# NEXTSTEP

**Title:** NSDate comparisons are incorrect

**Entry Number:**
**Last Updated:** Mar 15 1995
**Document Revision:** 0395A

**Keywords:** NSDate

## Question

Q: In EOF 1.0, why do two identical NSDates compare as NSOrderedAscending instead of NSOrderedSame?

Q: Why does an array of NSDates not sort properly when there are one second differences between some of the elements?

## Answer

In EOF 1.0 and EOF 1.1, the isEqual: and compare: methods in NSDate were implemented to use a one second delta. In EOF 1.1, the behavior of these methods was improved, but remains incorrect for dates differing by less than one second.

The following category implements the correct behavior:

```
@implementation NSDate (Comparisons)
- (NSComparisonResult)compare:(NSDate *)anotherDate {
```

```
    NSTimeInterval delay = [self timeIntervalSinceDate:anotherDate];

    if (delay < 0.0) return NSOrderedAscending;
    if (delay > 0.0) return NSOrderedDescending;
    return NSOrderedSame;
}

- (BOOL)isEqual:(id)other {
    if (other == self)
       return YES;
    if (! [other isKindOfClass:[NSDate class]])
        return NO;
    return  [self timeIntervalSinceReferenceDate] == [other
timeIntervalSinceReferenceDate];
}

- (unsigned)hash {
    /* the invariant: [x isEqual:y] => [x hash]==[y hash] must hold */
    // use int to get a neg number, then xform into unsigned
    return (int)[self timeIntervalSinceReferenceDate];
}
@end
```

## Valid For EOF 1.0 and EOF 1.1

**See Also:**