Exercise 1

Exercise 1

Find **PRE- and POST-conditions** for this function.

1. Function:

```
double f (const double i,
          const double j,
          const double k)
{
  if (i > j)
    if (i > k)
      return i;
    else
      return k;
  else
    if (j > k)
      return j;
    else
      return k;
}
```

Exercise 1

PRE-Condition:
(not needed)

POST-Condition:

```
// POST: return value is
//       the maximum of
//       i,j and k
```

1. Function:

```
double f (const double i,
          const double j,
          const double k)
{
  if (i > j)
    if (i > k)
      return i;
    else
      return k;
  else
    if (j > k)
      return j;
    else
      return k;
}
```

Exercise 1

Find **PRE- and POST-conditions** for this function.

2. Function:

```
double g (const int i, const int j)
{
  double r = 0.0;
  for (int k = i; k <= j; ++k)
    r += 1.0 / k;
  return r;
}
```

Exercise 1

2. Function:

```
double g (const int i, const int j)
{
  double r = 0.0;
  for (int k = i; k <= j; ++k)
    r += 1.0 / k;
  return r;
}
```

PRE-Condition:
POST-Condition:

```
// PRE: 0 not contained in {i, ..., j}
// POST: return value is the sum
//          1/i + 1/(i+1) + ... + 1/j
```

(From: Script Exercise 79)

Exercise 2

Exercise 2

Find **3 mistakes** in this program.

```
# include <iostream>

double f (const double x) {
   return g(2.0 * x);
}

bool g (const double x) {
   return x % 2.0 == 0;
}

void h () {
   std::cout << result;
}

int main () {
   const double result = f(3.0);
   h();

   return 0;
}
```

Exercise 2

Problem 1: g() not yet known

scope of g starts later

```
# include <iostream>

double f (const double x) {
   return g(2.0 * x);
}

bool g (const double x) {
   return x % 2.0 == 0;
}

void h () {
   std::cout << result;
}

int main () {
   const double result = f(3.0);
   h();

   return 0;
}
```

Exercise 2

Problem 1: `g()` not yet known

scope of `g` starts later

```
# include <iostream>

double f (const double x) {
   return g(2.0 * x);
}

bool g (const double x) {
   return x % 2.0 == 0;
}

void h () {
   std::cout << result;
}

int main () {
   const double result = f(3.0);
   h();

   return 0;
}
```

Problem 2: Modulo

no modulo for `double`

Exercise 2

Problem 1: `g()` not yet known

scope of `g` starts later

```
# include <iostream>

double f (const double x) {
   return g(2.0 * x);
}

bool g (const double x) {
   return x % 2.0 == 0;
}

void h () {
   std::cout << result;
}

int main () {
   const double result = f(3.0);
   h();

   return 0;
}
```

Problem 2: Modulo

no modulo for `double`

Problem 3: `h()` does not «see» `result`

`result` is out-of-scope

Exercise 3

Exercise 3

- What is the **return value** of this program?
- You can neglect possible over- or underflows for this exercise.

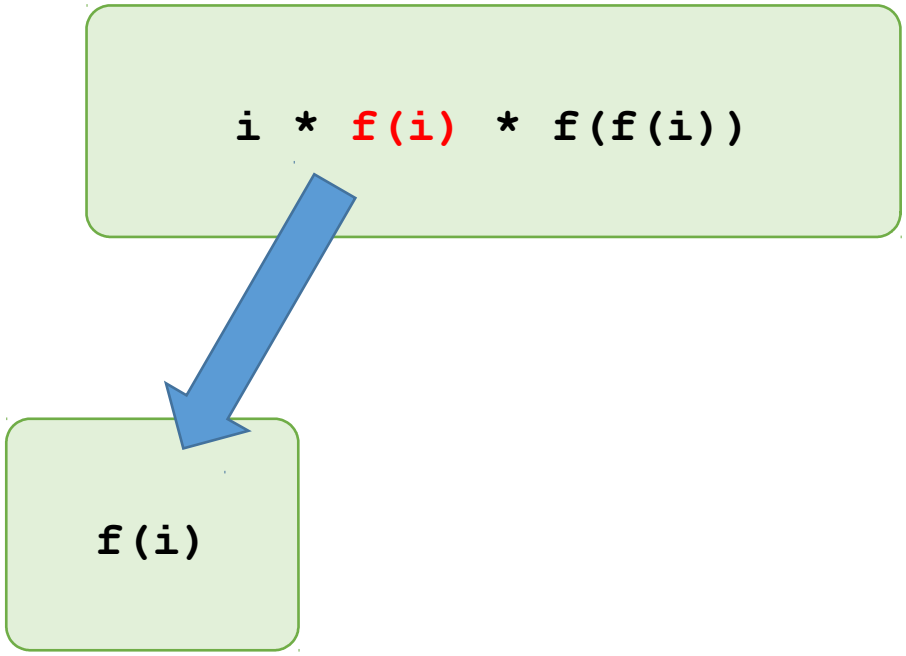
```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3

