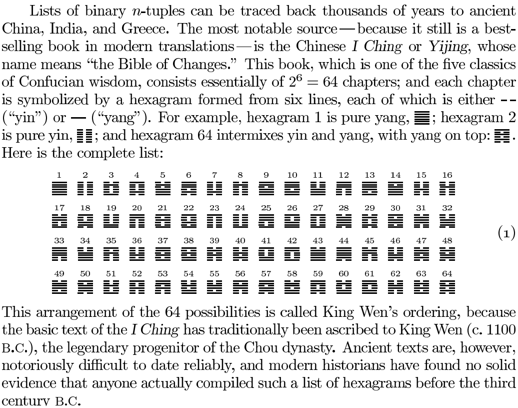
Generating Elementary Combinatorial Objects

Lucia Moura

Winter 2021

Combinatorial Generation: an old subject

Excerpt from: D. Knuth, *History of Combinatorial Generation*, in pre-fascicle 4B , The Art of Computer Programming Vol 4.



Lists of binary n-tuples can be traced back thousands of years to ancient China, India, and Greece. The most notable source — because it still is a best-selling book in modern translations — is the Chinese *I Ching* or *Yijing*, whose name means "the Bible of Changes." This book, which is one of the five classics of Confucian wisdom, consists essentially of $2^6 = 64$ chapters; and each chapter is symbolized by a hexagram formed from six lines, each of which is either -- ("yin") or — ("yang"). For example, hexagram 1 is pure yang, ䷀; hexagram 2 is pure yin, ䷁; and hexagram 64 intermixes yin and yang, with yang on top: ䷿. Here is the complete list:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
䷀	䷁	䷂	䷃	䷄	䷅	䷆	䷇	䷈	䷉	䷊	䷋	䷌	䷍	䷎	䷏
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
䷐	䷑	䷒	䷓	䷔	䷕	䷖	䷗	䷘	䷙	䷚	䷛	䷜	䷝	䷞	䷟
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
䷠	䷡	䷢	䷣	䷤	䷥	䷦	䷧	䷨	䷩	䷪	䷫	䷬	䷭	䷮	䷯
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
䷰	䷱	䷲	䷳	䷴	䷵	䷶	䷷	䷸	䷹	䷺	䷻	䷼	䷽	䷾	䷿

(1)

This arrangement of the 64 possibilities is called King Wen's ordering, because the basic text of the *I Ching* has traditionally been ascribed to King Wen (c. 1100 B.C.), the legendary progenitor of the Chou dynasty. Ancient texts are, however, notoriously difficult to date reliably, and modern historians have found no solid evidence that anyone actually compiled such a list of hexagrams before the third century B.C.

Combinatorial Generation

We are going to look at combinatorial generation of:

- Subsets
- k-subsets
- Permutations

To do a sequential generation, we need to impose some order on the set of objects we are generating.

Many types of ordering are possible; we will discuss two types: **lexicographical** ordering and **minimal change** ordering.

Combinatorial Generation (cont'd)

Let $\mathcal{S}$ be a finite set and $N = |\mathcal{S}|$.
A rank function is a bijection

$$\text{RANK}\colon \mathcal{S} \to \{0, 1, \dots, N-1\}.$$

Its inverse is another bijection:

$$\text{UNRANK}\colon \{0, 1, \dots, N-1\} \to \mathcal{S}.$$

A rank function defines an ordering on $\mathcal{S}$.
Once an ordering is chosen, we can talk about the following types of algorithms:

- Successor: given an object, return its successor.
- Rank: given an object $S \in \mathcal{S}$, return $\text{RANK}(\mathrm{S})$
- Unrank: given a rank $i \in \{0, 1, \dots, N-1\}$, return $\text{UNRANK}(i)$, its corresponding object.

Generating Subsets (of an n-set): Lexicographical Ordering

Represent a subset of an n-set by its **characteristic vector**:

subset X of {1,2,3}	characteristic vector
{1,2}	[1,1,0]
{3}	[0,0,1]

Definition

The **characteristic vector** *of a subset $T \subseteq X$ is a vector $\mathcal{X}(T) = [x_{n-1}, x_{n-2}, \ldots, x_1, x_0]$ where*

$$x_i = \begin{cases} 1, & \text{if } n-i \in T \\ 0, & \text{otherwise.} \end{cases}$$

Example: lexicographical ordering of subsets of a 3-set

lexico rank	$\mathcal{X}(T) = [x_2, x_1, x_0]$	T
0	$[0, 0, 0]$	$\emptyset$
1	$[0, 0, 1]$	$\{3\}$
2	$[0, 1, 0]$	$\{2\}$
3	$[0, 1, 1]$	$\{2, 3\}$
4	$[1, 0, 0]$	$\{1\}$
5	$[1, 0, 1]$	$\{1, 3\}$
6	$[1, 1, 0]$	$\{1, 2\}$
7	$[1, 1, 1]$	$\{1, 2, 3\}$

Note that the order is lexicographical on $\mathcal{X}(T)$ and not on T.
Note that $\mathcal{X}(T)$ corresponds to the binary representation of rank!

Ranking

More efficient implementation:

```
SubsetLexRank $(n, T)$
    $r \leftarrow 0$;
    for $i \leftarrow 1$ to $n$ do
        $r \leftarrow 2 * r$;
        if $(i \in T)$ then $r \leftarrow r + 1$;
    return $r$;
```

Books'version:

```
if $(i \in T)$ then
  $r \leftarrow r + 2^{n-i}$
```

This is like a conversion from the binary representation to the number.

Unranking

```
SubsetLexUnrank $(n, r)$
    $T \leftarrow \emptyset$;
    for $i \leftarrow n$ downto $1$ do
        if ($r$ mod $2 = 1$) then $T \leftarrow T \cup \{i\}$;
        $r \leftarrow \lfloor \frac{r}{2} \rfloor$;
    return $T$;
```

This is like a conversion from number to its binary representation.

Successor

The following algorithm is adapted for circular ranking, that is, the successor of the largest ranked object is the object of rank 0.

```
SubsetLexSuccessor $(n, T)$
    $i \leftarrow 0$;
    while $(i \leq n-1)$ and $(n-i \in T)$ do
        $T \leftarrow T \setminus \{n-i\}$;
        $i \leftarrow i+1$;
    if $(i \leq n-1)$ then $T \leftarrow T \cup \{n-i\}$;
    return $T$;
```

This algorithm works like an increment on a binary number.

Examples: successor of a subset in lexicographical ordering

1. $\textsc{SubsetLexSuccessor}(3, \{2,3\}) = \{1\}$.

$\{2,3\}$	$[\overline{0}, \underline{1, 1}]$
$\{1\}$	$[1, 0, 0]$

2. $\textsc{SubsetLexSuccessor}(4, \{1,4\}) = \{1,3\}$.

$\{1,4\}$	$[1, 0, \overline{0}, \underline{1}]$
$\{1,3\}$	$[1, 0, 1, 0]$

Generating Subsets (of an n-set) : Minimal Change Ordering

In minimal change ordering, successive sets are as similar as possible.

