# Types of User-defined Functions in C Programming

These 4 programs below check whether the integer entered by the user is a prime number or not.

The output of all these programs below is the same, and we have created a user-defined function in each example. However, the approach we have taken in each example is different.

## **Example 1: No arguments passed and no return value**

1. #include <stdio.h>
2. void checkPrimeNumber();
3. int main()
4. {
5. checkPrimeNumber(); // argument is not passed
6. return 0;
7. }
8. // return type is void meaning doesn't return any value
9. void checkPrimeNumber()
10. {
11. int n, i, flag = 0;
12. printf("Enter a positive integer: ");
13. scanf("%d",&n);
14. for(i=2; i <= n/2; ++i)
15. {
16. if(n%i == 0)
17. {
18. flag = 1;
19. }
20. }
21. if (flag == 1)
22. printf("%d is not a prime number.", n);
23. else
24. printf("%d is a prime number.", n);
25. }

The checkPrimeNumber() function takes input from the user, checks whether it is a prime number or not and displays it on the screen.

The empty parentheses in checkPrimeNumber(); statement inside the main() function indicates that no argument is passed to the function.

The return type of the function is void. Hence, no value is returned from the function.

## **Example 2: No arguments passed but a return value**

1. #include <stdio.h>
2. int getInteger();
3. int main()
4. {
5. int n, i, flag = 0;
6. // no argument is passed
7. n = getInteger();
8. for(i=2; i<=n/2; ++i)
9. {
10. if(n%i==0){
11. flag = 1;
12. break;
13. }
14. }
15. if (flag == 1)
16. printf("%d is not a prime number.", n);
17. else
18. printf("%d is a prime number.", n);
19. return 0;
20. }
21. // returns integer entered by the user
22. int getInteger()
23. {
24. int n;
25. printf("Enter a positive integer: ");
26. scanf("%d",&n);
27. return n;
28. }

The empty parentheses in the n = getInteger(); statement indicates that no argument is passed to the function. And, the value returned from the function is assigned to n.

Here, the getInteger() function takes input from the user and returns it. The code to check whether a number is prime or not is inside the main() function.

## **Example 3: Argument passed but no return value**

1. #include <stdio.h>
2. void checkPrimeAndDisplay(int n);
3. int main()
4. {
5. int n;
6. printf("Enter a positive integer: ");
7. scanf("%d",&n);
8. // n is passed to the function
9. checkPrimeAndDisplay(n);
10. return 0;
11. }
12. // return type is void meaning doesn't return any value
13. void checkPrimeAndDisplay(int n)
14. {
15. int i, flag = 0;
16. for(i=2; i <= n/2; ++i)
17. {
18. if(n%i == 0){
19. flag = 1;
20. break;
21. }
22. }
23. if(flag == 1)
24. printf("%d is not a prime number.",n);
25. else
26. printf("%d is a prime number.", n);
27. }

The integer value entered by the user is passed to the checkPrimeAndDisplay() function.

Here, the checkPrimeAndDisplay() function checks whether the argument passed is a prime number or not and displays the appropriate message.

## **Example 4: Argument passed and a return value**

1. #include <stdio.h>
2. int checkPrimeNumber(int n);
3. int main()
4. {
5. int n, flag;
6. printf("Enter a positive integer: ");
7. scanf("%d",&n);
8. // n is passed to the checkPrimeNumber() function
9. // the returned value is assigned to the flag variable
10. flag = checkPrimeNumber(n);
11. if(flag == 1)
12. printf("%d is not a prime number",n);
13. else
14. printf("%d is a prime number",n);
15. return 0;
16. }
17. // int is returned from the function
18. int checkPrimeNumber(int n)
19. {
20. int i;
21. for(i=2; i <= n/2; ++i)
22. {
23. if(n%i == 0)
24. return 1;
25. }
26. return 0;
27. }

The input from the user is passed to the checkPrimeNumber() function.

The checkPrimeNumber() function checks whether the passed argument is prime or not.

If the passed argument is a prime number, the function returns 0. If the passed argument is a non-prime number, the function returns 1. The return value is assigned to the flag variable.

Depending on whether flag is 0 or 1, an appropriate message is printed from the main() function.

## **Which approach is better?**

Well, it depends on the problem you are trying to solve. In this case, passing argument and returning a value from the function (example 4) is better.

A function should perform a specific task. The checkPrimeNumber() function doesn't take input from the user nor it displays the appropriate message. It only checks whether a number is prime or not.