

# 1-5 Data Structures

魏恒峰

hfwei@nju.edu.cn

2019 年 11 月 14 日



# Generating All Permutations



---

---

```
1: procedure PERMS( $A[], l$ )
2:   if  $l = A.size - 1$  then
3:     print  $A$ 
4:   else
5:     for  $i \leftarrow l$  to  $A.size - 1$  do
6:       SWAP( $A[i], A[l]$ )
7:       PERMS( $A, l + 1$ )
```

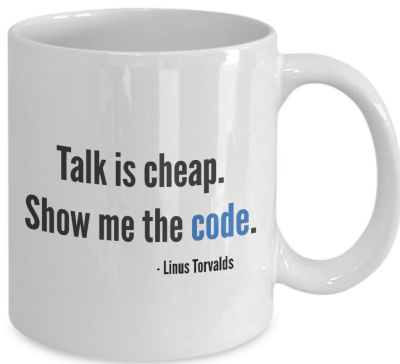
---

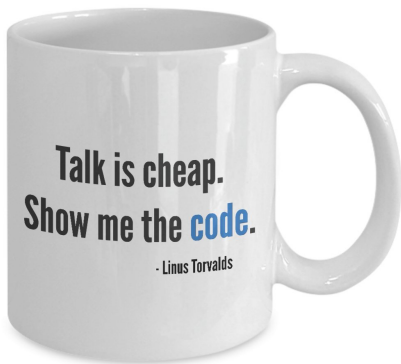
---

---

```
1: procedure PERMS( $A[], l$ )
2:   if  $l = A.size - 1$  then
3:     print  $A$ 
4:   else
5:     for  $i \leftarrow l$  to  $A.size - 1$  do
6:       SWAP( $A[i], A[l]$ )
7:       PERMS( $A, l + 1$ )
8:       SWAP( $A[i], A[l]$ )
```

---



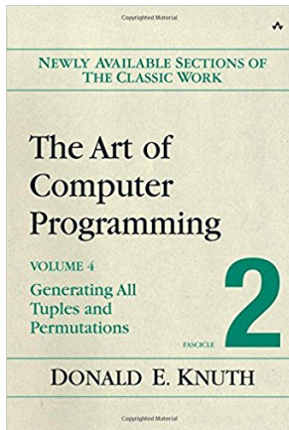


# 大型“车祸”现场



## Iteration Version of PERMS

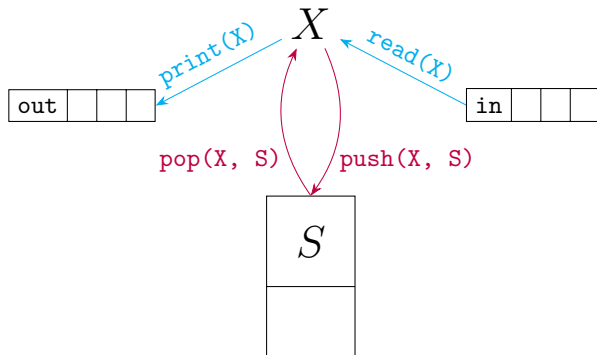
For more about “Generating All Permutations”:





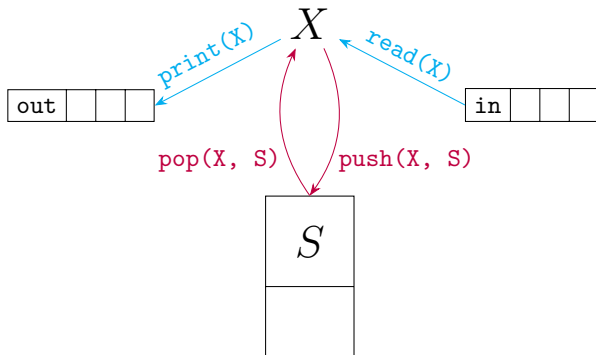
# Stackable Permutations

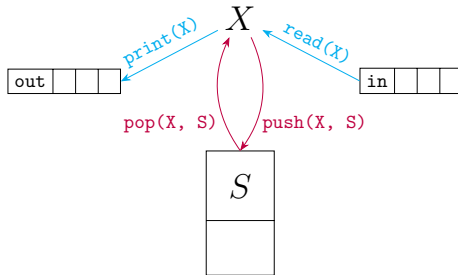
## Definition (Stackable Permutations)



