n-k)}$$

Application :

1. Finding Similarity between two signals or sequences.
2. Synchronization between base station and mobile station.

Downsampling

Given a discrete sequence x[n] , the downsampled signal y[n] is

$$\boldsymbol{y[n] = x[Mn]}$$

Application :

- To reduce the sampling rate of a signal.

Upsampling

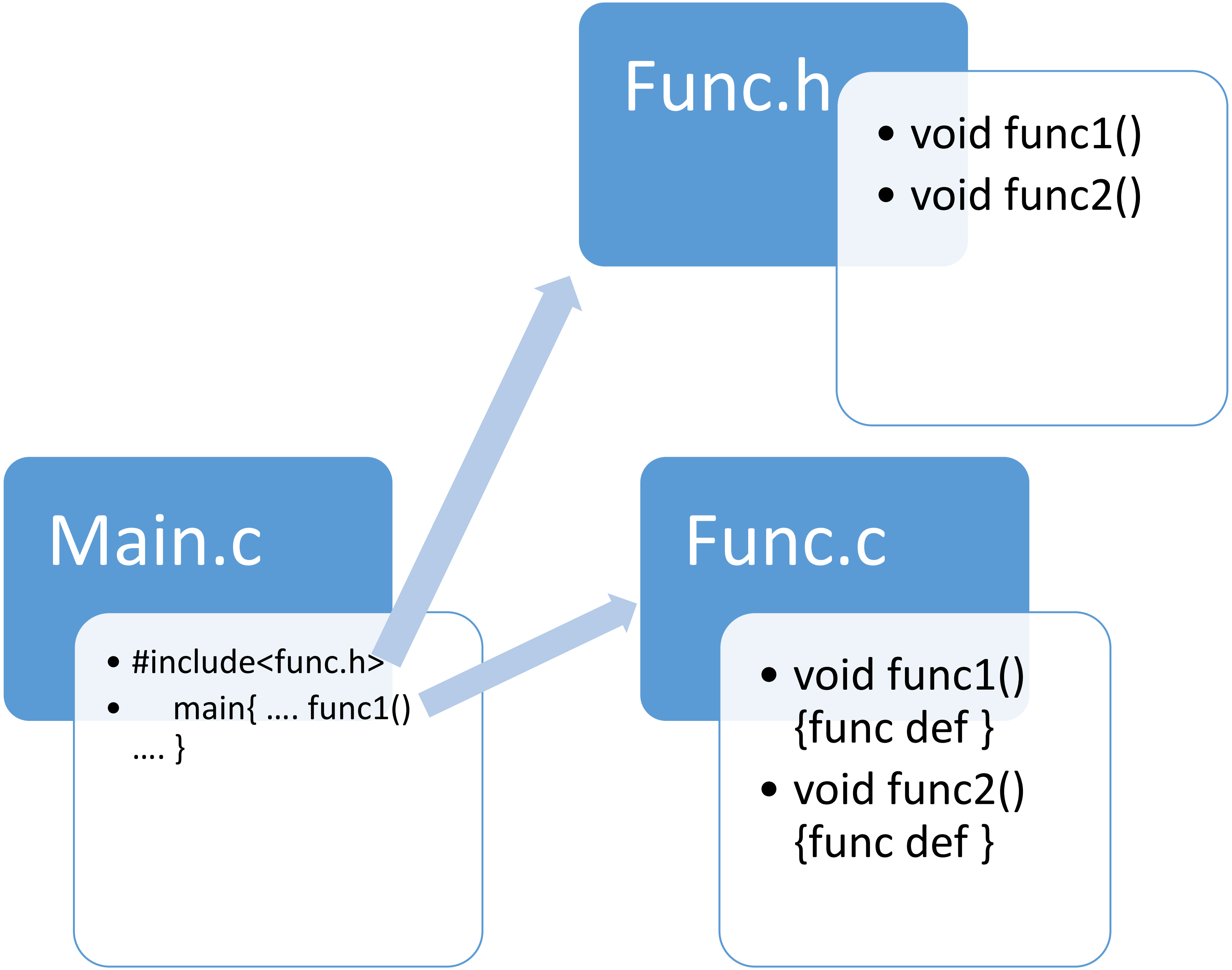
Given a discrete sequence x[n] , the upsampled signal y[n] is

$$\boldsymbol{y[n] = \begin{cases} x[n/L], & if\ n\ is\ a\ multiple\ of\ L \\ 0, & otherwise \end{cases}}$$