```
i * f(i) * f(f(i))
```

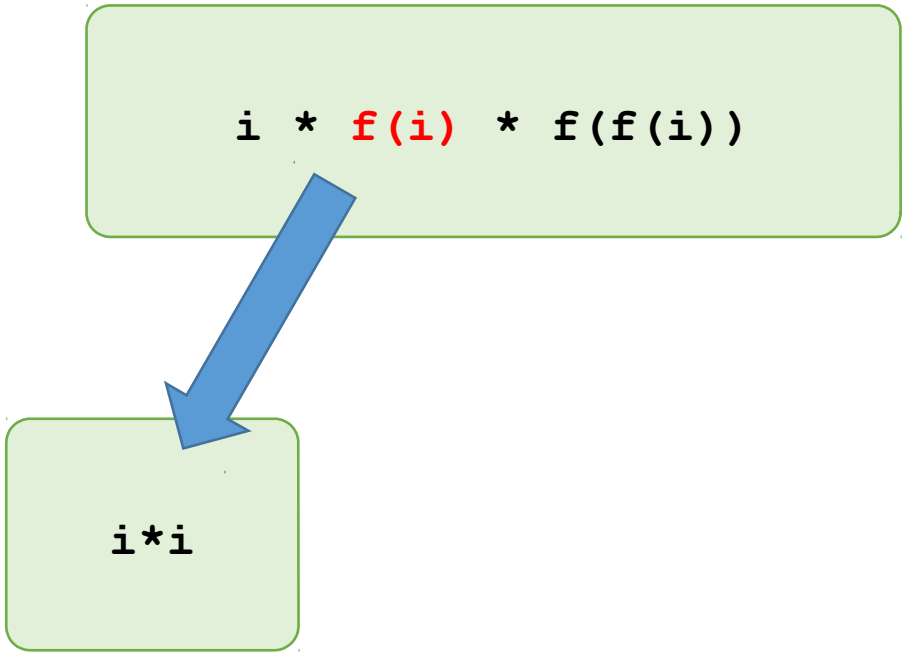
```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