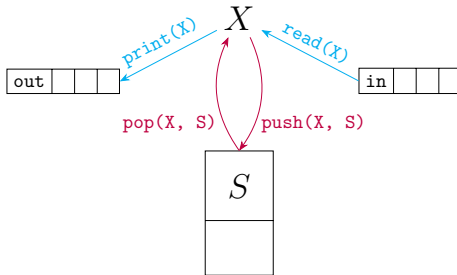
## Definition (Stackable Permutations)

$$\text{out} = (a_1, \dots, a_n) \xleftarrow[\substack{X=0}]{\substack{S=\emptyset}} \text{in} = (1, \dots, n)$$





We can assume that  $X$  is always blank.



We can assume that  $X$  is always blank.

Proof.

What are the possible operations following  $\text{read}(X)/\text{pop}(X, S)$ ?



## DH 2.12: Stackable Permutations

(a) Show that the following permutations *are* stackable:

(i)  $(3, 2, 1)$

(ii)  $(3, 4, 2, 1)$

(iii)  $(3, 5, 7, 6, 8, 4, 9, 2, 10, 1)$

## DH 2.12: Stackable Permutations

(a) Show that the following permutations *are* stackable:

- (i)  $(3, 2, 1)$
- (ii)  $(3, 4, 2, 1)$
- (iii)  $(3, 5, 7, 6, 8, 4, 9, 2, 10, 1)$



## DH 2.13: Stackable Permutations Checking Algorithm

To check whether a given permutation can be obtained by a stack.

read   print   push   pop   is-empty

X = 0      S =  $\emptyset$       in  $\neq$  EOF



## DH 2.13: Stackable Permutations Checking Algorithm

To check whether a given permutation can be obtained by a stack.

read    print    push    pop    is-empty

X = 0    S =  $\emptyset$     in != EOF

```
foreach 'a' in out:
    if (! is-empty(S)
        && 'a' == top(S))
        pop(S, X)
        print(X)
```

## DH 2.13: Stackable Permutations Checking Algorithm

To check whether a given permutation can be obtained by a stack.

read    print    push    pop    is-empty

X = 0    S =  $\emptyset$     in != EOF

```
foreach 'a' in out:
    if (! is-empty(S)
        && 'a' == top(S))
        pop(S, X)
        print(X)
```

```
else // T.B.C
    while (in != EOF)
        read(X)
        if (X == 'a')
            print(X)
            break
        else
            push(X, S)
    if (in == EOF)
        ERR
```

## DH 2.12: Stackable Permutations

(b) **Prove** that the following permutations are *not* stackable:

(i)  $(3, 1, 2)$

(ii)  $(4, 5, 3, 7, 2, 1, 6)$

## DH 2.12: Stackable Permutations

(b) **Prove** that the following permutations are *not* stackable:

(i)  $(3, 1, 2)$

(ii)  $(4, 5, 3, 7, 2, 1, 6)$

$(3, 1, 2)$

$(4, 5, 3, 7, 2, 1, 6)$

## DH 2.12: Stackable Permutations

(b) **Prove** that the following permutations are *not* stackable:

(i)  $(3, 1, 2)$

(ii)  $(4, 5, 3, 7, 2, 1, 6)$

$(3, 1, 2)$

$(4, 5, 3, 7, 2, 1, 6)$

$$\text{out} = \cdots a_i \cdots a_j \cdots a_k : i < j < k \wedge a_j < a_k < a_i$$

## DH 2.12: Stackable Permutations

(b) **Prove** that the following permutations are *not* stackable:

(i)  $(3, 1, 2)$

(ii)  $(4, 5, 3, 7, 2, 1, 6)$

$(3, 1, 2)$

$(4, 5, 3, 7, 2, 1, 6)$

$$\text{out} = \cdots a_i \cdots a_j \cdots a_k : i < j < k \wedge a_j < a_k < a_i$$

312-Pattern

## Theorem (Stackable Permutations)

A permutation  $(a_1, \dots, a_n)$  is stackable  $\iff$  it is not the case that

312-Pattern :  $\text{out} = \dots a_i \dots a_j \dots a_k : i < j < k \wedge a_j < a_k < a_i$

## Theorem (Stackable Permutations)

A permutation  $(a_1, \dots, a_n)$  is stackable  $\iff$  it is not the case that

312-Pattern :  $out = \dots a_i \dots a_j \dots a_k : i < j < k \wedge a_j < a_k < a_i$

Proof.



