Worksheet: Futures

Here is a recursive, sequential divide-and-conquer function for finding a maximum value in an array:

```
static int findMax(int[] X, int lo, int hi) {
    if ( lo > hi ) return 0;
    else if ( lo == hi ) return X[lo];
    else {
        int mid = (lo+hi)/2;
        var max1 =
                findMax(X, lo, mid);
        var max2 =
                findMax(X, mid+1, hi);

        return (max1 > max2)? max1 : max2;
    }
} // findMax
```

Indicate in the code the changes you need to make to this function in order to create a parallel, recursive divide-and-conquer function for finding a maximum value in an array. Are there any Future.get() operations that are guaranteed to be non-blocking?

Worksheet: Futures

Here is a recursive, sequential divide-and-conquer function for finding a maximum value in an array:

```
static int findMax(int[] X, int lo, int hi) throws SuspendableException {
    if ( lo > hi ) return 0;
    else if ( lo == hi ) return X[lo];
    else {
        int mid = (lo+hi)/2;
        var max1 = future(() ->
                findMax(X, lo, mid));
        var max2 = future(() ->
                findMax(X, mid+1, hi));
        // Parent now waits for the future values
        return (max1.get() > max2.get())? max1.get() : max2.get();
    }
} // findMax
```

Indicate in the code the changes you need to make to this function in order to create a parallel, recursive divide-and-conquer function for finding a maximum value in an array. Are there any Future.get() operations that are guaranteed to be non-blocking?

Yes, the second max1.get() and max2.get().