The **hamming distance** between two vectors is defined as the number of bits in which the two vectors differ.

Example: $dist(\underline{0}00\underline{1}010, \underline{1}00\underline{0}010) = 2$.

When we apply to the subsets corresponding to the binary vectors, it is equivalent to:

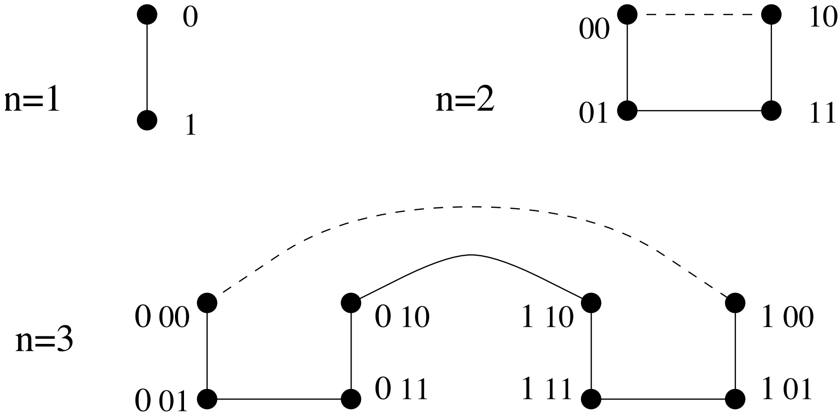
$dist(T_1, T_2) = |T_1 \triangle T_2| = |(T_1 \setminus T_2) \cup (T_2 \setminus T_1)|$.

A **Gray Code** is a sequence of vectors with successive vectors having hamming distance exactly 1.

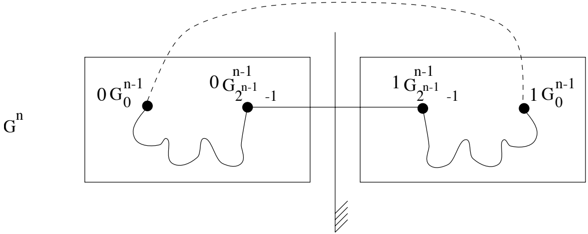
Example: $[00, 01, 11, 10]$.

We will now see a construction for one possible type of Gray Codes...

Construction for Binary Reflected Gray Codes



In general, build G_n as follows:



More formally, we define G^n inductively as follows:

$$\begin{aligned} G^1 &= [0,1] \\ G^n &= [0G_0^{n-1}, \cdots, 0G_{2^{n-1}-1}^{n-1}, 1G_{2^{n-1}-1}^{n-1}, \cdots 1G_0^{n-1}] \end{aligned}$$

Theorem (2.1)

For any $n \geq 1$, G^n is a gray code.

Exercise: prove this theorem by induction on n.

Successor

Examples:

$$\begin{aligned}
G_1 &= [0, 1] \\
G_2 &= [00, 01, 11, 10] \\
G_3 &= [000, 001, 011, 010, 110, 111, 101, 100] \\
G_4 &= [0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, \\
&\quad 1100, 1101, 1111, 1110, 1010, 1011, 1001, 1000].
\end{aligned}$$

Rules for calculating successor:

- If vector has even weight (even number of 1's): flip last bit.
- If vector has odd weight (odd number of 1's): from right to left, flip bit after the first 1.

```
GRAYCODESUCCESSOR $(n, T)$
        if ($|T|$ is even) then
        $U \leftarrow T \triangle \{n\}$;          (flip last bit)
        else
            $j \leftarrow n$;
            while ($j \notin T$) and ($j > 0$) do $j \leftarrow j - 1$;
            if ($j = 1$) then $U \leftarrow \emptyset$;      (I changed for circular order)
                          else $U \leftarrow T \triangle \{j - 1\}$;
        return $U$;
```

Ranking and Unranking

	r	0	1	2	3	4	5	6	7
$b_3b_2b_1b_0$	bin.rep. r	000	001	010	011	100	101	110	111
$a_2a_1a_0$	G^3_r	000	001	011	010	110	111	101	100

Set $b_3 = 0$ in the example above.

We need to relate $(b_n b_{n-1} \dots b_0)$ and $(a_{n-1} a_{n-2}, \dots a_0)$.

Lemma (Lemma 1.)

Let $P(n)$: "For $0 \leq r \leq 2^n - 1, a_j \equiv b_j + b_{j+1} \pmod 2$, for all $0 \leq j \leq n-1$". Then, $P(n)$ holds for all $n \geq 1$.

Lemma (Lemma 1.)

Let $P(n)$: "For $0 \leq r \leq 2^n - 1, a_j \equiv b_j + b_{j+1} \pmod 2$, for all $0 \leq j \leq n-1$". Then, $P(n)$ holds for all $n \geq 1$.

Proof: We will prove $P(n)$ by induction on n.
Basis: $P(1)$ holds, since $a_0 = b_0$ and $b_1 = 0$.
Induction step: Assume $P(n-1)$ holds. We will prove $P(n)$ holds.
Case 1. $r \leq 2^{n-1} - 1$ **(first half of G_n).**
Note that $b_{n-1} = 0 = a_{n-1}$ and $b_n = 0$, which implies

$$a_{n-1} = 0 = b_{n-1} + b_n. \tag{1}$$

By induction,

$$a_j \equiv b_j + b_{j+1} \pmod 2, \text{ for all } 0 \leq j \leq n-2. \tag{2}$$

Equations (1) and (2) imply $P(n)$.

Proof of Lemma 1 (cont'd)

Case 2. $2^{n-1} \leq r \leq 2^n - 1$ **(second half of** G_n**).**
Note that $b_{n-1} = 1 = a_{n-1}$ and $b_n = 0$, which implies

$$a_{n-1} \equiv 1 \equiv b_{n-1} + b_n \pmod 2. \tag{3}$$

Now, $G^n_r = 1G^{n-1}_{2^n-1-r} = 1a_{n-2}a_{n-3}\ldots a_1a_0$. The binary representation of $2^n - 1 - r$ is $0(1-b_{n-2})(1-b_{n-3})\ldots(1-b_1)(1-b_0)$.
By induction hypothesis we know that, for all $0 \leq j \leq n-2$,

$$\begin{aligned} a_j &\equiv (1-b_j) + (1-b_{j+1}) \pmod 2 && (4) \\ &\equiv b_j + b_{j+1} \pmod 2 && (5) \end{aligned}$$

Equations (3) and (5) imply $P(n)$. □

Lemma (Lemma 2.)