NO PROOF WARRANTY





## DH 2.12: Stackable Permutations

(c) How many permutations of  $A_4$  *cannot* be obtained by a stack?

$(1, 4, 2, 3), (2, 4, 1, 3), (3, 1, 2, 4), (3, 1, 4, 2), (3, 4, 1, 2)$   
 $(4, 1, 2, 3), (4, 1, 3, 2), (4, 2, 1, 3), (4, 2, 3, 1), (4, 3, 1, 2)$

## DH 2.12: Stackable Permutations

(c) How many permutations of  $A_4$  *cannot* be obtained by a stack?

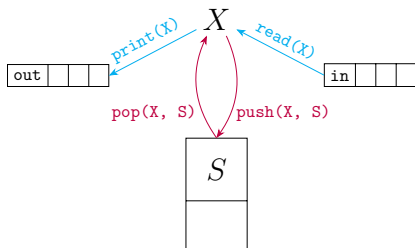
$(1, 4, 2, 3), (2, 4, 1, 3), (3, 1, 2, 4), (3, 1, 4, 2), (3, 4, 1, 2)$   
 $(4, 1, 2, 3), (4, 1, 3, 2), (4, 2, 1, 3), (4, 2, 3, 1), (4, 3, 1, 2)$

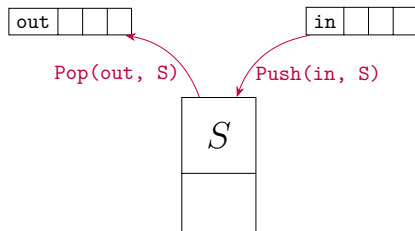
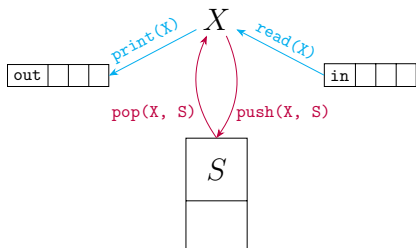
## DH 2.12: Stackable Permutations

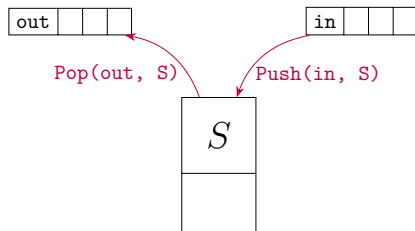
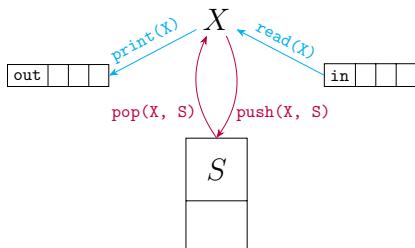
(c) How many permutations of  $A_4$  *cannot* be obtained by a stack?

$(1, 4, 2, 3), (2, 4, 1, 3), (3, 1, 2, 4), (3, 1, 4, 2), (3, 4, 1, 2)$   
 $(4, 1, 2, 3), (4, 1, 3, 2), (4, 2, 1, 3), (4, 2, 3, 1), (4, 3, 1, 2)$

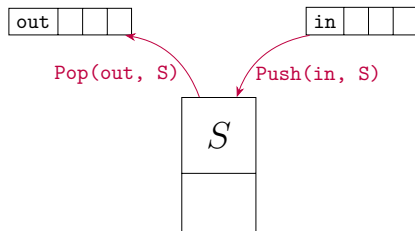
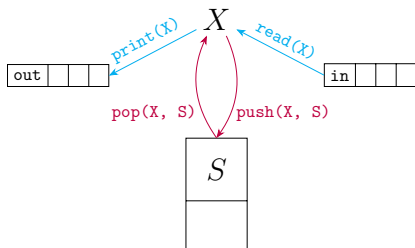
*Q* : What about  $A_n$ ?





