# Power Round

Names: _____

Team Name: _____

## Instructions

1. Do not begin until instructed to by the proctor.
2. You will have 90 minutes to solve the problems during this round.
3. Your submission will be graded and assigned point values out of the total points possible per problem. Your total score will be the sum of the points you receive for each problem.
4. Submissions will be graded on correctness as well as clarity of proof. A proof with significant progress towards a solution may receive more credit than a correct answer with no justification.
5. **You may use the result of a previous problem in the proof of a later problem, even if you do not submit a correct solution to the referenced problem.** However, you may not use the result of a later problem in the proof of an earlier problem.
6. Please submit each part of each problem on a separate page. Write your team name, problem number, and page number clearly at the top of each page.
7. No calculators or electronic devices are allowed.
8. All submitted work must be the work of your own team. You may collaborate with your team members, but no one else.
9. When time is called, please put your pencil down and hold your paper in the air. **Do not continue to write.** If you continue writing, your score may be disqualified.
10. Do not discuss the problems with anyone outside of your team until all papers have been collected.
11. If you have a question or need to leave the room for any reason, please raise your hand quietly.
12. Good luck!

## Acceptable Answers

1. Solutions should be written in proof format. All answers, reasoning, and deductions must be explained and justified, unless the problem explicitly asks for you to "compute". Problems asking you to "show", "prove", or "justify" **require proof!**
2. Proofs will be graded both on correctness as well as clarity of presentation.
3. Partial credit may be awarded for significant progress towards a solution.
4. Each problem must be written starting on a new, blank page. Two different problems should not be written on the same page.
5. At the top right corner of each page, please clearly print your team name, problem number, and page number.
6. Answers must be written legibly to receive credit. Ambiguous answers may be marked incorrect, even if one of the possible interpretations is correct.

Let $[n]$ be the set $\{1, 2, \ldots, n\}$, or the set of positive integers between 1 and $n$. A *permutation* $p = \{p_1, p_2, \ldots, p_n\}$ of $[n]$ is a reordering of $[n]$ such that such that each $p_k$ is an element of $[n]$ for all $k$ and each element of $[n]$ occurs in exactly one position of $p$. For instance, $\{2, 4, 3, 1, 5\}$ is a permutation of $[5]$.

**Problem 1.** Let $p$ be a permutation of $[n]$. Define the number of *inversions* of $p$ as the number of pairs of indices $(i, j)$ such that $i < j$ and $p_i > p_j$.

(a) (1 point) Compute the number of inversions in the permutation $\{3, 1, 6, 2, 5, 4\}$.

(b) (2 points) What is the maximal number of inversions in a permutation of $[n]$? Prove your answer and describe a permutation which achieves that maximum number.

(c) (1 point) For $i = 0, 1, 2, 3, 4, 5, 6$, find a permutation of $[4]$ with $i$ inversions.

(d) (2 points) Let $M_n$ be the maximum number of inversions achievable by a permutation of $[n]$. For all $j$ between 0 and $M_n$, does there always exist a permutation $p_j$ of $[n]$ such that $p_j$ has $j$ inversions?

**Solution:**

(a) 6.

(b) The total number of ordered pairs of distinct indices is $\binom{n}{2}$, which upper bounds the number of inversions. The permutation $\{n, n-1, \ldots, 3, 2, 1\}$ achieves this upper bound, since for each $i < j$, $p_i > p_j$ since the permutation is in reverse order.

(c) Varies.

(d) The answer is yes. Proceed by induction. The identity permutation has 0 swaps; now, assuming there exists a permutation with $k$ swaps with $k < \binom{n}{2}$, that permutation is not in reverse order so we can find adjacent indices such that $p_i < p_{i+1}$. Swapping them creates a permutation with one more inversion.

**Problem 2.** Let $p$ be a permutation of $[n]$. Every second, the following operation is performed on $p$: Randomly pick $1 \leq i < j \leq n$ such that $p_i > p_j$ and swap $p_i$ with $p_j$. If no such $i, j$ exist, do nothing.

(a) (2 points) Show that after a finite number of seconds, the operation will no longer change $p$.

(b) (2 points) Find and prove the maximum number of seconds it can take for the operation to stop changing a permutation $p$.

**Solution:**

(a) We claim that each swap reduces the number of inversions by at least 1. It does not affect indices less than $i$ and greater than $j$; for indices $k$ such that $i < k < j$, if $p_j < p_k < p_i$ then the number of inversions is reduced, otherwise it is not affected. Since the inversion $(i, j)$ is removed, the number of reductions is at least one. Since the number of inversions is finite to begin with, the process will eventually end.