Let $n \geq 1$, $0 \leq r \leq 2^n - 1$. Then,

$$b_j \equiv \sum_{i=j}^{n-1} a_i \pmod 2, \quad \text{for all } 0 \leq j \leq n-1.$$

Proof:

$$\begin{aligned} \sum_{i=j}^{n-1} a_i &\equiv \sum_{i=j}^{n-1} (b_i + b_{i+1}) \pmod 2 \quad \text{[By Lemma 1]} \\ &\equiv b_j + 2b_{j+1} + \ldots + 2b_{n-1} + b_n \pmod 2 \\ &\equiv b_j + b_n \pmod 2 \\ &\equiv b_j \pmod 2 \qquad \text{[Since } b_n = 0]. \quad \square \end{aligned}$$

Let $n \geq 1$, $0 \leq r \leq 2^n - 1$.
We haved proved the following properties hold, for all $0 \leq i \leq n-1$,

$$b_i \equiv \sum_{j=i}^{n-1} a_j \pmod 2.$$

$$a_i \equiv b_i + b_{i+1} \pmod 2,$$

The first property is used for ranking:

```
$\textsc{GrayCodeRank}\ (n, T)$
    $r \leftarrow 0$; $b \leftarrow 0$;
    for $i \leftarrow n-1$ downto $0$ do
        if $((n-i) \in T)$ then            ( if $a_i = 1$)
            $b \leftarrow 1 - b$;          ( $b_i = \overline{b_{i+1}}$)
        $r \leftarrow 2r + b$;
    return $r$;
```

The second property is used for unranking:

$$a_i \equiv b_i + b_{i+1} \pmod 2, \text{ for all } 0 \leq i \leq n-1$$

```
GrayCodeUnrank $(n, r)$
    $T \leftarrow \emptyset$; $b' \leftarrow r \bmod 2$; $r' \leftarrow \lfloor \frac{r}{2} \rfloor$;
    for $i \leftarrow 0$ to $n-1$ do
        $b \leftarrow r' \bmod 2$
        if $(b \neq b')$ then $T \leftarrow T \cup \{n-i\}$;
        $b' \leftarrow b$; $r' \leftarrow \lfloor \frac{r'}{2} \rfloor$;
    return $T$;
```

Generating k-subsets (of an n-set): Lexicographical Ordering

Example: $k = 3$, $n = 5$.

rank	T	$\vec{T}$
0	$\{1,2,3\}$	$[1,2,3]$
1	$\{1,2,4\}$	$[1,2,4]$
2	$\{1,2,5\}$	$[1,2,5]$
3	$\{1,3,4\}$	$[1,3,4]$
4	$\{1,3,5\}$	$[1,3,5]$
5	$\{1,4,5\}$	$[1,4,5]$
6	$\{2,3,4\}$	$[2,3,4]$
7	$\{2,3,5\}$	$[2,3,5]$
8	$\{2,4,5\}$	$[2,4,5]$
9	$\{3,4,5\}$	$[3,4,5]$

Successor

Example/idea: $n = 10$, $\textsc{Successor}(\{\ldots, \underline{5}, 8, 9, 10\})$=$\{\ldots, \underline{6}, 7, 8, 9\}$

```
kSubsetLexSuccessor $(\vec{T}, k, n)$
    $\vec{U} \leftarrow \vec{T}$; $i \leftarrow k$;
    while $(i \geq 0)$ and $(t_i = n - k + i)$ do $i \leftarrow i - 1$;
    if $(i = 0)$ then $\vec{U} = [1, 2, \ldots, k]$;
    else for $j \leftarrow i$ to $k$ do
        $u_j \leftarrow (t_i + 1) + j - i$;
    return $\vec{U}$;
```

Ranking

How many subsets preceed $\vec{T} = [t_1, t_2, \ldots, t_k]$?

all sets $[X, \ldots]$ with $1 \leq X \leq t_1 - 1$

$(\sum_{j=1}^{t_1-1} \binom{n-j}{k-1})$

all sets $[t_1, X, \ldots]$ with $t_1 + 1 \leq X \leq t_2 - 1$

$(\sum_{j=t_1+1}^{t_2-1} \binom{n-j}{k-2})$

$\vdots$

all sets $[t_1, \ldots, t_{k-1}, X, \ldots]$ with $t_{k-1} + 1 \leq X \leq t_k - 1$

$(\sum_{j=t_{k-1}+1}^{t_k-1} \binom{n-j}{k-(k-1)})$

Thus,

$$rank(T) = \sum_{i=1}^{k} \sum_{j=t_{i-1}+1}^{t_i-1} \binom{n-j}{k-i},$$

where $t_0 := 0$.

```
kSubsetLexRank ($\vec{T}, k, n$)
    $r \leftarrow 0$;
    $t_0 \leftarrow 0$;
    for $i \leftarrow 1$ to $k$ do
        for $j \leftarrow t_{i-1} + 1$ to $t_i - 1$ do
            $r \leftarrow r + \binom{n-j}{k-i}$;
    return r;
```

Unranking

$$t_1 = x \iff \sum_{j=1}^{x-1} \binom{n-j}{k-1} \leq r < \sum_{j=1}^{x} \binom{n-j}{k-1}$$

$$t_2 = x \iff \sum_{j=t_1+1}^{x-1} \binom{n-j}{k-2} \leq r - \sum_{j=1}^{t_1-1} \binom{n-j}{k-1} < \sum_{j=t_1+1}^{x} \binom{n-j}{k-2}$$

etc.

```
kSubsetLexUnrank $(r, k, n)$
        $x \leftarrow 1$;
        for $i \leftarrow 1$ to $k$ do
            while $(r \geq \binom{n-x}{k-i})$ do
                $r \leftarrow r - \binom{n-x}{k-i}$;
                $x \leftarrow x + 1$;
            $t_i \leftarrow x$;
            $x \leftarrow x + 1$;
        return $\vec{T}$ ;
```

Generating k-subsets : Minimal Change Ordering

The minimum Hamming distance possible between k-subsets is 2.

Revolving Door Ordering

It is based on Pascal's Identity: $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$.

We define the sequence of k-subsets $A^{n,k}$ based on $A^{n-1,k}$ and the reverse of $A^{n-1,k-1}$, as follows:

$$\begin{aligned} A^{n,k} &= \left[A_0^{n-1,k}, \ldots, A^{n-1,k}_{\binom{n-1}{k}-1}, |A^{n-1,k-1}_{\binom{n-1}{k-1}-1} \cup \{n\}, \ldots, A_0^{n-1,k-1} \cup \{n\}\right], \\ &\quad for\ 1 \leq k \leq n-1 \\ A^{n,0} &= [\emptyset] \\ A^{n,n} &= [\{1, 2, \ldots, n\}] \end{aligned}$$

Example: Building $A^{5,3}$ using $A^{4,3}$ and $A^{4,2}$

$$\begin{aligned}
A^{4,3} &= [\{1,2,3\},\{1,3,4\},\{2,3,4\},\{1,2,4\}] \\
A^{4,2} &= [\{1,2\},\{2,3\},\{1,3\},\{3,4\},\{2,4\},\{1,4\}]
\end{aligned}$$

$$\begin{aligned}
A^{5,3} &= [\{1,2,3\},\{1,3,4\},\{2,3,4\},\{1,2,4\},| \\
&\quad |\{1,4,\mathbf{5}\},\{2,4,\mathbf{5}\},\{3,4,\mathbf{5}\},\{1,3,\mathbf{5}\},\{2,3,\mathbf{5}\},\{1,2,\mathbf{5}\}]
\end{aligned}$$

Generating k-subsets (of an n-set): Minimal Change Ordering

To see that the revolving door ordering is a minimal change ordering, prove:

1. $A^{n,k}_{\binom{n}{k}-1} = \{1, 2, \ldots, k-1, n\}$.
2. $A^{n,k}_0 = \{1, 2, \ldots, k\}$.
3. For any n, k, $1 \leq k \leq n$, $A^{n,k}$ is a minimal change ordering of $\mathcal{S}^n_k$.