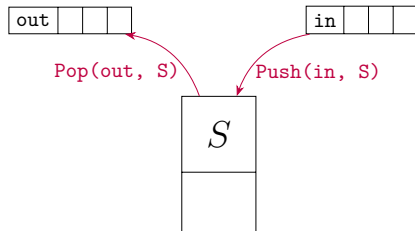
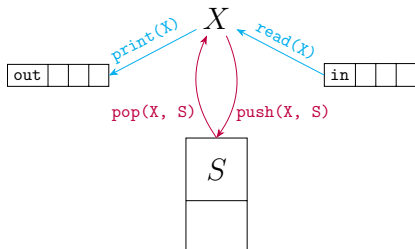


$Q$  : Are  $S + X$  and  $S$  are **equivalent**?

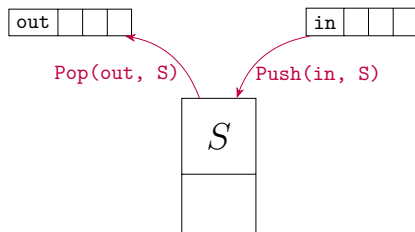
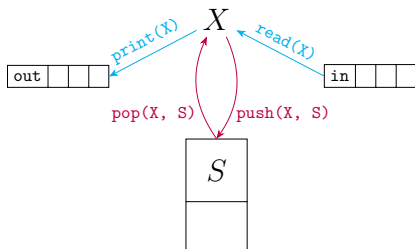


$Q$  : Are  $S + X$  and  $S$  are **equivalent**?

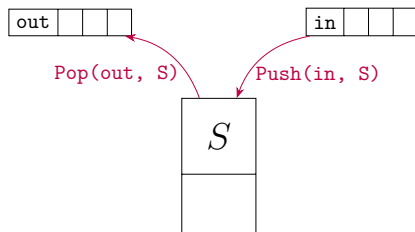
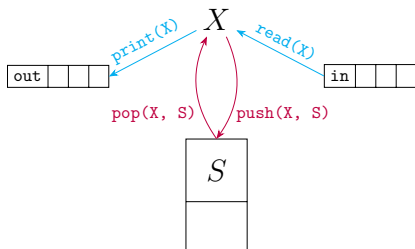
Producing the same set of permutations.







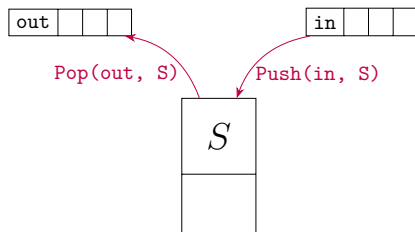
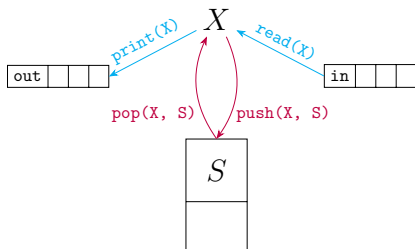
By simulations.



By simulations.

Simulate  $S$  by  $S + X$ :

- Push
- Pop

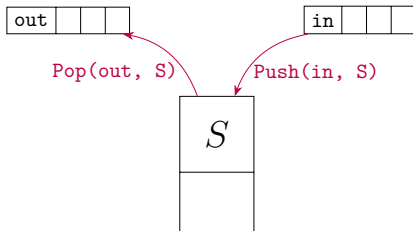
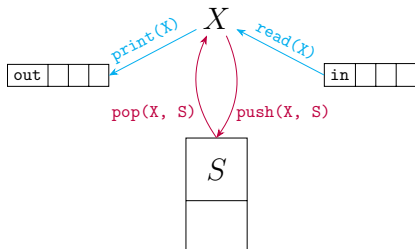


By simulations.

Simulate  $S$  by  $S + X$ :

- ▶ Push
- ▶ Pop

Simulate  $S + X$  by  $S$ :



By simulations.

