

Local Gargbage

July 20, 2009

The following example shows why uninitialized local variables frequently contain nonsense or “garbage values.”

In this code, two functions use two differently named local variables (`powerOf2::r` and `powerOf3::s`). When the code path enters `powerOf3`, `s` will be initialized with the last value of `powerOf2::r`!

Why? Because they both end up occupying the same memory space. Since `powerOf3::s` is not properly initialized, its bits reflect the last value of `powerOf2::r`.

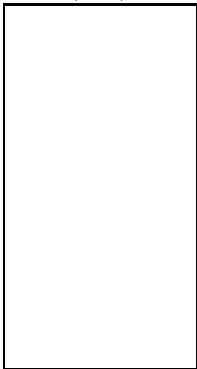
```
1 /**
2  * powerOf3::s is garbage
3  */
4 #include <iostream>
5 using namespace std;
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

The Operating System calls main()

<OS>

Program Stack

↓Stack↓



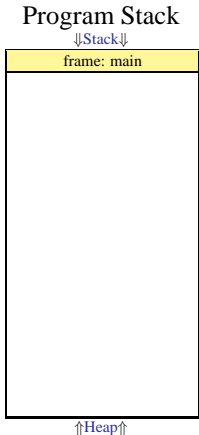
↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

The Operating System calls main()

create call frame for main

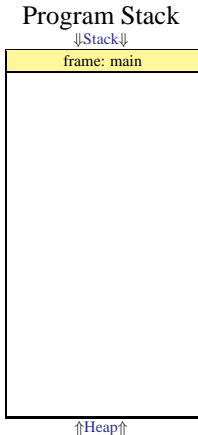
<OS>



```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 22: Function main entry

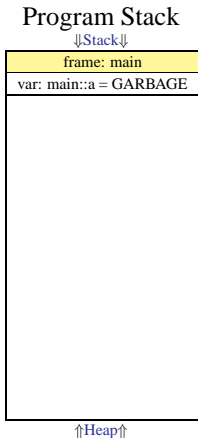
<OS> \implies main



```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main() // <==
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 24: Declare a

<OS> \implies main



```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b; // <==
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

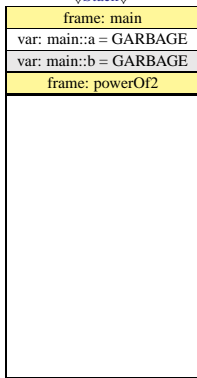

Line 25: Call powerOf2(4)

create call frame for powerOf2

<OS> \implies main

Program Stack

↓Stack↓



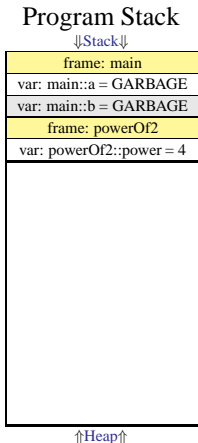
↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);           // <==
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 25: Call powerOf2(4)

create formal param power of powerOf2

<OS> \implies main



```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4); // <==
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 6: Function powerOf2 entry

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4

↑Heap↑

```
6 int powerOf2( int power )      // <==
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```


Line 9: Declare x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 1
var: powerOf2::x = 1

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) { // <==
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 10: Store $r *= 2$

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 2
var: powerOf2::x = 1

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2; // <==
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 9: Increment x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 2
var: powerOf2::x = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) { // <==
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 10: Store $r *= 2$

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 4
var: powerOf2::x = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2; // <==
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 9: Increment x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 4
var: powerOf2::x = 3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) { // <==
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 10: Store $r *= 2$

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 8
var: powerOf2::x = 3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2; // <==
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 9: Increment x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 8
var: powerOf2::x = 4

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) { // <==
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 10: Store $r *= 2$

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16
var: powerOf2::x = 4

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2; // <==
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 9: Increment x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16
var: powerOf2::x = 5

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) { // <==
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```


Line 11: Destroy loop counter x

<OS> \implies main \implies powerOf2

Program Stack

↓Stack↓

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }                                     // <==
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 11: Remember the Stack State

This is NOT done by the program, it is a feature of this slideshow.