Ranking

The ranking algorithm is based on the following fact (prove it as an exercise):

$$rank(T) = \sum_{i=1}^{k} (-1)^{k-i} \left(\binom{t_i}{i} - 1 \right) = \begin{cases} \sum_{i=1}^{k} (-1)^{k-i} \binom{t_i}{i}, & k \text{ even} \\ \left[\sum_{i=1}^{k} (-1)^{k-i} \binom{t_i}{i} \right] - 1, & k \text{ odd} \end{cases}$$

Hint: Prove the first equality by induction and the second, directly.

```
$\textsc{KsubsetRevDoorRank}(\vec{T}, k)$
        $r \leftarrow -(k \bmod 2)$;
        $s \leftarrow 1$;
        for $i \leftarrow k$ downto $1$ do
            $r \leftarrow r + s\binom{t_i}{i}$
            $s \leftarrow -s$;
        return $r$;
```

Ranking Algorithm Example 1

$$\textsc{rank}(136) = r = ?$$

Look at $\downarrow \binom{6}{3} = +20$, then $\uparrow \binom{3}{2} = -3$, then $\downarrow \binom{1}{1} = +1$ then -1

123	156	
134	256	
234	356	
124	456	
145	146	
245	246	
345	346	
135	$\underline{1}\overline{3}6$	$-\binom{3}{2} = -3 \downarrow \; +\binom{1}{1} = 1 - 1 \quad r = 17$
235	236	
125	$12\underline{6}$	$+\binom{6}{3} = +20 \uparrow$

Ranking Algorithm Example 2

$$\text{RANK}(245) = r = ?$$

Look at $\downarrow \binom{5}{3} = +10$, then $\uparrow \binom{4}{2} = -6$, then $\downarrow \binom{2}{1} = +2$ then -1

123
134
234
124
$1\overline{4}5 \quad -\binom{4}{2} = -6 \downarrow$
$\underline{2}45 \quad +\binom{2}{1} = +2 - 1$
345
135
235
$12\underline{5} \quad +\binom{5}{3} = +10 \uparrow$

156
256
356
456
146
246
346
136
236
126

Unranking

IDEA/ Example: $n = 7$, $k = 4$, $r = 8$

$4 \in T$, $5, 6, 7 \notin T$	$5 \in T$, $6, 7 \notin T$	$6 \in T$, $7 \notin T$	$7 \in T$
$\binom{4}{4} = 1$	$\binom{5}{4} = 5$	$\binom{6}{4} = 15$	$\binom{7}{4} = 21$

Unranking

IDEA/ Example: $n = 7$, $k = 4$, $r = 8$

$4 \in T$, $5, 6, 7 \notin T$	$5 \in T$, $6, 7 \notin T$	$6 \in T$, $7 \notin T$	$7 \in T$
$\binom{4}{4} = 1$	$\binom{5}{4} = 5$	$\binom{6}{4} = 15$	$\binom{7}{4} = 21$

We can determine the largest element in the set: $r = 8$ implies $\{\_, \_, \_, 6\}$.

Unranking

IDEA/ Example: $n = 7$, $k = 4$, $r = 8$

$4 \in T$, $5, 6, 7 \notin T$	$5 \in T$, $6, 7 \notin T$	$6 \in T$, $7 \notin T$	$7 \in T$
$\binom{4}{4} = 1$	$\binom{5}{4} = 5$	$\binom{6}{4} = 15$	$\binom{7}{4} = 21$

We can determine the largest element in the set: $r = 8$ implies $\{\_, \_, \_, 6\}$.
Now, solve it recursively for $n' = 5$, $k' = 3$, $r' = \binom{6}{4} - r - 1 = 6$.

Unranking

IDEA/ Example: $n = 7$, $k = 4$, $r = 8$

$4 \in T$, $5, 6, 7 \notin T$	$5 \in T$, $6, 7 \notin T$	$6 \in T$, $7 \notin T$	$7 \in T$
$\binom{4}{4} = 1$	$\binom{5}{4} = 5$	$\binom{6}{4} = 15$	$\binom{7}{4} = 21$

We can determine the largest element in the set: $r = 8$ implies $\{\_, \_, \_, 6\}$.
Now, solve it recursively for $n' = 5$, $k' = 3$, $r' = \binom{6}{4} - r - 1 = 6$.

```
KsubsetRevDoorUnrank($r, k, n$)
        $x \leftarrow n$;
        for $i \leftarrow k$ downto 1 do
            While $\binom{x}{i} > r$ do   $x \leftarrow x - 1$;
            $t_i \leftarrow x + 1$
            $r \leftarrow \binom{x+1}{i} - r - 1$;
        return $\vec{T}$;
```

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots \; [12] \; \ldots, \binom{5}{3} = 10, \ldots, \; \binom{4}{3} = 4, \ldots, \; \binom{3}{3} = 1$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots\ [12]\ \ldots, \binom{5}{3} = 10, \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 5$, $k = 2$, $r = 20 - 12 - 1 = 7$, $T = [?, ?, 6]$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots [12] \ldots, \binom{5}{3} = 10, \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 5$, $k = 2$, $r = 20 - 12 - 1 = 7$, $T = [?, ?, 6]$
- $\binom{5}{2} = 10, \ldots [7] \ldots, \binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots, \binom{2}{2} = 1$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots$ [12] $\ldots, \binom{5}{3} = 10, \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 5$, $k = 2$, $r = 20 - 12 - 1 = 7$, $T = [?, ?, 6]$
- $\binom{5}{2} = 10, \ldots$ [7] $\ldots, \binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots, \binom{2}{2} = 1$

- $n = 4$, $k = 1$, $r = 10 - 7 - 1 = 2$, $T = [?, 5, 6]$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots\ [12]\ \ldots, \binom{5}{3} = 10, \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 5$, $k = 2$, $r = 20 - 12 - 1 = 7$, $T = [?, ?, 6]$
- $\binom{5}{2} = 10, \ldots [7] \ldots, \binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots, \binom{2}{2} = 1$

- $n = 4$, $k = 1$, $r = 10 - 7 - 1 = 2$, $T = [?, 5, 6]$
- $\binom{4}{1} = 4, \ldots, \binom{3}{1} = 3, \ldots [2], \ldots, \binom{2}{1} = 2, \ldots, \binom{1}{1} = 1$