(b) Starting from the permutation $\{n, n-1, n-2, \ldots, 2, 1\}$, if at each turn two adjacent elements that are in the wrong order are swapped, then the number of inversions reduces by 1. It takes $\binom{n}{2}$ seconds for the operation to stop changing $p$.

**Problem 3.** Let $p, p'$ be two distinct permutations of $[n]$.

(a) (2 points) Suppose we can modify $p$ by choosing $1 \leq i < n$ and swapping $p_i$ with $p_{i+1}$. Is it always possible to reach any $p'$ by repeatedly modifying $p$? Justify your answer.

(b) (2 points) Would it always be possible if the operation were instead to choose $i$ such that $1 \leq i < n-1$, and then swapping $p_i$ with $p_{i+2}$? Justify your answer.

**Solution:**

(a) Yes, it is possible. Define a function $f$ mapping $p'_k \mapsto k$, and repeatedly apply the operation from problem 2 in terms of inversions on $f(p_k)$. The process stops when $\{f(p_1), f(p_2), \ldots, f(p_n)\} = \{1, 2, \ldots, n\}$, or $p = \{p'_1, p'_2, \ldots, p'_n\} = p'$.

(b) No, it wouldn't be. Suppose $p = \{1, 3, 2\}$. The operation only permits swapping 1 and 2, which keeps the 3 out of order. For the more general case, see problem 5a.

**Problem 4** (4 points)**.** Call permutations $p, q$ of $[n]$ *swap-adjacent* if $p$ can be turned into $q$ by swapping any two elements. Show that there exists an ordering $p_1, p_2, \ldots, p_{n!}$ of the $n!$ distinct permutations of $[n]$ such that $p_i$ is swap-adjacent with $p_{i+1}$ for all $1 \leq i < n!$. Note that in this problem, $p_k$ is a permutation with index $k$, and not an element of some permutation.

**Solution:** Proceed by induction. Clearly $n = 1$ works; now suppose such an ordering exists for permutations of $[n]$. We show that an ordering exists for permutations of $[n+1]$. Case on the last element of the permutation. Starting with the identity, keeping $n+1$ in place we can generate all permutations of the first $n$ elements through swaps. Then, swap $n+1$ with $n$ and repeat the process, each time after going through all permutations of the first $n$ elements swapping the last element with something that was never there previously. This generates the union of all permutations which ends in a specific number, which is just all permutations, visiting each one exactly once.

**Problem 5.** Let $n$ and $k$ be positive integers with $n > k$, and suppose $p$ is a permutation of $[n]$. A *$k$-cyclic-shift* is an operation on $p$ that chooses some contiguous sublist of size $k$ from $p$ and shifts it to the right while wrapping the rightmost element around; formally, if $p = \{p_1, p_2, \ldots p_n\}$, then for some $1 \leq i \leq n-k+1$, the updated permutation $p'$ under a *$k$-cyclic-shift* positioned at $i$ would be

$$p' = \{p_1, p_2, \ldots, p_{i-1}, \underbrace{p_{i+k-1}, p_i, p_{i+1}, \ldots, p_{i+k-2}}_{\text{shifted sublist}}, p_{i+k}, p_{i+k+1}, \ldots, p_n\}.$$

For parts (a) and (b), assume $k$ is odd and $k \geq 3$.

(a) (5 points) Which permutations $p$ can be transformed to the identity permutation $\{1, 2, \ldots, n\}$ through a series of *$k$-cyclic-shifts*?

(b) (3 points) Specify and prove a minimal set of permutations $P$ such that any permutation of size $n$ can be reached from $P$ through a series of *$k$-cyclic-shifts*.

(c) (5 points) Suppose $k$ is even and $k \geq 4$. Answer questions (a) and (b) given these new constraints, and justify your answers.

**Solution:**

(a) Let $I$ be the number of inversions of $p$; We claim that $p$ can be transformed to the identity permutation if and only if $I$ is even, and that a minimal set of permutations $P = \{(1, 2, 3, \ldots, n), (2, 1, 3, \ldots, n)\}$ satisfies the required condition.

First, note that swapping two consecutive elements in $p$ increases or decreases $I$ by exactly 1, and since a *k-cyclic-shift* consists of $k-1$ swaps of adjacent elements that bring the rightmost element in the shifted sublist to the left, $I$ changes by an even amount. Thus, since the identity permutation has no inversions, $p$ can only be transformed to the identity permutation if $I$ is even.

