

Call by Value

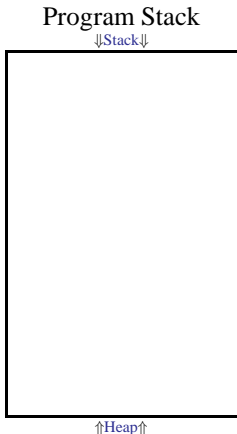
July 20, 2009

These figures show how call by value prevents a function's changes to its parameters from being seen in the calling function.

- ▶ The `arg` parameter of `increment` is passed by value (`int`).
- ▶ When `increment` calls `arg++`, the function changes its own “local” copy of the variable (located on the stack, underneath of its call frame).
- ▶ `main::v` is unaffected since its value is stored elsewhere on the stack.

The Operating System calls main ()

<OS>

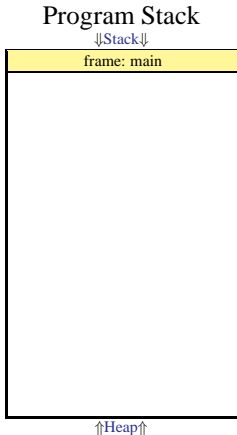


```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         " before increment " << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         " after increment " << endl;
21
22     return 0;
23 }
```

The Operating System calls `main()`

create call frame for `main`

<OS>



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         "\u0322before\u0322increment " << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         "\u0322after\u0322increment " << endl;
21
22     return 0;
23 }
```

Line 11: Function main entry

<OS> \implies main

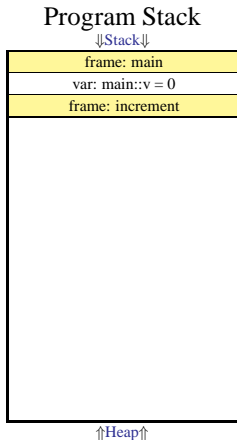


```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main() // <==
12 {
13     int v(0);
14     cout << "v=" << v <<
15         "_before_increment" << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         "_after_increment" << endl;
21
22     return 0;
23 }
```


Line 17: Call function **increment**

create call frame for **increment**

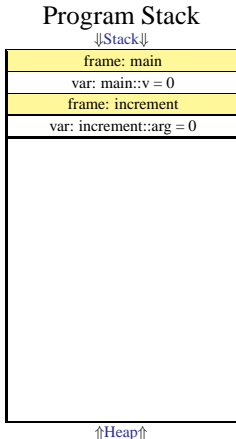
<OS> \implies main



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         "\nbefore_increment" << endl;
16
17     increment( v );           // <==
18
19     cout << "v=" << v <<
20         "\nafter_increment" << endl;
21
22     return 0;
23 }
```


Line 5: Function increment entry

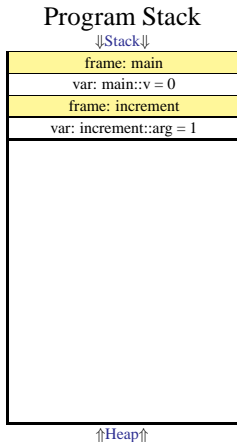
<OS> \implies main \implies increment



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )           // <==
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         " _before_increment " << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         " _after_increment " << endl;
21
22     return 0;
23 }
```

Line 7: Increment arg

<OS> \implies main \implies increment

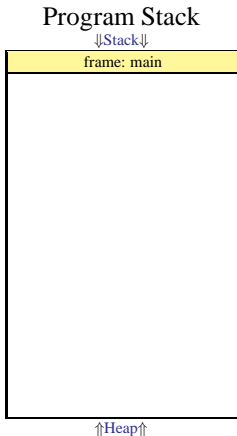


```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg; // <==
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         "\u2014before\u2014increment" << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         "\u2014after\u2014increment" << endl;
21
22     return 0;
23 }
```


Line 22: Return to OS

discard the auto (local) variable `main::v`

<OS> \implies main



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         " before increment" << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         " after increment" << endl;
21
22     return 0; // <==
23 }
```

Line 22: Return to OS

discard call frame, return to caller

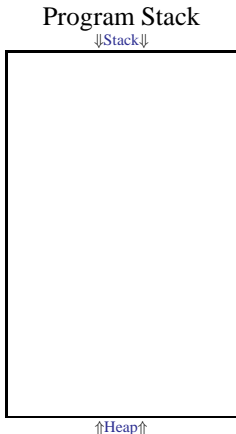
<OS> \implies main



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         " before increment" << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         " after increment" << endl;
21
22     return 0; // <==
23 }
```

Return from main

<OS>



```
1 #include <iostream>
2 using namespace std;
3
4 /** increments arg by one */
5 void increment( int arg )
6 {
7     ++arg;
8     return;
9 }
10
11 int main()
12 {
13     int v(0);
14     cout << "v=" << v <<
15         "\u0322before\u0322increment" << endl;
16
17     increment( v );
18
19     cout << "v=" << v <<
20         "\u0322after\u0322increment" << endl;
21
22     return 0;
23 }
```

finis