Middle of Three

Given three distinct numbers A, B and C. Find the number with value in middle (Try to do it with minimum comparisons).

```
def middle(self,A,B,C):
        if A<B:
            return B if B<C else max(A,C)
        return A if A<C else max(B,C)
```

Maximum and minimum of an array using minimum number of comparisons

Pair MaxMin(array, array_size)
if array_size = 1
return element as both max and min
else if arry_size = 2
one comparison to determine max and min
return that pair
else /* array_size > 2 */
recur for max and min of left half
recur for max and min of right half
one comparison determines true max of the two candidates
one comparison determines true min of the two candidates
return the pair of max and min

```
def getMinMax(low, high, arr):
    arr_max = arr[low]
    arr_min = arr[low]

    # If there is only one element
    if low == high:
        arr_max = arr[low]
        arr_min = arr[low]
        return (arr_max, arr_min)

    # If there is only two element
    elif high == low + 1:
        if arr[low] > arr[high]:
            arr_max = arr[low]
            arr_min = arr[high]
        else:
            arr_max = arr[high]
            arr_min = arr[low]
        return (arr_max, arr_min)
    else:

        # If there are more than 2 elements
        mid = int((low + high) / 2)
        arr_max1, arr_min1 = getMinMax(low, mid, arr)
        arr_max2, arr_min2 = getMinMax(mid + 1, high, arr)

    return (max(arr_max1, arr_max2), min(arr_min1, arr_min2))

# Driver code
arr = [1000, 11, 445, 1, 330, 3000]
high = len(arr) - 1
low = 0
arr_max, arr_min = getMinMax(low, high, arr)
print('Minimum element is ', arr_min)
print('nMaximum element is ', arr_max)
```