Application :

- To Increase the sampling rate of a signal.

Programming Control flow



Func.h

- void func1()
- void func2()

Main.c

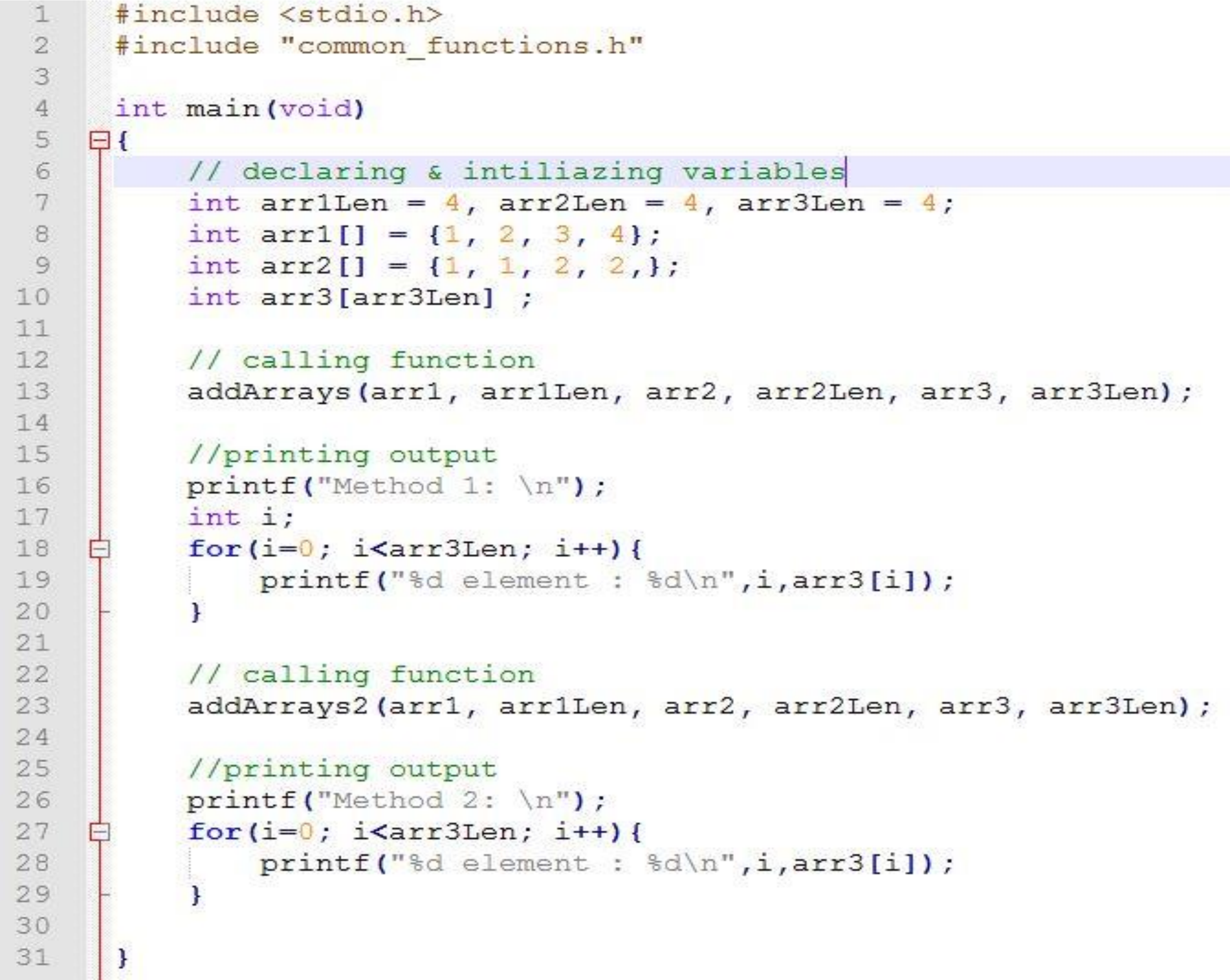
- #include<func.h>
- main{ …. func1() …. }

