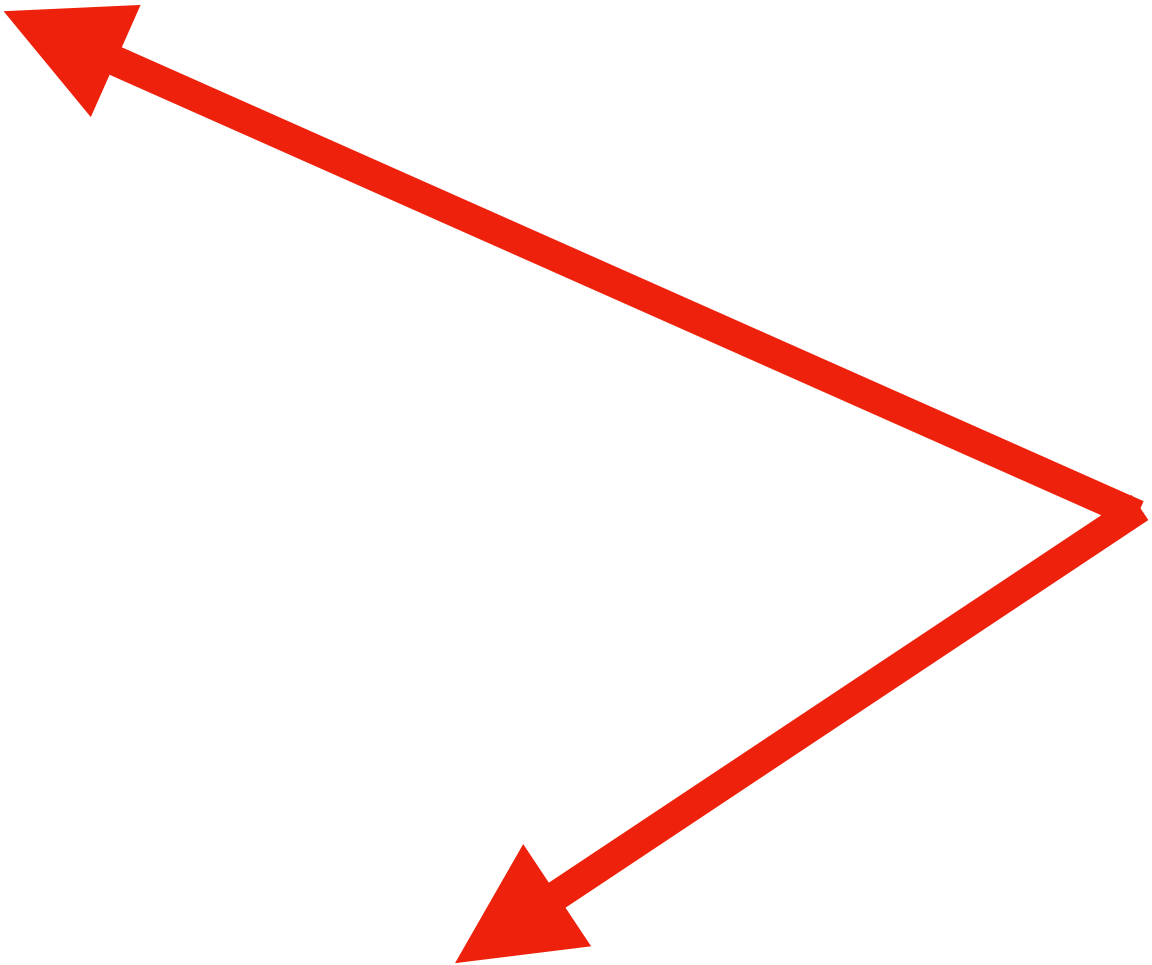
```
auto gain = 1.0f;

void realtimeThreadEntry()
{
    register auto gain_copy = gain;
    while (rocketFlying)
    {
        // do some dsp ...

        for (int i = 0; i < n; ++i)
           sensorInOut[i] *= gain_copy;
    }
}

// called on another thread
void setSensorGain (float newGain)
{
    gain = newGain;
}
```



Cached

No effect

Data race

Undefined behaviour

Anything can happen!
(Including exploding rockets)

40

```
auto gain = 1.0f;

void realtimeThreadEntry()
{
    register auto gain_copy = gain;
    while (rocketFlying)
    {
        // do some dsp ...

        for (int i = 0; i < n; ++i)
           sensorInOut[i] *= gain_copy;
    }
}

// called on another thread
void setSensorGain (float newGain)
{
    gain = newGain;
}
```

**Cached**

**No effect**

Data race

↓

Undefined behaviour

↓

Anything can happen!
(Including exploding rockets)

```
bool threadRunning;

bool proveFermatsLastTheorem() // Thread 1 {
    threadRunning = true;
    for (int n = 3; threadRunning; ++n) {
        if (pow (x, n) + pow (y, n) == pow (z, n)) {
            return false;
        }
    }

    return true;
}

void testTheorem () {
    bool result;
    startThread ([] () (result = proveFermatsLastTheorem));
    Sleep (2000);
    threadRunning = false;
    std::cout << result << std::endl;
}
```