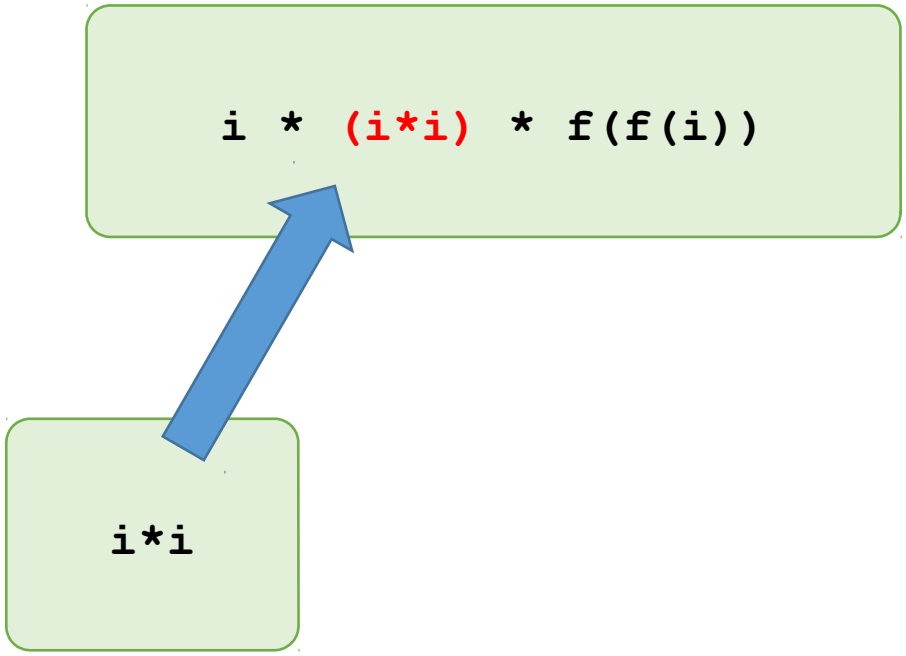

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



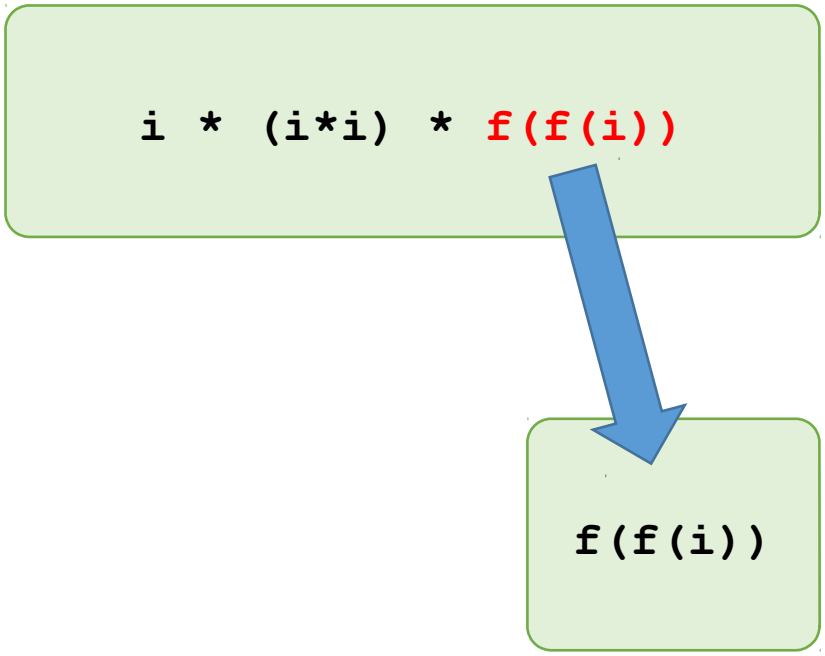

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



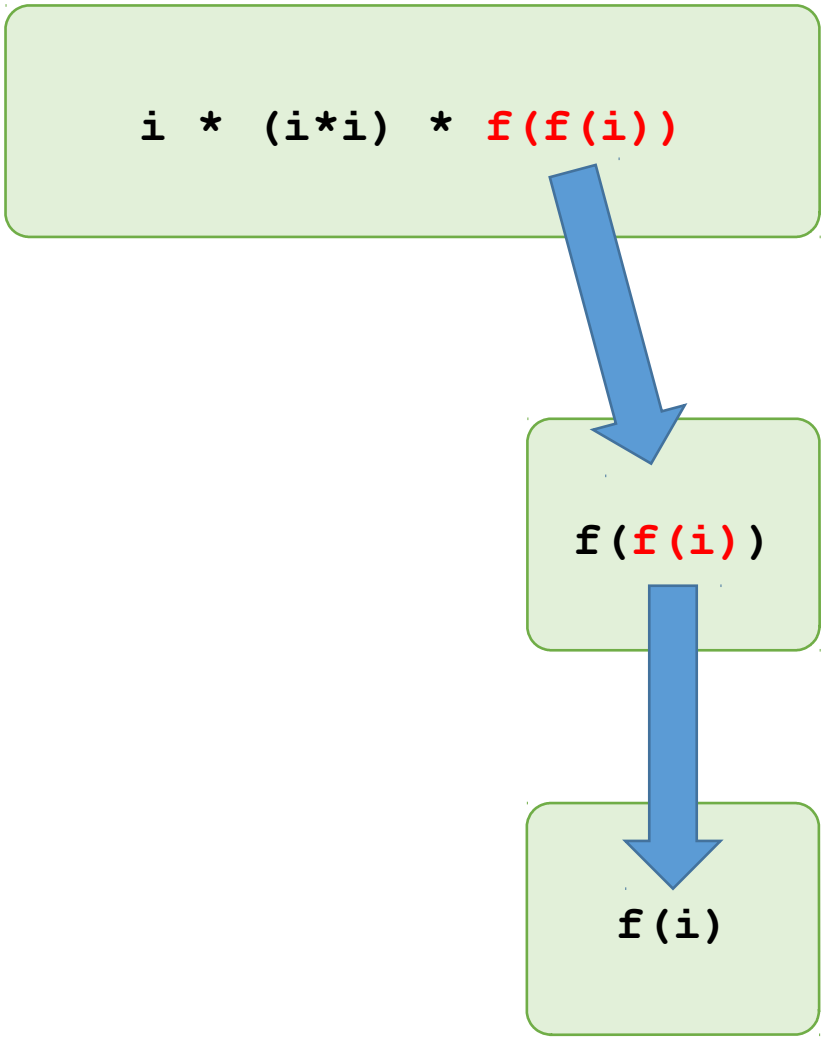

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



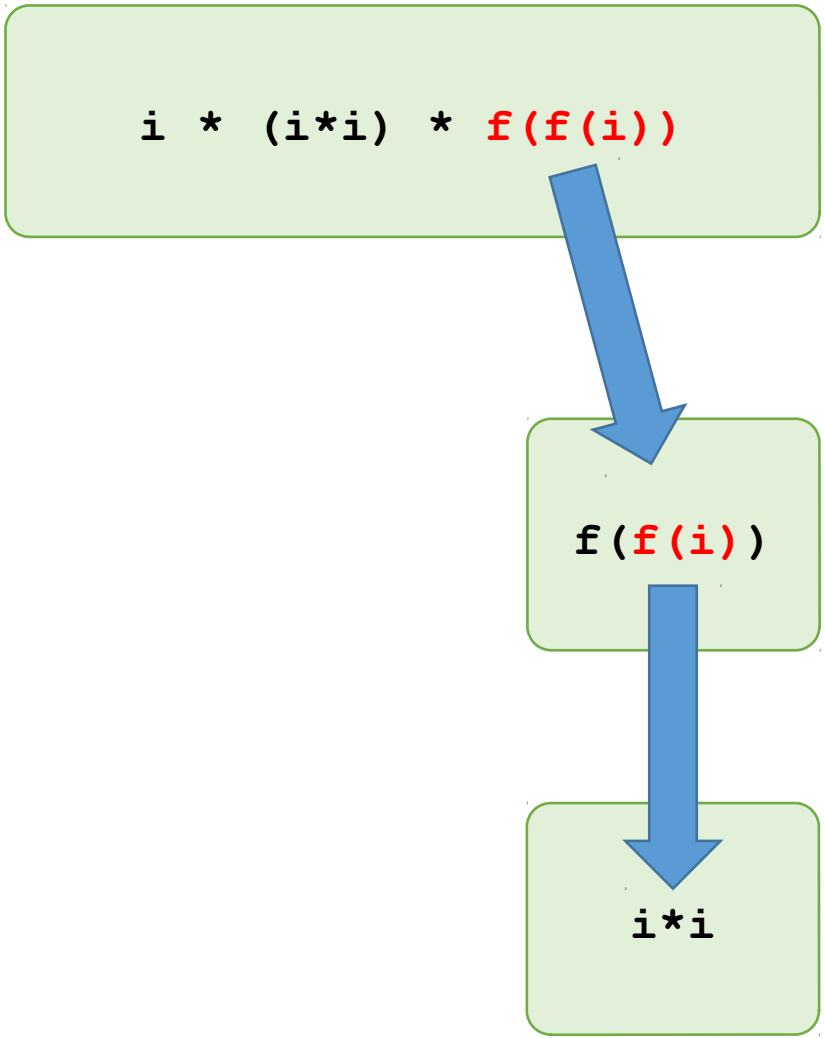

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