Func.c

- void func1() {func def }
- void func2() {func def }

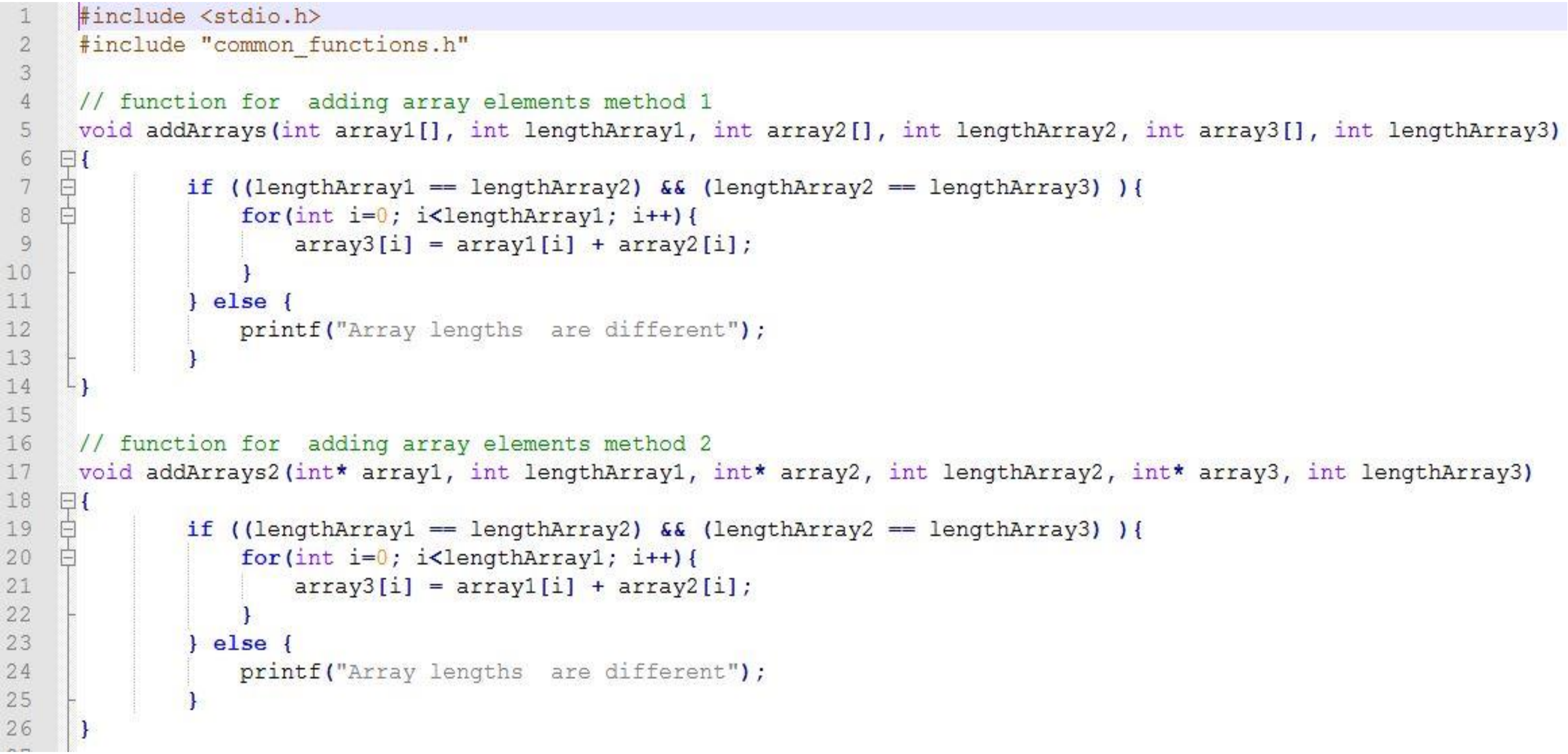
Sample code

main.c



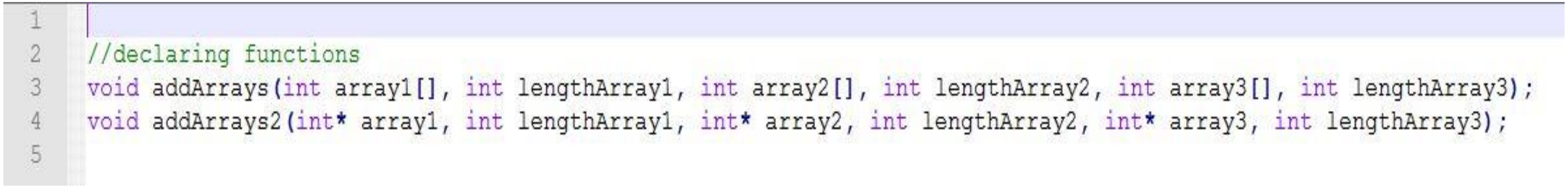
```c
#include <stdio.h>
#include "common_functions.h"

int main(void)
{
    // declaring & intiliazing variables
    int arr1Len = 4, arr2Len = 4, arr3Len = 4;
    int arr1[] = {1, 2, 3, 4};
    int arr2[] = {1, 1, 2, 2,};
    int arr3[arr3Len] ;

    // calling function
    addArrays(arr1, arr1Len, arr2, arr2Len, arr3, arr3Len);

    //printing output
    printf("Method 1: \n");
    int i;
    for(i=0; i<arr3Len; i++){
        printf("%d element : %d\n",i,arr3[i]);
    }

    // calling function
    addArrays2(arr1, arr1Len, arr2, arr2Len, arr3, arr3Len);

    //printing output
    printf("Method 2: \n");
    for(i=0; i<arr3Len; i++){
        printf("%d element : %d\n",i,arr3[i]);
    }

}
```

common_functions.c



```c
#include <stdio.h>
#include "common_functions.h"

// function for  adding array elements method 1
void addArrays(int array1[], int lengthArray1, int array2[], int lengthArray2, int array3[], int lengthArray3)
{
        if ((lengthArray1 == lengthArray2) && (lengthArray2 == lengthArray3) ){
            for(int i=0; i<lengthArray1; i++){
                array3[i] = array1[i] + array2[i];
            }
        } else {
            printf("Array lengths  are different");
        }
}

// function for  adding array elements method 2
void addArrays2(int* array1, int lengthArray1, int* array2, int lengthArray2, int* array3, int lengthArray3)
{
        if ((lengthArray1 == lengthArray2) && (lengthArray2 == lengthArray3) ){
            for(int i=0; i<lengthArray1; i++){
                array3[i] = array1[i] + array2[i];
            }
        } else {
            printf("Array lengths  are different");
        }
}
```

common_functions.h



```c

//declaring functions
void addArrays(int array1[], int lengthArray1, int array2[], int lengthArray2, int array3[], int lengthArray3);
void addArrays2(int* array1, int lengthArray1, int* array2, int lengthArray2, int* array3, int lengthArray3);

```