Design by Contract

- The contract specifies *preconditions* , *postconditions* and *invariants*, which participating components must honor.
- E.g. NDEGUG
  - Not defined in Debug. So asserts are checked.
  - Defined in Build. So asserts are ignored.


```
#include <cassert>
using namespace std;
namespace feb05 {
		const double PI = 22.0 / 7.0;
		double computeArea(double radius) {
				assert(radius != 0); //Pre-condition
				assert(PI > 3.1 && PI < 3.2); //Invariant
				auto r = PI * radius * radius;
				assert(r != 0); //Post-condition
				return r;
		}
}
using namespace feb05;
void feb05main() {
		computeArea(3);
}
```

Guidelines

- DRY

7-Feb-23 5:23 PM https://vijaynathani.github.io/ 2

Q021: BookRentals; Q11: BookRental; Q33 – SurveyData; Q37 – FOC, TT

Q10 – Jbutton – Only for Java.

Q34 – replace; Q35 – BookRental

Q70 – Array Scan, Java only.

There are three numbers in software: 0, 1, and infinity. 0 represents the things we do not do in a system (we do those for free). 1 represents the things we do once and only once. But at the moment we do something twice, we should treat it as infinitely many and create cohesive services that allow it to be reused.