Now if $k > 3$, through a series of *k-cyclic-shifts*, it is possible to effectively perform a *3-cyclic-shift*: consider any sublist of $p$ of size $k+1$ and assume WLOG it is the first $k+1$ elements. If we apply a *k-cyclic-shift* to the first $k$-sublist and then to the second, we will get $(k, k+1, 1, 2, \ldots, k-1)$, which is the same as cyclically shifting the $k+1$ elements by 2. Noting that $k$ is odd, we can then get from any permutation of the $k+1$ elements to another as long as they are cyclic shifts of one another by an even amount. Thus, noting the same simplification for our permutations of size $k$ allowing us to identify it with any cyclically shifted permutation, we perform the following transformations:

$$(2, 3, 1, 4, \ldots, k+1) \mapsto (1, 4, \ldots, k+1, 2, 3) \mapsto (1, 2, 3, 4, \ldots, k+1) \tag{1}$$

$$(1, 4, 2, 3, 5, \ldots, k+1) \mapsto (1, 2, 3, 5, \ldots, k+1, 4) \mapsto (3, 5, \ldots, k+1, 4, 1, 2)$$
$$\mapsto (3, 4, 1, 2, 5, \ldots, k+1) \mapsto (1, 2, 5, \ldots, k+1, 3, 4) \mapsto (1, 2, 3, 4, 5, \ldots, k+1) \tag{2}$$

Thus, it is possible to perform a *3-cyclic-shift* on the first two sublists of size 3 due to (1) and (2) respectively. However, since we get from any permutation of the $k+1$ elements to another that are cyclic shifts of one another by an even amount, it is possible to perform a *3-cyclic-shift* on any sublist of size 3. As our choice of the $k+1$ sublist was arbitrary, we can perform *3-cyclic-shift* on any sublist of size 3 in $p$ through a series of *k-cyclic shifts*.

On the other hand, we can perform a *k-cyclic-shift* through a series of *3-cyclic-shifts*: as a *k-cyclic-shift* moves the rightmost element in the chosen sublist to $k-1$ positions to the left while leaving the rest of the elements the same, performing $\frac{k-1}{2}$ *3-cyclic-shifts* to move the rightmost element to the leftmost position in a $k$-sublist effectively performs a $k$-cyclic-shift. Thus, as *3-cyclic-shifts* can be represented as a series of *k-cyclic-shifts* and vice versa, it suffices to solve the problem using *3-cyclic-shifts*.

(b) Now, note that one can initially transform $p$ to be one of the permutations in $P$ by iterating from $i = n, \ldots, 3$ and moving $i$ in $p$ to position $i$. Since the two permutations in $P$ have inversion counts of 0 and 1, if $I$ was initially even, then it would be transformed to the identity permutation. Finally, $P$ must be a minimal set of permutations since it is of size 2 and not all permutations can be transformed into the identity permutation, such as the other permutation in $P$ whose inversion count is odd.

(c) All permutations $p$ can in fact be transformed into the identity permutation, and so a minimal set $P$ would be $\{p\}$. We do this by showing that a swap of 2 adjacent elements can be represented as a series of *k-cyclic-shifts*, and it is clear that $p$ can be transformed to the identity permutation using any swap-based sorting algorithm such as Bubble Sort.

We proceed as in the case when $k$ is odd: choosing any sublist of $p$ of size $k+1$, again WLOG the leftmost one, applying a *k-cyclic-shift* to the first $k$-sublist and then to the second yields $(k, k+1, 1, 2, \ldots, k-1)$. Since we can effectively cyclically shift the $k+1$ sublist to the right by 2 and $k+1$ is odd, it is possible to cyclically shift the $k+1$ sublist by any integer amount. Thus, we can perform the following transformation:

$$(2,1,3,4,\ldots,k+1) \mapsto (1,3,4,\ldots,k+1,2) \mapsto (1,2,3,4,\ldots,k+1) \tag{3}$$

From the above, we can swap the first two elements in our sublist of $k+1$ elements, and since we can cyclically shift the $k+1$ sublist by any integer amount, we can swap any two adjacent elements in our sublist. As the chosen sublist was arbitrary, we can swap any two adjacent elements in $p$, and so we are done.

**Problem 6.** Let $n$ be positive integers and suppose $p$ is a permutation of $[n] = \{1, 2, \ldots, n\}$. Define a *$k$-cyclic-shift* as in problem 1 for $1 \leq k \leq n$. Suppose performing a *$k$-cyclic-shift* costs $C$ to perform. What is the minimum cost to sort $p$?

(a) (4 points) Solve the problem if $C = k$.

(b) (4 points) Solve the problem if $C = k^2$.

