

Return pointer from functions in C

We have seen in the last chapter how C programming allows to return an array from a function. Similarly, C also allows to return a pointer from a function. To do so, you would have to declare a function returning a pointer as in the following example –

```
int * myFunction() {  
    .  
    .  
    .  
}
```

Second point to remember is that, it is not a good idea to return the address of a local variable outside the function, so you would have to define the local variable as **static** variable.

Now, consider the following function which will generate 10 random numbers and return them using an array name which represents a pointer, i.e., address of first array element.

```
#include <stdio.h>  
#include <time.h>  
  
/* function to generate and return random numbers. */  
int * getRandom( ) {  
  
    static int  r[10];  
    int i;  
  
    /* set the seed */  
    srand( (unsigned)time( NULL ) );  
  
    for ( i = 0; i < 10; ++i) {  
        r[i] = rand();  
        printf("%d\n", r[i] );  
    }
```

[Live Demo](#)

```
}

return r;
}

/* main function to call above defined function */
int main () {

    /* a pointer to an int */
    int *p;
    int i;

    p = getRandom();

    for ( i = 0; i < 10; i++ ) {
        printf("(p + [%d]) : %d\n", i, *(p + i) );
    }

    return 0;
}
```

When the above code is compiled together and executed, it produces the following result –

```
1523198053
1187214107
1108300978
430494959
1421301276
930971084
123250484
106932140
1604461820
149169022
*(p + [0]) : 1523198053
*(p + [1]) : 1187214107
*(p + [2]) : 1108300978
*(p + [3]) : 430494959
*(p + [4]) : 1421301276
*(p + [5]) : 930971084
```

```
*(p + [6]) : 123250484  
*(p + [7]) : 106932140  
*(p + [8]) : 1604461820  
*(p + [9]) : 149169022
```