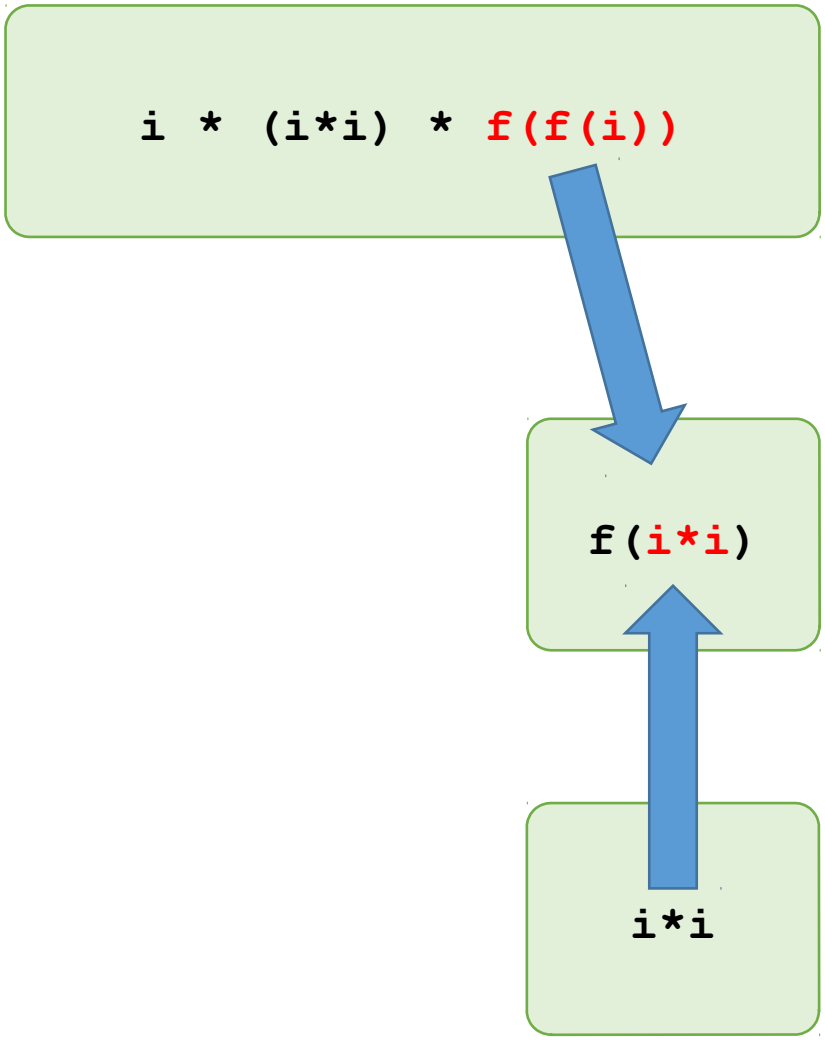

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



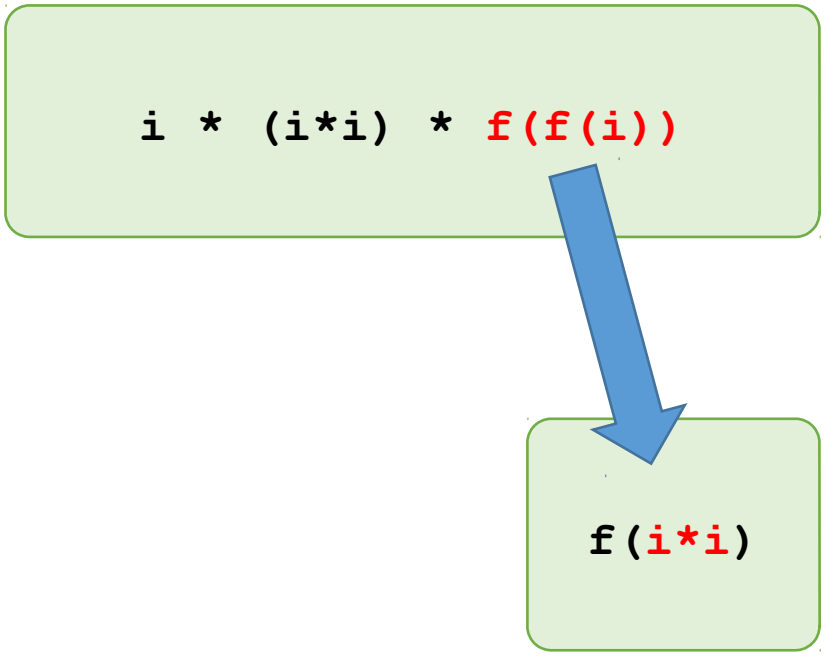

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3



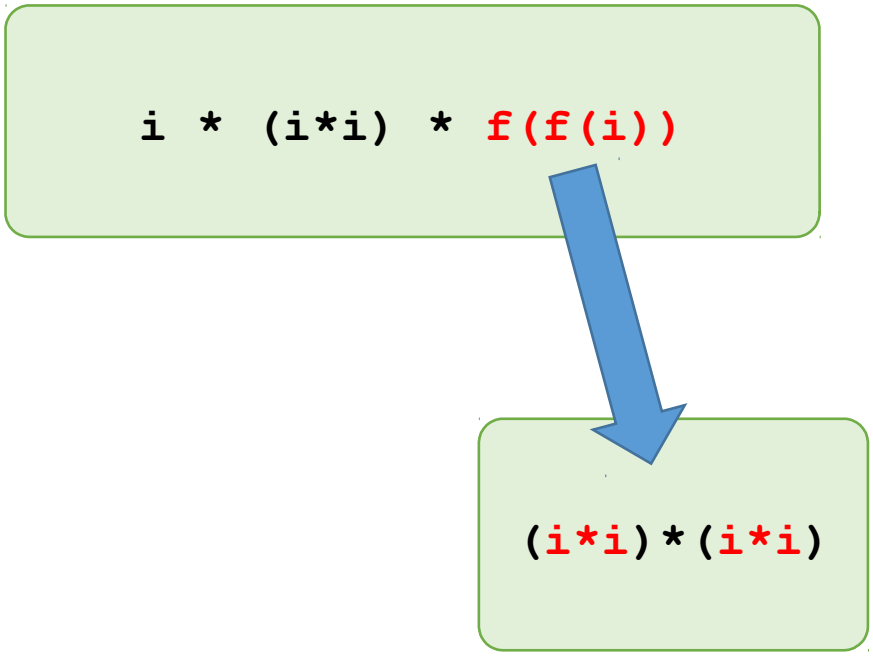

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3




```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