<OS> \implies main \implies powerOf2

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }                                     // <==
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 12: Return value of r

<OS> \implies main \implies powerOf2

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r; // <==
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 12: Return value of r copy return value

<OS> \implies main \implies powerOf2

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r; // <==
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 12: Return value of r

discard the auto (local) variable powerOf2::r

<OS> \implies main \implies powerOf2

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r; // <==
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 12: Return value of `r` discard `powerOf2` parameter `power`

<OS> \implies main \implies `powerOf2`

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: <code>powerOf2</code>
var: <code>powerOf2</code> ::power = 4
var: <code>powerOf2</code> ::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: <code>powerOf2</code>

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r; // <==
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 12: Return value of r discard call frame, return to caller

<OS> \implies main \implies powerOf2

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r; // <==
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 25: Return from powerOf2

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);           // <==
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 26: Call powerOf3(2)

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2); // <==
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 26: Call powerOf3(2)

create call frame for powerOf3

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2); // <==
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 26: Call powerOf3(2)

create formal param power of powerOf3

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2); // <==
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 14: Function powerOf3 entry

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )      // <==
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 16: Declare s

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = ???

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized! // <==
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 16: Declare s

powerOf3::s Where powerOf2::r Was!

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized! // <==
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 17: Declare x

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 16
var: powerOf3::x = 1

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) { // <==
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 18: Store `s*=3`

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 48
var: powerOf3::x = 1

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3; // <==
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 17: Increment x

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 48
var: powerOf3::x = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) { // <==
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 18: Store s*=3

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144
var: powerOf3::x = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3; // <==
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 17: Increment x

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144
var: powerOf3::x = 3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) { // <==
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 17: Break out of for loop

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144
var: powerOf3::x = 3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) { // <==
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 19: Destroy loop counter x

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     } // <==
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 20: Return value of s

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = GARBAGE
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s; // <==
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 20: Return value of s copy return value

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144
frame: powerOf3
var: powerOf3::power = 2
var: powerOf3::s = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s; // <==
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 20: Return value of s

discard the auto (local) variable powerOf3::s

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144
frame: powerOf3
var: powerOf3::power = 2

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s; // <==
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 20: Return value of s

discard powerOf3 parameter power

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144
frame: powerOf3

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s; // <==
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 20: Return value of s discard call frame, return to caller

<OS> \implies main \implies powerOf3

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s; // <==
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 26: Return from powerOf3

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2); // <==
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 27: Print to console

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl; // <==
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Line 28: Print to console

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl; // <==
29     return 0;
30 }
```

Line 29: Return to OS

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16
var: main::b = 144

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0; // <==
30 }
```

Line 29: Return to OS

discard the auto (local) variable main::b

<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack

frame: main
var: main::a = 16

↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0; // <==
30 }
```


Line 29: Return to OS

discard call frame, return to caller

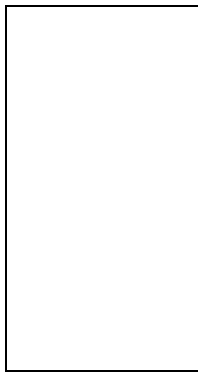
<OS> \implies main

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack



↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0; // <==
30 }
```

Return from main

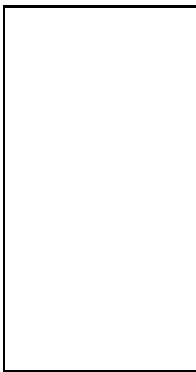
<OS>

Snapshot

frame: main
var: main::a = GARBAGE
var: main::b = GARBAGE
frame: powerOf2
var: powerOf2::power = 4
var: powerOf2::r = 16

↓Stack↓

Program Stack



↑Heap↑

```
6 int powerOf2( int power )
7 {
8     int r(1);
9     for( int x(1); x<=power; x++ ) {
10         r *= 2;
11     }
12     return r;
13 }
14 int powerOf3( int power )
15 {
16     int s; // uninitialized!
17     for( int x(1); x<=power; x++ ) {
18         s *= 3;
19     }
20     return s;
21 }
22 int main()
23 {
24     double a, b;
25     a = powerOf2(4);
26     b = powerOf3(2);
27     cout << "2*2*2*2=" << a << endl;
28     cout << "3*3=" << b << endl;
29     return 0;
30 }
```

Buffer Overflow Attacks

This simple observation: if you know what *used* to be on the stack (or heap), you can accurately *predict* how a program will *erroneously* behave; is the fundamental building block of **Buffer Overflow Attacks** on the Internet.

It is a non-trivial amount of work to develop a buffer overflow attack, but people do it, so much so that buffer overflow attacks usually account for about 20% of known Internet exploits.

finis