**Solution:**

(a) Note that a *$k$-cyclic-shift* corresponds to selecting some index $k \leq i \leq n$, and moving $p_i$ $k-1$ elements to the left while leaving the remaining list intact. Now, for $i = 1, \ldots, n$, let $c_i$ be the number of elements $i < j \leq n$ such that $j$ comes before $i$ in $p$. Since *$k$-cyclic-shifts* only move elements to the left, through some sequence of operations, element $i$ in $p$ must move at least $c_i$ to the left, and so the cost of the operations performed solely on moving element $i$ would be $\geq c_i + 1$ if $c_i \geq 1$ and 0 otherwise. Thus, the total cost is lower bounded by $(\sum_{i=1}^{n} c_i) + m$, where $m$ is the number of $c_i$'s which are nonzero.

We claim that this cost can be achieved: iterate from $i = 1, \ldots, n$ and move element $i$ to position $i$ in $p$ using a single $c_i + 1$-*cyclic-shift*, where a *1-cyclic-shift* corresponds to doing nothing and thus has an incurred cost of 0. At the $i$-th iteration, the only elements which could have been moved to the left of element $i$ are those less than $i$, which are sorted and at the front of the array. Thus, the elements greater than $i$ and to the left of it in $p$ are exactly the same and all directly to the left, and so performing a $c_i + 1$-*cyclic-shift* will put element $i$ in position $i$. Therefore, this procedure will terminate with a sorted list and with the desired total cost which is known to be minimal.

(b) It is always more optimal to use *2-cyclic-shifts* (or swaps) compared to *$k$-cyclic-shifts* for $k \geq 3$ because we can always perform the latter using $k-1$ swaps, which costs $4(k-1) < k^2$ and holds since $(k-2)^2 > 0$ due to $k \geq 3$. Thus, we limit our set of operations to that of swaps.

Let $I$ be the number of inversions in $p$; that is, the number of pairs $(i, j)$ with $1 \leq i < j \leq n$ such that $p_i > p_j$. Since each swap can only change $I$ by 1 (can only change whether the two participating elements in the swap have a pair in the inversion count), the total cost of sorting with only swaps is lower bounded by $4I$. However, this can be achieved as in the algorithm described in part (a), but we may consider a simpler algorithm: for $I$ iterations, find a pair of adjacent elements that are out of order and swap them. Such a pair must exist for $I$ iterations because the number of inversions in our current list would be nonzero implying that the list isn't sorted and thus must have a pair of elements out of order. Upon swapping those two, the inversion count would decrease by 1, and so after $I$ iterations, the list would be sorted and the total cost would be $4I$.

**Problem 7** (3 points)**.** Suppose $G$ is an $m \times n$ matrix whose entries are some permutation of $[mn] = \{1, 2, \ldots, mn\}$ where both $m$ and $n$ are positive integers. A *2-fix* on $G$ is an operation that selects some $2 \times 2$ subsquare of $G$ and sorts the elements by row and then by column; that is, if the elements in the subsquare

are $a < b < c < d$, then the resulting subsquare after sorting would be

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

When is it possible for $G$ to always become sorted by rows and then columns through a series of *2-fixes*?

**Solution:** We claim that it can be done always if and only if $n = 2$. Indeed, in this case, one can simply push the elements down into the right order from largest to smallest. For $n \geq 3$, consider the matrix $A$ constructed by starting with the sorted matrix and then swapping the bottom left corner with the rightmost element in the second to last row (that is, $(m-1)n$ and $(m-1)n+1$). Then no *2-fix* can modify $A$ since 1) the two elements don't share any $2 \times 2$ subsquare and don't have any integers between them, thus serving the same same role in all possible *2-fixes*, of which none are possible in the sorted matrix.

**Problem 8.** Suppose $G$ is an $m \times n$ matrix whose entries are some permutation of $[mn] = \{1, 2, \ldots, mn\}$ where both $m$ and $n$ are positive integers. A *2-rotation* on $G$ is an operation that selects some $2 \times 2$ subsquare of $G$ and rotates the elements clockwise. That is, we may transform

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \mapsto \begin{bmatrix} c & a \\ d & b \end{bmatrix}$$

(a) (4 points) Suppose $m = 2$ and $n = 3$. Specify and prove a minimal set of matrices $S$ such that any matrix $G$ can be transformed into some element in $S$ through a series of *2-rotations*.

(b) (4 points) Suppose $m = 2$ and $n \geq 4$. Prove that it is possible to reach any other matrix from $G$ through a series of *2-rotations*.

(c) (3 points) Suppose $m, n \geq 3$. Prove that it is possible to reach any other matrix from $G$ through a series of *2-rotations*.

