1. Write an alternate quicksort algorithm as discussed in class. The partitioning will be with the help of two pointers l, r. l moves left to right and r moves right to left from the end. And everything that is to the left of l is less than the pivot, and everything to the right of r is greater than the pivot.

2. Here is an alternative sorting alorithm called Pancake sorting which uses only one operation `flip(arr, i)` which flips the elements $0 \cdots i$. For example `flip([1,2,3,4,5],2)` returns `[3,2,1,4,5]`. The sorting algorithm is as follows

```
pancake_sort(array, n)
    cur_size = n
    while cur_size >1
        max_index = find_max(arr[:cur_size])
        if max_index != (cur_size - 1)
            flip(arr, max_index)
            flip(arr, cur_size-1)
        cur_size = cur_size - 1
```

   Code up the algorithm and the functions i.e. `flip, find_max` and `pancake_sort`

3. Write a function `product_map` which when given an array of elements `arr` output a new list `out` such that, `out[i]` is the product of the first largest, second largest and third largest elements among `arr[0...i]`. If there is less than 3 elements output -1. For example :

   `product_map([1,2,3,4,5]) = [-1,-1,6,24,60]`

   Hint: Use the heap implementation.

4. Implement a class `Matrix` which represents a two dimensional matrix.

   (a) The constructor will recieve an array arr, two integer values n and m. Where n represents the number of rows and m represents the number of columns. arr contains the elements left to right of a row and rows are ordered top to bottom.
   For example
   $$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$
   the matrix will be instantiated as follows

```
m = Matrix([1,2,3,4,5,6], 2, 3)
```

   (b) Implement a member function `row` which takes the index of the row and returns the elements of that row.

   (c) Implement a member function `column` which takes the index of the column and returns the elements of that column.

(d) Implement a member function `transpose` which transposes (interchanges rows and columns) the current matrix and changes the dimensions respectively. For example, transpose of

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

will be

$$M = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$