Unranking Algorithm: Example 1

- $n = 6$, $k = 3$, $r = 12$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots [12] \ldots, \binom{5}{3} = 10, \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 5$, $k = 2$, $r = 20 - 12 - 1 = 7$, $T = [?, ?, 6]$
- $\binom{5}{2} = 10, \ldots [7] \ldots, \binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots, \binom{2}{2} = 1$

- $n = 4$, $k = 1$, $r = 10 - 7 - 1 = 2$, $T = [?, 5, 6]$
- $\binom{4}{1} = 4, \ldots, \binom{3}{1} = 3, \ldots [2], \ldots, \binom{2}{1} = 2, \ldots, \binom{1}{1} = 1$

- $T = [3, 5, 6]$

Unranking Algorithm: Example 2

- $n = 6$, $k = 3$, $r = 7$, $T = [?, ?, ?]$

Unranking Algorithm: Example 2

- $n = 6$, $k = 3$, $r = 7$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

Unranking Algorithm: Example 2

- $n = 6,\ k = 3,\ r = 7,\ T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 4,\ k = 2,\ r = 10 - 7 - 1 = 2,\ T = [?, ?, 5]$

Unranking Algorithm: Example 2

- $n = 6$, $k = 3$, $r = 7$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 4$, $k = 2$, $r = 10 - 7 - 1 = 2$, $T = [?, ?, 5]$
- $\binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots [2] \ldots, \binom{2}{2} = 1$

Unranking Algorithm: Example 2

- $n = 6$, $k = 3$, $r = 7$, $T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots, \binom{4}{3} = 4, \ldots, \binom{3}{3} = 1$

- $n = 4$, $k = 2$, $r = 10 - 7 - 1 = 2$, $T = [?, ?, 5]$
- $\binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots [2] \ldots, \binom{2}{2} = 1$

- $n = 2$, $k = 1$, $r = 3 - 2 - 1 = 0$, $T = [?, 3, 5]$

Unranking Algorithm: Example 2

- $n = 6,\ k = 3,\ r = 7,\ T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots,\ \binom{4}{3} = 4, \ldots,\ \binom{3}{3} = 1$

- $n = 4,\ k = 2,\ r = 10 - 7 - 1 = 2,\ T = [?, ?, 5]$
- $\binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots [2] \ldots, \binom{2}{2} = 1$

- $n = 2,\ k = 1,\ r = 3 - 2 - 1 = 0,\ T = [?, 3, 5]$
- $\binom{2}{1} = 2, \ldots, \binom{1}{1} = 1, \ldots [0]$

Unranking Algorithm: Example 2

- $n = 6,\ k = 3,\ r = 7,\ T = [?, ?, ?]$
- $\binom{6}{3} = 20, \ldots, \binom{5}{3} = 10, \ldots [7] \ldots,\ \binom{4}{3} = 4, \ldots,\ \binom{3}{3} = 1$

- $n = 4,\ k = 2,\ r = 10 - 7 - 1 = 2,\ T = [?, ?, 5]$
- $\binom{4}{2} = 6, \ldots, \binom{3}{2} = 3, \ldots [2] \ldots, \binom{2}{2} = 1$

- $n = 2,\ k = 1,\ r = 3 - 2 - 1 = 0,\ T = [?, 3, 5]$
- $\binom{2}{1} = 2, \ldots, \binom{1}{1} = 1, \ldots [0]$

- $T = [1, 3, 5]$

Successor

Let $\vec{T} = [1, 2, 3, \ldots, j-1, t_j, \ldots]$, where $j = \min\{i : t_i \neq i\}$.
Consider fours cases for computing successor:

- Case A: $k \equiv j \pmod 2$
    - Case A1: if $t_{j+1} = t_j + 1$ then move j to the right, and remove $t_j + 1$.
      Example: $\textsc{Successor}(\{\underline{1,2,3},\mathbf{7},\overline{8},12\}) = \{\{\underline{1,2,3,4},\mathbf{7},12\}$.
    - Case A2: if $t_{j+1} \neq t_j + 1$ then move j to the left, and add $t_j + 1$.
      Example: $\textsc{Successor}(\{\underline{1,2,3},\mathbf{7},10,12\}) = \{\underline{1,2},\mathbf{7},\overline{8},10,12\}$.
- Case B: $k \not\equiv j \pmod 2$
    - Case B1: if $j > 1$ then increment t_{j-1} and (if exists) t_{j-2}.
      Example: $\textsc{Successor}(\{\underline{1,2,3},\mathbf{7},10\}) = \{1,\underline{3,4},\mathbf{7},10\}$.
    - Case B2: if $j = 1$ then decrement t_1
      Example: $\textsc{Successor}\{7,9,10,12\}) = \{6,9,10,12\}$.

For each case, prove $\text{Rank}(\text{successor}(T)) - \text{Rank}(T) = 1$.
Proof of case A1: $\text{Successor}(\{\underline{1,2,3},\mathbf{7},\overline{8},12\}) = \{\{\underline{1,2,3,4},\mathbf{7},12\}$.
$\text{rank}(\text{successor}(T)) - \text{rank}(T) =$

$$\begin{aligned} &= (-1)^{k-j}\binom{j}{j} + (-1)^{k-j-1}\binom{t_j}{j+1} \\ &\quad -(-1)^{k-j}\binom{t_j}{j} - (-1)^{k-j-1}\binom{t_j+1}{j+1} \\ &= \binom{j}{j} + \left(\binom{t_j+1}{j+1} - \binom{t_j}{j+1} - \binom{t_j}{j}\right) = 1 + 0 = 1. \end{aligned}$$

Prove other cases A2, B1, B2, similarly.

Successor Algorithm: Case A2

- $\{ \quad 1 \quad 2 \quad 3 \quad 7 \quad 10 \quad 12 \quad \}$

Successor Algorithm: Case A2

- $\{\ \ 1\ \ \ 2\ \ \ 3\ \ \ 7\ \ \ 10\ \ \ 12\ \ \}$
- $\{\ \ 1\ \ \ 2\ \ \ \underset{\uparrow}{3}\ \ \ \underset{\downarrow}{7}\ \ \ \underset{\uparrow}{10}\ \ \ \underset{\downarrow}{12}\ \ \}$ $(1237$ is the last of $A^{7,4})$

Successor Algorithm: Case A2

- $\{ \quad 1 \quad 2 \quad 3 \quad 7 \quad 10 \quad 12 \quad \}$
- $$\begin{array}{cccccccc} \{ & 1 & 2 & 3 & 7 & 10 & 12 & \} \\ & & & \uparrow & \downarrow & \uparrow & \downarrow & \end{array}$$ (1237 is the last of $A^{7,4}$)

- $\{ \quad ? \quad ? \quad ? \quad 8 \quad 10 \quad 12 \quad \}$ (now the block ending in 8 starts)

Successor Algorithm: Case A2

- { 1 2 3 7 10 12 }
- { 1 2 3 7 10 12 } (1237 is the last of $A^{7,4}$)

