IO monads

```
mainIO = hello `seqIO` hello
       where
         hello = putsIO "hello"
```

hello is an effect and we can now do it twice
without breaking referential transparency

IO monads

In general, an action may also return a value. Again, there are two combinators. The first is again trivial:

```
unitIO :: a -> IO a
```

Lift a pure A to IO[A]

If `x` is of type `a`, then `unitIO x` denotes the action that, when performed, does nothing save return `x`. The second combines two actions:

```
bindIO :: IO a -> (a -> IO b) -> IO b
```

flatMap or bind

sequences two effects together