Simulate  $S$  by  $S + X$ :

- ▶ Push
- ▶ Pop

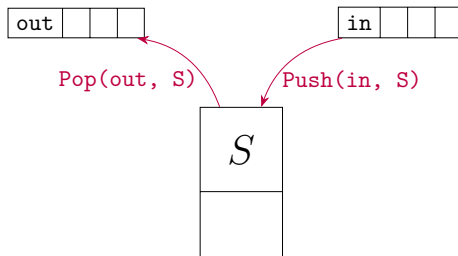
Simulate  $S + X$  by  $S$ :

By iterative transformations.



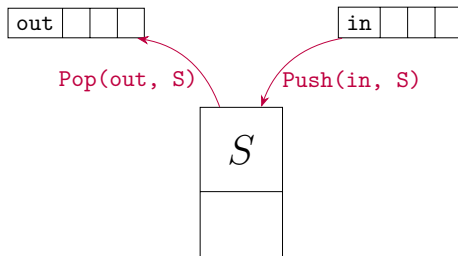
## DH 2.12: Stackable Permutations

How many permutations of  $\{1 \cdots n\}$  are stackable on the model  $S$ ?



## DH 2.12: Stackable Permutations

How many permutations of  $\{1 \cdots n\}$  are stackable on the model  $S$ ?



$Q$  : How many *admissible* operation sequences of “Push” and “Pop”?

## Definition (Admissible Operation Sequences)

An operation sequence of “Push” and “Pop” is *admissible* if and only if

## Definition (Admissible Operation Sequences)

An operation sequence of “Push” and “Pop” is *admissible* if and only if

(i)  $\# \text{ of “Push”} = n$        $\# \text{ of “Pop”} = n$



## Definition (Admissible Operation Sequences)

An operation sequence of “Push” and “Pop” is *admissible* if and only if

- (i) # of “Push” =  $n$       # of “Pop” =  $n$
- (ii)  $\forall$  prefix : (# of “Pop”)  $\leq$  (# of “Push”)

## Definition (Admissible Operation Sequences)

An operation sequence of “Push” and “Pop” is *admissible* if and only if

- (i) # of “Push” =  $n$       # of “Pop” =  $n$
- (ii)  $\forall$  prefix : (# of “Pop”)  $\leq$  (# of “Push”)

# of stackable perms = # of admissible operation sequences

## Theorem

*Different admissible operation sequences correspond to different permutations.*

## Theorem

*Different admissible operation sequences correspond to different permutations.*

## Proof.

Push Push Push Pop Pop **Push**...

Push Push Push Pop Pop **Pop**...



## Theorem

The number of admissible operation sequences of “*Push*” and “*Pop*” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .

## Theorem

*The number of admissible operation sequences of “Push” and “Pop” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .*

Proof: The Reflection Method.

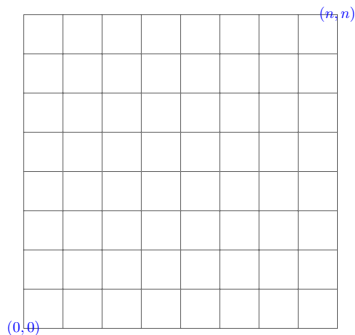
Push :  $\rightarrow$       Pop :  $\uparrow$

## Theorem

The number of admissible operation sequences of “*Push*” and “*Pop*” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .

Proof: The Reflection Method.

Push :  $\rightarrow$       Pop :  $\uparrow$

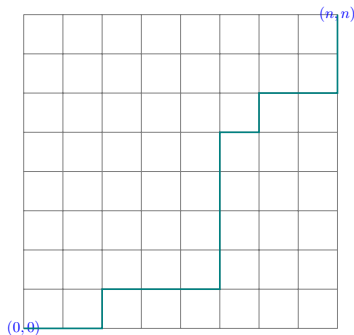


## Theorem

The number of admissible operation sequences of “*Push*” and “*Pop*” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .

Proof: The Reflection Method.