  ↑ ↓ ↑ ↓
- { ? ? ? 8 10 12 } (now the block ending in 8 starts)
- { 1 2 **7** 8 10 12 } (before 8, put last of $A^{7,3}$)

Successor Algorithm: Case A1

- $\{ \quad 1 \quad 2 \quad 3 \quad 7 \quad 8 \quad 12 \quad \}$

Successor Algorithm: Case A1

- { 1 2 3 7 8 12 }
- $\begin{array}{cccccccc} \{ & 1 & 2 & 3 & 7 & 8 & 12 & \} \\ & & & & \downarrow & \uparrow & \downarrow & \end{array}$ 1237 is the last of $A^{7,4}$; can't $++7$

Successor Algorithm: Case A1

- { 1 2 3 7 8 12 }
- { 1 2 3 7 8 12 } 1237 is the last of $A^{7,4}$; can't $++7$

  $\qquad\qquad\qquad\downarrow\ \uparrow\ \downarrow$

- { ? ? ? ? 7 12 } 12378 is 1st ending on 8 of $A^{8,5}$

Successor Algorithm: Case A1

- $\{ \quad 1 \quad 2 \quad 3 \quad 7 \quad 8 \quad 12 \quad \}$
- $\begin{array}{cccccccc} \{ & 1 & 2 & 3 & 7 & 8 & 12 & \} \\ & & & & \downarrow & \uparrow & \downarrow & \\ \hline \end{array}$ 1237 is the last of $A^{7,4}$; can't $++7$
- $\{ \quad ? \quad ? \quad ? \quad ? \quad 7 \quad 12 \quad \}$ 12378 is 1st ending on 8 of $A^{8,5}$
- $\{ \quad 1 \quad 2 \quad 3 \quad \mathbf{4} \quad 7 \quad 12 \quad \}$ pred in $A^{8,5}$ is last of $A^{7,5}$, so 12347

Successor Algorithm: Case B1

- $\{ \quad 1 \quad 2 \quad 3 \quad 4 \quad 7 \quad 12 \quad \}$

Successor Algorithm: Case B1

- $\{ \quad 1 \quad 2 \quad 3 \quad 4 \quad 7 \quad 12 \quad \}$
- $\{ \quad 1 \quad 2 \quad 3 \quad 4 \quad 7 \quad 12 \quad \}$ 12347 is last of $A^{7,5}$

$$\begin{array}{cccccc} 1 & 2 & 3 & 4 & 7 & 12 \\ & & \uparrow & \downarrow & \uparrow & \downarrow \\ \hline \end{array}$$

Successor Algorithm: Case B1

- $\{\ 1\ \ 2\ \ 3\ \ 4\ \ 7\ \ 12\ \}$
- $\begin{array}{cccccccc} \{ & 1 & 2 & 3 & 4 & 7 & 12 & \} \\ & & & \uparrow & \downarrow & \uparrow & \downarrow & \\ \hline \end{array}$ 12347 is last of $A^{7,5}$
- $\begin{array}{cccccccc} \{ & ? & ? & ? & \mathbf{5} & 7 & 12 & \} \\ & & & \uparrow & \downarrow & \uparrow & \downarrow & \end{array}$ pred is second to last of $A^{7,5}$

Successor Algorithm: Case B1

- { 1 2 3 4 7 12 }

- { 1 2 3 4 7 12 } 12347 is last of $A^{7,5}$

  ↑ ↓ ↑ ↓

- { ? ? ? **5** 7 12 } pred is second to last of $A^{7,5}$

  ↑ ↓ ↑ ↓

- { 1 2 **4** **5** 7 12 } i.e. successor$(1234) = 1245$

Successor Algorithm: Case B2

- $\{ \quad 6 \quad 9 \quad 10 \quad 12 \quad 16 \quad 19 \quad \}$

Successor Algorithm: Case B2

- $\{ \quad 6 \quad 9 \quad 10 \quad 12 \quad 16 \quad 19 \quad \}$
- $\{ \quad 6 \quad 9 \quad 10 \quad 12 \quad 16 \quad 19 \quad \}$ leftmost possible change is in 1st
  $\quad\quad \uparrow \quad \downarrow \quad \uparrow \quad \downarrow \quad \uparrow \quad \downarrow$

Successor Algorithm: Case B2

- $\{\quad 6 \quad 9 \quad 10 \quad 12 \quad 16 \quad 19 \quad\}$
- $\begin{array}{cccccccc} \{ & 6 & 9 & 10 & 12 & 16 & 19 & \} \\ & \uparrow & \downarrow & \uparrow & \downarrow & \uparrow & \downarrow & \end{array}$ leftmost possible change is in 1st
- $\{\quad 5 \quad 9 \quad 10 \quad 12 \quad 16 \quad 19 \quad\}$ given direction, do $6--$

Generating k-subsets (of an n-set): Minimal Change Ordering

```
KSUBSETREVDOORSUCCESSOR($\vec{T}, k, n$)
        $t_{k+1} \leftarrow n + 1$;
        $j \leftarrow 1$;
        While $(j \leq k)$ and $(t_j = j)$ do  $j \leftarrow j + 1$;
        if $(k \not\equiv j \pmod 2)$ then
            if $(j = 1)$ then $t_1 \leftarrow t_1 - 1$; (Case B2)
            else  (Case B1)
                  $t_{j-1} \leftarrow j$;
                  $t_{j-2} \leftarrow j - 1$;
        else if $(t_{j+1} \neq t_j + 1)$ then (Case A2)
                  $t_{j-1} \leftarrow t_j$;
                  $t_j \leftarrow t_j + 1$
              else  (Case A1)
                    $t_{j+1} \leftarrow t_j$;
                    $t_j \leftarrow j$;
        return $\vec{T}$;
```

Generating Permutations: Lexicographical Ordering

A permutation is a bijection $\Pi : \{1, 2, \ldots, n\} \rightarrow \{1, 2, \ldots, n\}$.

We represent it by a list: $\Pi = [\Pi[1], \Pi[2], \ldots, \Pi[n]]$.

Lexicographical Ordering: $n = 3$

rank	permutation
0	$[1, 2, 3]$
1	$[1, 3, 2]$
2	$[2, 1, 3]$
3	$[2, 3, 1]$
4	$[3, 1, 2]$
5	$[3, 2, 1]$

Successor

Example: $\Pi = [3, 5, \mathbf{4}, \underline{7, \overline{6}, 2, 1}]$

Let $i =$ index right before a decreasing suffix $= 3$.
Let $j=$ index of the successor of $\pi[i]$ in $\{\Pi[i+1], \ldots, \Pi[n]\}$
$\pi[i] = 4$, successor of $\pi[i] = 4$ in $\{7, 6, 2, 1\}$ is 6, $\pi[5] = 6$, so $j = 5$.

