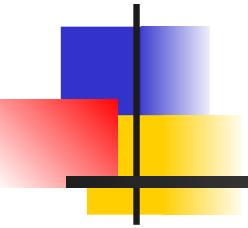

Assertions

"Programming by Contract"

Assertions

- **Assertions** are tests that should always be true at a given point in a program.
- Assertions help verify program correctness during development.
- If an assertion is false, an exception is raised.
- After the program is completed, assertions can be disabled using a compiler option, so they do not effect "production" code.

Use of Assertions

Pre-conditions: conditions that should always be true when the method is invoked

Post-conditions: conditions that should be true when the method returns

Example: when we play a Game, the game should not be null.

```
public int play(Game game) {
      // game should not be null!
      assert game != null : "game is
null";
```

"assert" versus throw AssertionError

- "assert" will throw an AssertionError.

```
// game should not be null!
assert game != null : "game is null";
```

- can we write this? is it equivalent?

```
// game should not be null!
if ( game == null )
    throw new AssertionError(
          "game is null");
```

(Answer is no. You can disable 1st code using compiler, but not the second.)

assert versus IllegalArgumentException

- If a parameter value is invalid, you could also throw exception:

```
public int play(Game game) {
      if (game == null) throw
            new
IllegalArgumentException("...");
```

For methods that are part of a public API (which can be called by other applications) throwing exception is better.

"assert" in other languages

- how to emulate assertions in C:

```
/* myheader.h */

#define DEBUG 1  /* 0 for production version */
```

```
#include <myheader.h>


#if DEBUG
        if ( fromStack == null )
        fprintf(stderr, "fromStack is null");
#endif
```