Push :  $\rightarrow$       Pop :  $\uparrow$



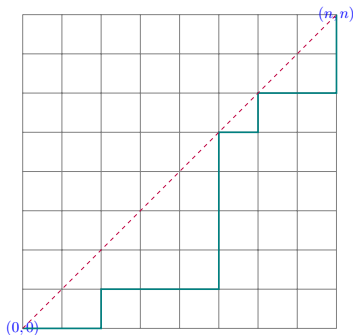


## Theorem

The number of admissible operation sequences of “*Push*” and “*Pop*” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .

Proof: The Reflection Method.

Push :  $\rightarrow$       Pop :  $\uparrow$

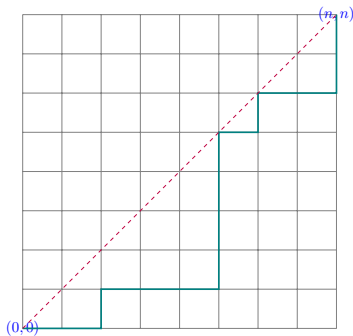


## Theorem

The number of admissible operation sequences of “*Push*” and “*Pop*” is  $\binom{2n}{n} - \binom{2n}{n-1}$ .

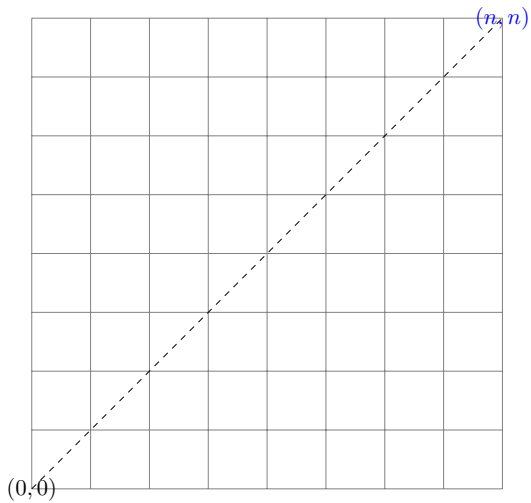
Proof: The Reflection Method.

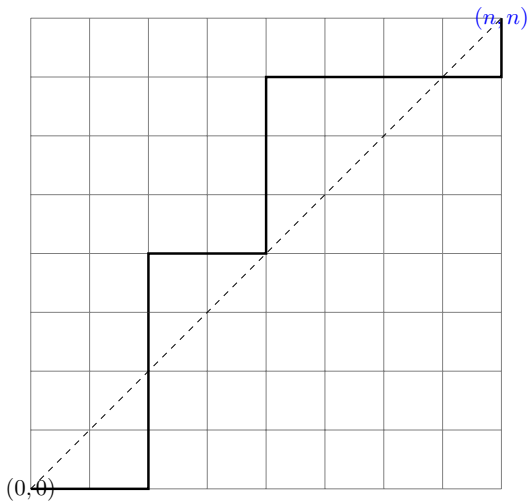
Push :  $\rightarrow$       Pop :  $\uparrow$

