```
1   //////// Lecture 71: Raw Pointers
2
3   // Discuss the smart pointers provided by the C++
      standard library, these smart pointers are based on
      the concept of resource acquisition as
      initialisation. c++ 11 PROVIDES 2 SMART POINTERS,
      and we'll see how to use these one by one.
4
5   class Integer {
6   ...
7   };
8
9   Integer* GetPointer(){
10      Integer* ptr = new Integer {};
11      return p;
12  }
13  void Display(Integer *p)
14  {
15      if (!p){return;}
16      std::cout << p->Getvalue() << std::endl;
17  }
18  void Operate(int value)
19  {
20      Integer *p = GetPointer();
21      if (p == nullptr) {
22          p = new Integer{};
23      }
24      p->SetValue(100);
25      Display(p);
26      delete p;
27      p = nullptr; // to avoid dangling pointer.
28      p = new Integer{};
29      *p = __LINE__; // C-macro that expands into a line
          number.
30      Display(p);
31      delete p;
32  }
33  // Note that the number of constructo calls match with
      the number of destructor calls, so this means that
      there are no memory leaks.
34  // but the code becomes really hard to read.
35
```