Swap $\Pi[i]$ and $\Pi[j]$, and reverse $\{\Pi[i+1], \ldots, \Pi[n]\}$.

$$\textsc{Successor}(\Pi) = [3, 5, \mathbf{6}, \underline{1, 2, 4, 7}]$$

Note that: $i = \max\{l : \Pi[l] < \Pi[l+1]\}$
$j = \max\{l : \Pi[l] > \Pi[i]\}.$

For the algorithm, we add: $\Pi[0] = 0$.

```
$\textsc{PermLexSuccessor}(n, \Pi)$
    $\Pi[0] \leftarrow 0;$
    $i \leftarrow n - 1;$
    while $(\Pi[i] > \Pi[i+1])$ do  $i \leftarrow i - 1;$
    if $(i = 0)$ then return $\Pi = [1, 2, \ldots, n]$
    $j \leftarrow n;$
    while $(\Pi[j] < \Pi[i])$ do  $j \leftarrow j - 1;$
    $t \leftarrow \Pi[j]; \Pi[j] \leftarrow \Pi[i]; \Pi[i] \leftarrow t;$ (swap $\Pi[i]$ and $\Pi[j]$)
    // In-place reversal of $\Pi[i+1], \ldots, \Pi[n]$:
    for $h \leftarrow i + 1$ to $\lfloor \frac{n+i}{2} \rfloor$ do
        $t \leftarrow \Pi[h]; \Pi[h] \leftarrow \Pi[n + i + 1 - h];$
        $\Pi[n + i + 1 - h] \leftarrow t;$
    return $\Pi;$
```

Ranking

How many permutations come before $\Pi = [3, 5, 1, 2, 4]$?

the ones of the form $\Pi = [1, \ldots]$ (there are $(n-1)! = 24$ of them)
the ones of the form $\Pi = [2, \ldots]$ (there are $(n-1)! = 24$ of them)
plus the rank of $[5, 1, 2, 4]$ as a permutation of $\{1, 2, 4, 5\}$, which is the standard rank of $[4, 1, 2, 3]$.

So,

$$\textsc{Rank}([3,5,1,2,4]) = 2 \times 4! + \textsc{Rank}([4,1,2,3])$$
$$= 2 \times 4! + 3 \times 3! + \textsc{Rank}([1,2,3])$$
$$= 2 \times 4! + 3 \times 3! + 0 \times 2! + \textsc{Rank}([1,2])$$
$$= 2 \times 4! + 3 \times 3! + 0 \times 2! + 0 \times 1! + \textsc{Rank}([1])$$
$$= 2 \times 4! + 3 \times 3! + 0 \times 2! + 0 \times 1! + 0 = 66$$

General Formula:

$\textsc{Rank}([1], 1) = 0$,
$\textsc{Rank}(\Pi, n) = (\Pi[1] - 1) \times (n-1)! + \textsc{Rank}(\Pi', n-1)$, where

$$\Pi'[i] = \begin{cases} \Pi[i+1] - 1, & \text{if } \Pi[i+1] > \Pi[1] \\ \Pi[i+1], & \text{if } \Pi[i+1] < \Pi[1] \end{cases}$$

```
PermLexRank($n, \Pi$)
    $r \leftarrow 0$;
    $\Pi' \leftarrow \Pi$;
    for $j \leftarrow 1$ to $n-1$ do     (Note: correction from book: $n \rightarrow n-1$)
        $r \leftarrow r + (\Pi'[j] - 1) * (n-j)!$
        for $i \leftarrow j+1$ to $n$ do
            if $(\Pi'[i] > \Pi'[j])$ then $\Pi'[i] = \Pi'[i] - 1$;
    return $r$;
```

Unranking

Unranking uses the factorial representation of r.
Let $0 \leq r \leq n! - 1$. Then, $(d_{n-1}, d_{n-2}, \ldots, d_1)$ is the factorial representation of r if

$$r = \sum_{i=1}^{n-1} d_i \times i!, \text{ where } 0 \leq d_i \leq i.$$

(Exercise: prove that such r has a unique factorial representation.)

Examples:

1. $\textsc{Unrank}(15, 4) = [3, 2, 4, 1]$

   $15 = \mathbf{2} \times 3! + \mathbf{1} \times 2! + \mathbf{1} \times 1!$, put $d_0 = \mathbf{0}$.

2	**1**	$\underline{\mathbf{1}}$	**0**
3	2	2	1

2	$\underline{\mathbf{1}}$	**1**	**0**
3	2	2	1

$\underline{\mathbf{2}}$	**1**	**1**	**0**
3	2	3	1

2	**1**	**1**	**0**
3	2	4	1

2. $\textsc{Unrank}(8, 4) = [2, 3, 1, 4]$

   $8 = \mathbf{1} \times 3! + \mathbf{1} \times 2! + \mathbf{0} \times 1!$,

1	**1**	$\underline{\mathbf{0}}$	**0**
2	2	1	1

1	$\underline{\mathbf{1}}$	**0**	**0**
2	2	1	2

$\underline{\mathbf{1}}$	**1**	**0**	**0**
2	2	1	3

1	**1**	**0**	**0**
2	3	1	4

3. $\textsc{Unrank}(21, 4) = [4, 2, 3, 1]$

   $21 = \mathbf{3} \times 3! + \mathbf{1} \times 2! + \mathbf{1} \times 1!$,

3	**1**	$\underline{\mathbf{1}}$	**0**
4	2	2	1

3	$\underline{\mathbf{1}}$	**1**	**0**
4	2	2	1

$\underline{\mathbf{3}}$	**1**	**1**	**0**
4	2	3	1

3	**1**	**1**	**0**
4	2	3	1

Justification: $\Pi[1] = d_{n-1} + 1$ because exactly d_{n-1} blocks of size $(n-1)!$ come before Π. Then, $\Pi[2], \Pi[3], \ldots, \Pi[n]$ is computed from permutation Π', as follows: $r' = r - d_{n-1} \times (n-1)!$ $\Pi' =$UNRANK$(r', n-1)$,

$$\Pi[i] = \begin{cases} \Pi'[i-1], & \text{if } \Pi'[i-1] < \Pi[1] \\ \Pi'[i-1] + 1, & \text{if } \Pi'[i-1] > \Pi[1] \end{cases} \qquad \text{for } 2 \leq i \leq n$$

```
PERMLEXUNRANK$(r, n)$
    $\Pi[n] \leftarrow 1$;
    for $j \leftarrow 1$ to $n-1$ do
        $d \leftarrow \frac{r \bmod (j+1)!}{j!}$;    // calculates $d_j$
        $r \leftarrow r - d * j!$;
        $\Pi[n-j] \leftarrow d + 1$;
        for $i \leftarrow n - j + 1$ to $n$ do
            if $(\Pi[i] > d)$ then $\Pi[i] \leftarrow \Pi[i] + 1$;
    return $\Pi$;
```

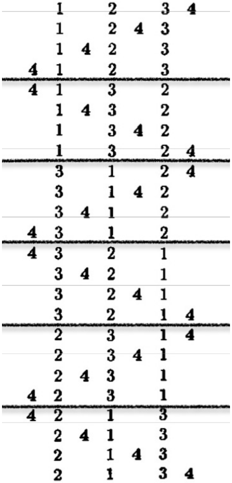
Generating permutations: Minimal Change Ordering