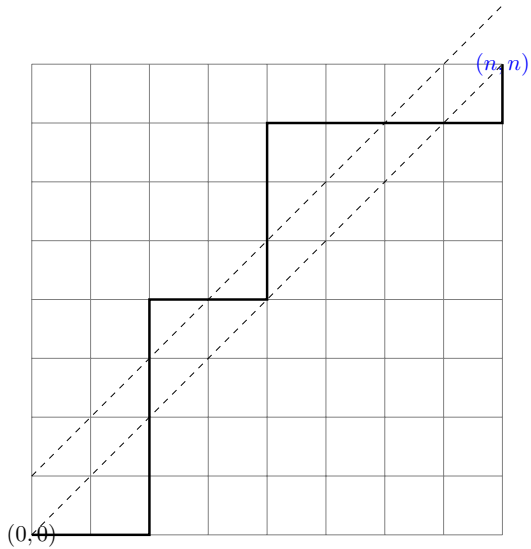


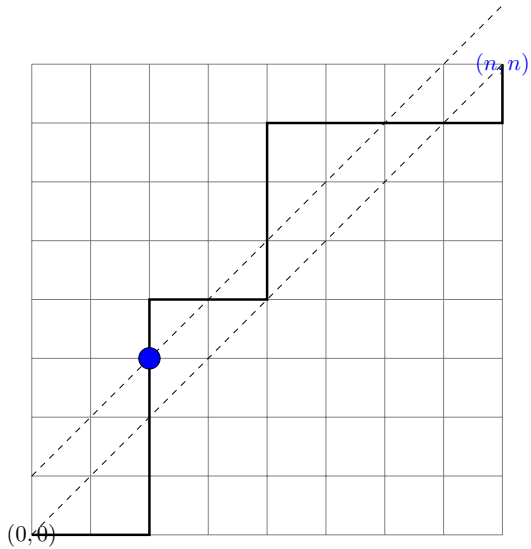
$$\underbrace{\binom{2n}{n}}_{\text{all}} - \underbrace{\binom{2n}{n-1}}_{\text{inadmissible}}$$

