When an argument is passed by-reference, the corresponding parameter is an alias for the argument. An alias is another name for the same object. Whenever the value of the parameter is altered, so, too, is the value of the argument changed at the same time. An argument that is passed by-reference cannot be an expression since expressions have no name.

Append an & to the type identifier in the parameter declaration of the parameter you wish to pass by reference. For example, an & is appended to type int in the parameter declaration of parameter k on line 3 in Figure 1.

```
1. #include <iostream>
2. using namespace std;
3. void Ref(int& k)
4. {
5.     cout << "Ref.k=" << k << endl;
6.     k++;  
7.     cout << "Ref.k=" << k << endl;
8. }
9. int main()
10. {
11.     int r=1;
12.     cout << "main.r=" << r << endl;
13.     Ref(r);
14.     cout << "main.r=" << r << endl;
15.     return 0;
16. }
```

**Figure 1.** Call by-reference example.

Program **p01** prints:
main.r=1
Ref.k=1
Ref.k=2
main.r=2

Program **p01** notes.
1. Integer variable r is initialized to 1 on line 9.
2. The value of variable r is printed on line 10.
3. Function Ref is called on line 11. The alias k is bound to variable r on line 11. Any operation on parameter k is applied to variable r.
4. The value of parameter k is printed on line 4.
5. Parameter k is incremented on line 5.
6. The incremented value of parameter k is printed on line 6.
7. Function Ref returns on line 7. The alias k is destroyed on line 7.
8. The incremented value of variable r is printed on line 12.
A classic use of passing an argument by-reference is employed in function swap. Function swap interchanges the values of its two parameters as shown in Figure 2.

```
#include <iostream>
using namespace std;

void swap(int & m, int & w)
{
    int b = m; m = w; w = b;
}

int main()
{
    int a = 1, b = 2;
    cout << "a = " << a << " b = " << b << endl;
    swap(a, b);
    cout << "a = " << a << " b = " << b << endl;
    return 0;
}
```

Figure 2. Program p02 and function swap.

Program p02 prints:

```
a=1 b=2
a=2 b=1
```

Program p02 notes:

1. Integer variables a and b are initialized to 1 and 2 respectively on line 7.
2. The values of variables a and b are printed on line 8 so that their values are displayed.
3. Function swap is called on line 9.
4. The values of variables a and b are printed again on line 10 to show that their values have been interchanged.