Minimal change for permutations: two permutations must differ by adjacent transposition.
The Trotter-Johnson algorithm follows the following ordering:

$$
\begin{array}{rcl}
T^1 & = & [[1]] \\
T^2 & = & [[1,\mathbf{2}],[\mathbf{2},1]] \\
T^3 & = & [[1,2,\mathbf{3}],[1,\mathbf{3},2],[\mathbf{3},1,2][\mathbf{3},2,1],[2,\mathbf{3},1],[2,1,\mathbf{3}]]
\end{array}
$$

How to build T^3 using T^2, and T^4 using T^3:

	1		2	**3**
	1	**3**	2	
3	1		2	
3	2		1	
	2	**3**	1	
	2		1	**3**



	1		2		3	4
	1		2	4	3	
	1	4	2		3	
4	1		2		3	
4	1		3		2	
	1	4	3		2	
	1		3	4	2	
	1		3		2	4
	3		1		2	4
	3		1	4	2	
	3	4	1		2	
4	3		1		2	
4	3		2		1	
	3	4	2		1	
	3		2	4	1	
	3		2		1	4
	2		3		1	4
	2		3	4	1	
	2	4	3		1	
4	2		3		1	
4	2		1		3	
	2	4	1		3	
	2		1	4	3	
	2		1		3	4

Ranking

Let

$$\Pi = [\Pi[1], \ldots, \Pi[k-1], \mathbf{\Pi[k] = n}, \Pi[k+1], \ldots, \Pi[n]].$$

Thus, Π is built from Π' by inserting n, where

$$\Pi' = [\Pi[1], \ldots, \Pi[k-1], \Pi[k+1], \ldots, \Pi[n]].$$

$\textsc{Rank}(\Pi, n) = n \times \ \textsc{Rank}(\Pi', n-1) + E$,

$$E = \begin{cases} n-k, & \text{if } \mathrm{Rank}(\Pi', n-1) \text{ is } \textit{even} \\ k-1, & \text{if } \mathrm{Rank}(\Pi', n-1) \text{ is } \textit{odd} \end{cases}$$

Example:
$\textsc{Rank}([3,4,2,1],4) = 4 \times \textsc{Rank}([3,2,1],3) + E = 4 \times 3 + (2-1) = 13.$

```
PermTrotterJohnsonRank($\Pi, n$)
    $r \leftarrow 0$;
    for $j \leftarrow 2$ to $n$ do
        $k \leftarrow 1$; $i \leftarrow 1$;
        while $(\Pi[i] \neq j)$ do
            if $(\Pi[i] < j)$ then $k \leftarrow k + 1$;
            $i \leftarrow i + 1$;
        if $(r \equiv 0 \mod 2)$ then $r \leftarrow j * r + j - k$;
                              else $r \leftarrow j * r + k - 1$;
    return $r$;
```

Unranking

Based on similar recursive principle.
Let $r' = \lfloor \frac{r}{n} \rfloor$, $\Pi' = \textsc{Unrank}(r', n-1)$.
Let $k = r - n \times r'$.
Insert n into Π' in position:

$$\begin{array}{ll} k+1, & \text{if } r' \text{ is odd} \\ n-k, & \text{if } r' \text{ is even} \end{array}$$

```
PermTrotterJohnsonUnrank$(n, r)$
    $\Pi[1] \leftarrow 1$;
    $r_2 \leftarrow 0$;
    for $j \leftarrow 2$ to $n$ do
        $r_1 \leftarrow \lfloor \frac{r*j!}{n!} \rfloor$;    // rank of $\Pi$ when restricted to $\{1, 2, \ldots, j\}$
        $k \leftarrow r_1 - j * r_2$;
        if ($r_2$ is even) then
            for $i \leftarrow j - 1$ downto $j - k$ do
                $\Pi[i + 1] \leftarrow \Pi[i]$;
            $\Pi[j - k] \leftarrow j$;
        else
            for $i \leftarrow j - 1$ downto $k + 1$ do
                $\Pi[i + 1] \leftarrow \Pi[i]$;
            $\Pi[k + 1] \leftarrow j$;
        $r_2 \leftarrow r_1$;
    return $\Pi$;
```

Successor

There are four cases to analyse:

- $\textsc{Rank}(\Pi')$ is even
  - If possible, move left:
    $\textsc{Successor}([1, \mathbf{4}, 2, 3])$=$([\mathbf{4}, 1, 2, 3])$
  - If n is in first position, get successor of the remaining permutation:
    $\textsc{Successor}([\mathbf{4}, 1, 2, 3])$=$([\mathbf{4}, 1, 3, 2])$,
- $\textsc{Rank}(\Pi')$ is odd
  - If possible, move right:
    $\textsc{Successor}([3, \mathbf{4}, 2, 1])$=$([3, 2, \mathbf{4}, 1])$
  - If n is in last position, get successor of the remaining permutation:
    $\textsc{Successor}([3, 2, 1, \mathbf{4}])$=$([2, 3, 1, \mathbf{4}])$.

We need to be able to determine the parity of $\textsc{Rank}(\Pi')$.

The parity of a permutation is the parity of the number of interchanges necessary for transforming the permutation into $[1, 2, \ldots, n]$.
$\Pi' = [5, 1, 3, 4, 2]$ is an even permutation since 2 steps are sufficient to convert it into $[1, 2, 3, 4, 5]$.

Note that: parity of $\textsc{Rank}(\Pi')$ = parity of Π', since in the Trotter-Johnson algorithm $[1, 2, \ldots, n]$ has rank 0, and each swap increases the rank by 1.

It is easy to compute the parity of a permutation in $\Theta(n^2)$:
$\textsc{PermParity}(n, \Pi) = |\{(i, j) : \Pi[i] > \Pi[j], 1 \leq i \leq j \leq n\}| \mod 2$.

See the textbook for a $\Theta(n)$ algorithm.

```
PermTrotterJohnsonSuccessor($n, \Pi$)
    $s \leftarrow 0$; done $\leftarrow$ false; m $\leftarrow$ n;
    for $i \leftarrow 1$ to $n$ do  $\rho[i] \leftarrow \Pi[i]$;
    while $(m > 1)$ and (not done) do
        $d \leftarrow 1$; while $(\rho[d] \neq m)$ do  $d \leftarrow d + 1$;
        for $i \leftarrow d$ to $m - 1$ do $\rho[i] \leftarrow \rho[i + 1]$;
        par $\leftarrow$ PermParity($m - 1, \rho$);
        if (par $= 1$) then
           if $(d = m)$ then $m \leftarrow m - 1$;
           else swap $\Pi[s + d], \Pi[s + d + 1]$
                done $\leftarrow$ true;
        else if $(d = 1)$ then $m \leftarrow m - 1$; $s \leftarrow s + 1$
             else swap $\Pi[s + d], \Pi[s + d - 1]$
                  done $\leftarrow$ true;
    if $(m = 1)$ then return $[1, 2, \ldots, n]$
                 else return $\Pi$;
```