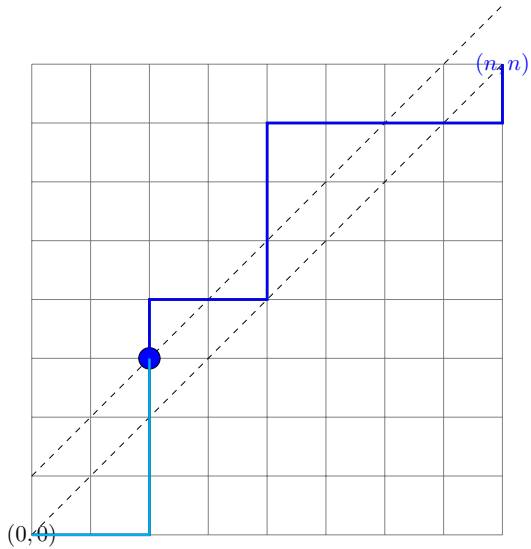


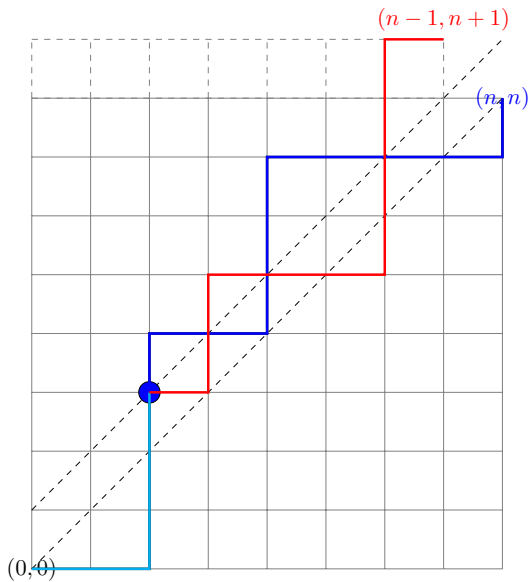




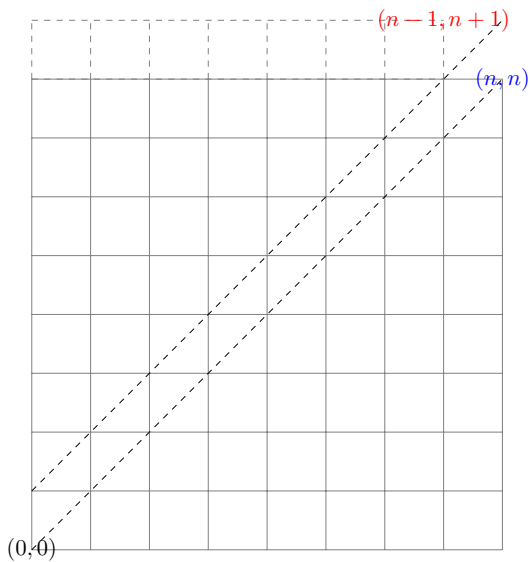


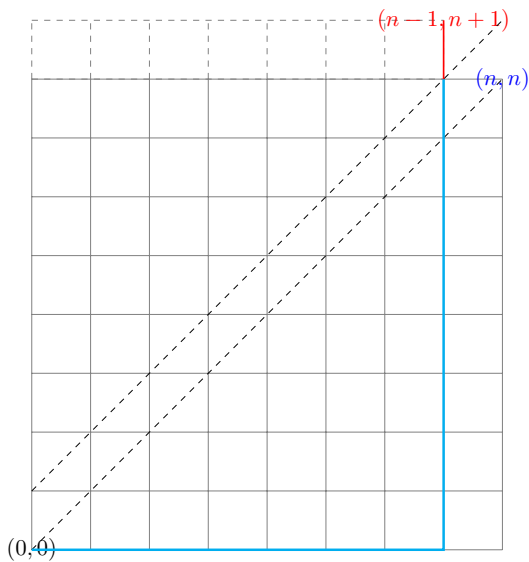


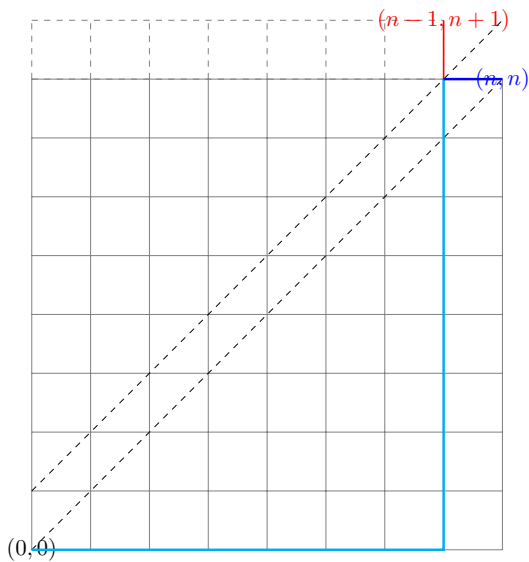


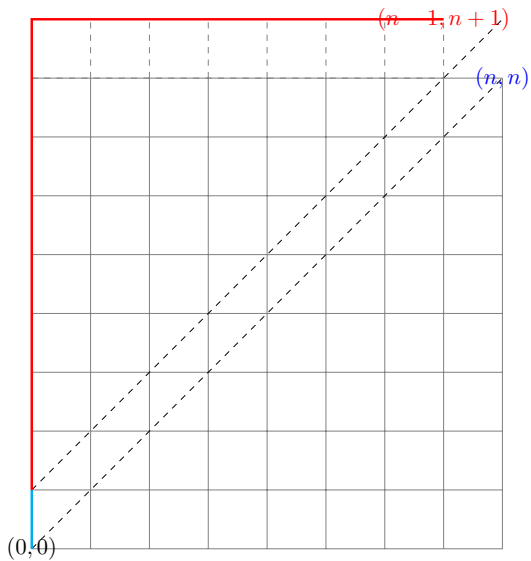


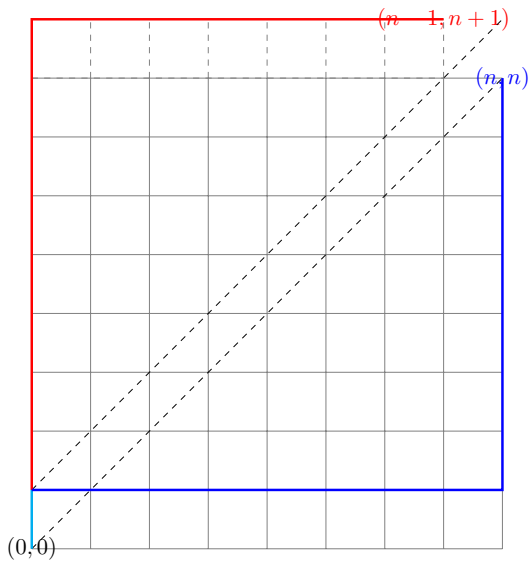








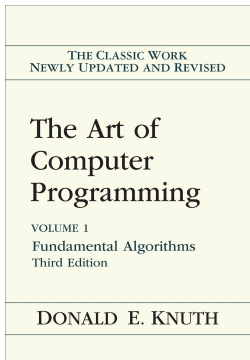




# Catalan Number

$(3, 2, 1) : (((()))$        $(1, 2, 3) : ()()()$

For more about “Stackable Permutations” (Section 2.2.1)



Thank  
You!