**Solution:**

(a) We claim that $S$ consisting of $\begin{bmatrix} 1 & 2 & * \\ 4 & * & * \end{bmatrix}$ where the $*$'s take on all 6 permutations of $\{3, 5, 6\}$. For ease of notation, let $T = \begin{bmatrix} 1 & 2 & a \\ 4 & b & c \end{bmatrix}$.

First, note that any matrix $G$ can be transformed into an element of $S$ by first separating out the 1 and 4 to the left and right columns of $G$, recombining in the right order in the middle column, and then rotating them to be on the left column. Afterwards, rotate the right subsquare until the 2 is in the top left corner.

Now suppose it is possible to transform from one element $T \in S$ to another. If all starred elements are different, then they must be some rotation of one another, and so performing a *2-rotation* makes two of the starred elements the same. Thus, it is possible to effectively perform a swap between some two adjacent elements in $T$ (2 and either $a$ or $b$ in this case). If two adjacent starred elements are different while the other is the same, then it is again possible to effectively perform a swap between some two adjacent elements in $T$. Finally, if $T$ can be transformed so that $a$ and $b$ swap, then we may perform the following transformation to swap two adjacent elements in $T$:

$$\begin{bmatrix} 1 & 2 & a \\ 4 & b & c \end{bmatrix} \mapsto \begin{bmatrix} 2 & b & a \\ 1 & 4 & c \end{bmatrix} \mapsto \begin{bmatrix} 2 & b & 4 \\ 1 & a & c \end{bmatrix} \mapsto \begin{bmatrix} a & b & 4 \\ 1 & 2 & c \end{bmatrix} \mapsto \begin{bmatrix} a & b & 2 \\ 1 & 4 & c \end{bmatrix} \mapsto \begin{bmatrix} 1 & a & 2 \\ 4 & b & c \end{bmatrix}$$

Now, since we can perform *2-rotations*, it is possible to perform any swap between any two adjacent elements in $T$: simply *2-rotate*, apply the swap, and *2-rotate* backwards (apply 3 times), repeating this procedure until all possible swaps have been built from *2-rotations* along with the assumption that some two elements in $S$ are reachable from each other. Thus, it is possible to transform $G$ to any other matrix in all cases under the assumption.

However, not all $G$ are reachable from one another: consider the list of elements constructed from the first row of $G$ followed by the second and let $I$ be its number of inversions. Manually checking, we see that 4 pairs in the inversion count are inverted, and so $I$ always changes by an even amount. Since there exist matrices whose corresponding lists have inversion counts of 0 and 1 (e.g. sorted list and sorted list with one pair of adjacent elements swapped), not all matrices are reachable from one another. Thus, this contradicts our assumption above that some two elements in $S$ are reachable from one another, and so $S$ is a desired minimal set.

(b) We will show that through a series of *2-rotations*, it is possible to effectively perform a swap between some two adjacent elements. Then, as in the solution to part (a), we can extend this ability to all pairs of adjacent elements in $G$ and so that $G$ can ultimately reach any other matrix (one way to see this is to sort both $G$ and the desired matrix and combine the steps appropriately).

WLOG Suppose $n = 4$. Consider the following transformation:

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

Thus, one can swap two elements in a column and some pair of elements across the diagonal of an adjacent $2\times 2$ subsquare (we can do the flipped version by symmetry by flipping all operations about the vertical). Using this, we can swap rows in a $2\times 3$ matrix as follows:

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

In addition, we can swap rows in a $2\times 4$ matrix as follows:

$$\begin{bmatrix}5&6&7&8\\1&2&3&4\end{bmatrix}\mapsto\begin{bmatrix}1&6&7&4\\5&2&3&8\end{bmatrix}\mapsto\begin{bmatrix}2&5&8&3\\6&1&4&7\end{bmatrix}\mapsto\begin{bmatrix}6&5&8&7\\2&1&4&3\end{bmatrix}\mapsto\begin{bmatrix}1&2&3&4\\5&6&7&8\end{bmatrix}$$

Combining the above two operations, we can swap the elements in a given column as desired.

(c) We proceed in the same manner. Indeed, we can easily swap two adjacent elements as follows:

$$\begin{bmatrix}2&1&3\\4&5&6\\7&8&9\end{bmatrix}\mapsto\begin{bmatrix}1&5&3\\2&4&6\\7&8&9\end{bmatrix}\mapsto\begin{bmatrix}1&3&6\\2&5&4\\7&8&9\end{bmatrix}\mapsto\begin{bmatrix}1&3&6\\7&2&4\\8&5&9\end{bmatrix}\mapsto\begin{bmatrix}1&2&3\\7&4&6\\8&5&9\end{bmatrix}\mapsto\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$$