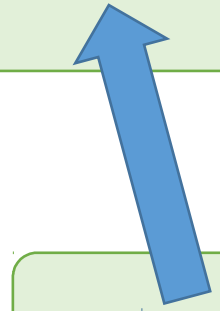
Exercise 3




```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3

```
i * (i*i) * ((i*i)*(i*i))
```



```
(i*i)*(i*i)
```

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

(From: Script Exercise 81)

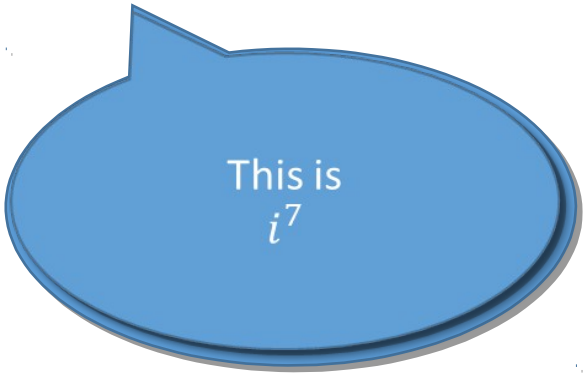
Exercise 3

```
i * (i*i) * ((i*i)*(i*i))
```

```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

Exercise 3

```
i * (i*i) * ((i*i)*(i*i))
```




```
#include <iostream>

int f (const int i) {
     return i * i;
}

int g (const int i) {
     return i * f(i) * f(f(i));
}

void h (const int i) {
     std::cout << g(i) << "\n";
}

int main () {
     int i;
     std::cin >> i;
     h(i);
     return 0;
}
```

(From: Script Exercise 81)