have it reference another object. We can also compare `point1` and `point3` and determine if they are referencing the same object. The other category of operations applies to the object being referenced; perhaps we could examine or change the internal state of one of the `Point` objects. For instance, we could examine some of `Point`'s *x* and *y* coordinates.

Before we describe what can be done with references, let us see what is not allowed. Consider the expression `point1*point2`. Since the stored values of `point1` and `point2` are 1000 and 1024, respectively, their product would be 1024000. However, this is a meaningless calculation that could not have any possible use. Reference variables store addresses, and there is no logical meaning that can be associated with multiplying two addresses.

Similarly, `point1++` has no Java meaning; it suggests that `point1` — 1000 — should be increased to 1001, but in that case it would not be referencing a valid `Point` object. Many languages define the *pointer*, which behaves like a reference variable. However, pointers in C++ are much more dangerous because arithmetic on stored addresses is allowed. Thus, in C++, `point1++` has a meaning. Because C++ allows pointers to primitive types, one must be careful to distinguish between arithmetic on addresses and arithmetic on the objects being referenced. This is done by explicitly *dereferencing* the pointer. In practice, C++'s unsafe pointers tend to cause numerous programming errors.

Some operations are performed on references themselves, while others are performed on the objects being referenced. In Java, the only operators that are allowed for reference types (with one exception made for `String`s) are assignment via `=` and equality comparison via `==` or `!=`.

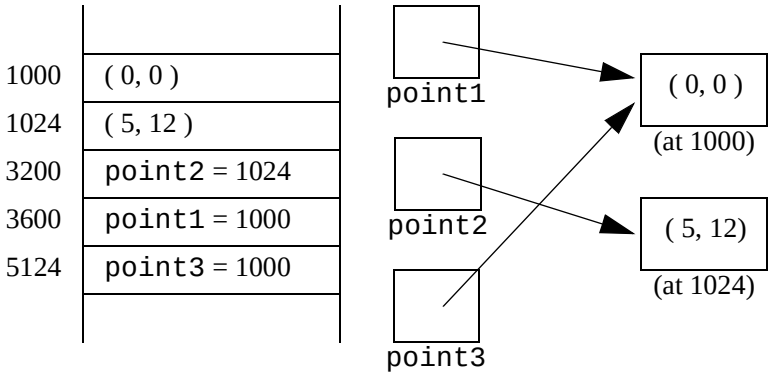

Figure 2.1 An illustration of a reference: The `Point` object stored at memory location 1000 is referenced by both `point1` and `point3`. The `Point` object stored at memory location 1024 is referenced by `point2`. The memory locations where the variables are stored are arbitrary