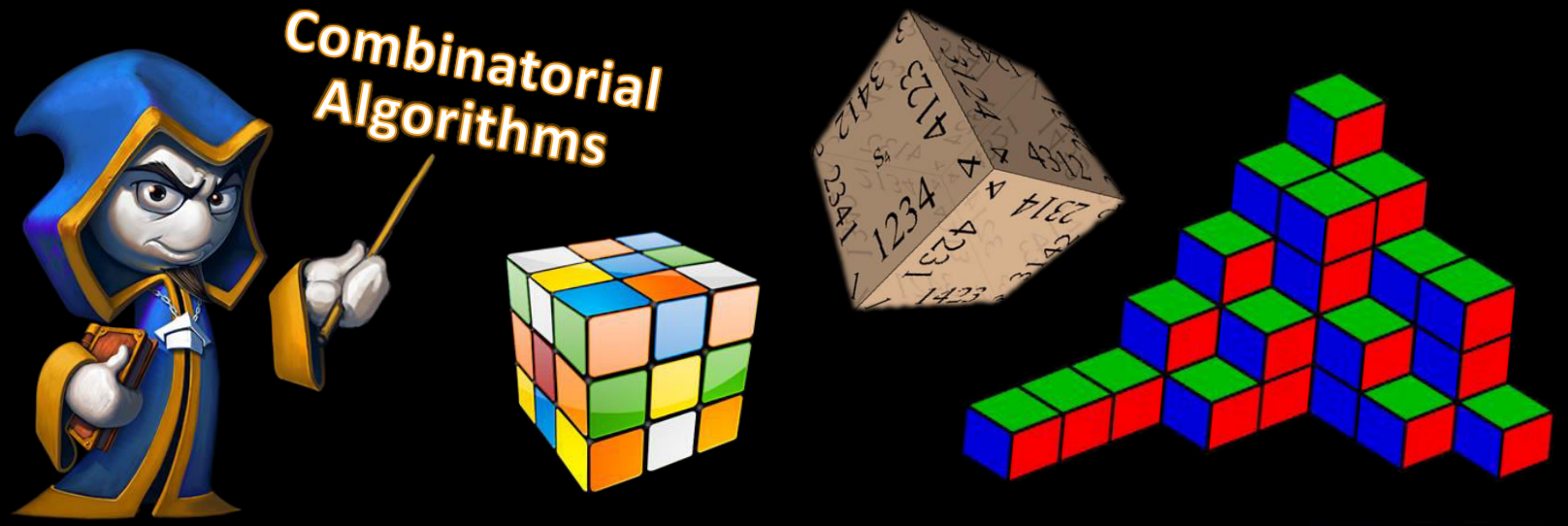
Combinatorial Algorithms

Generating Permutations, Variations, Combinations, …



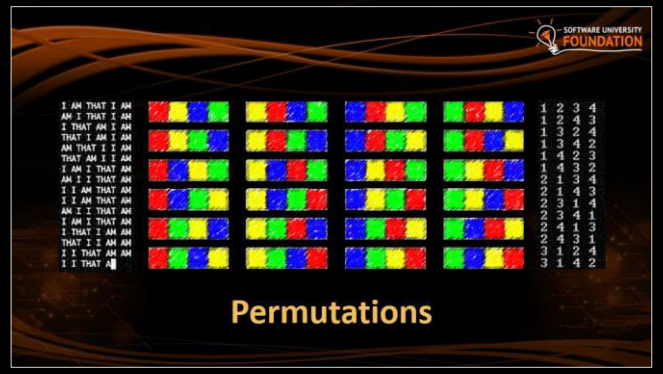
SoftUni Team

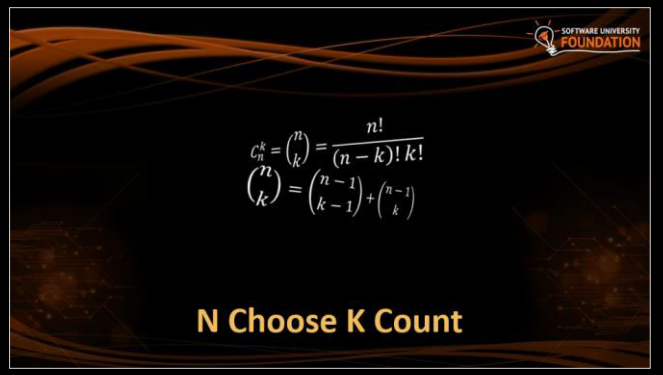
Technical Trainers

Software University

http://softuni.bg

Table of Contents





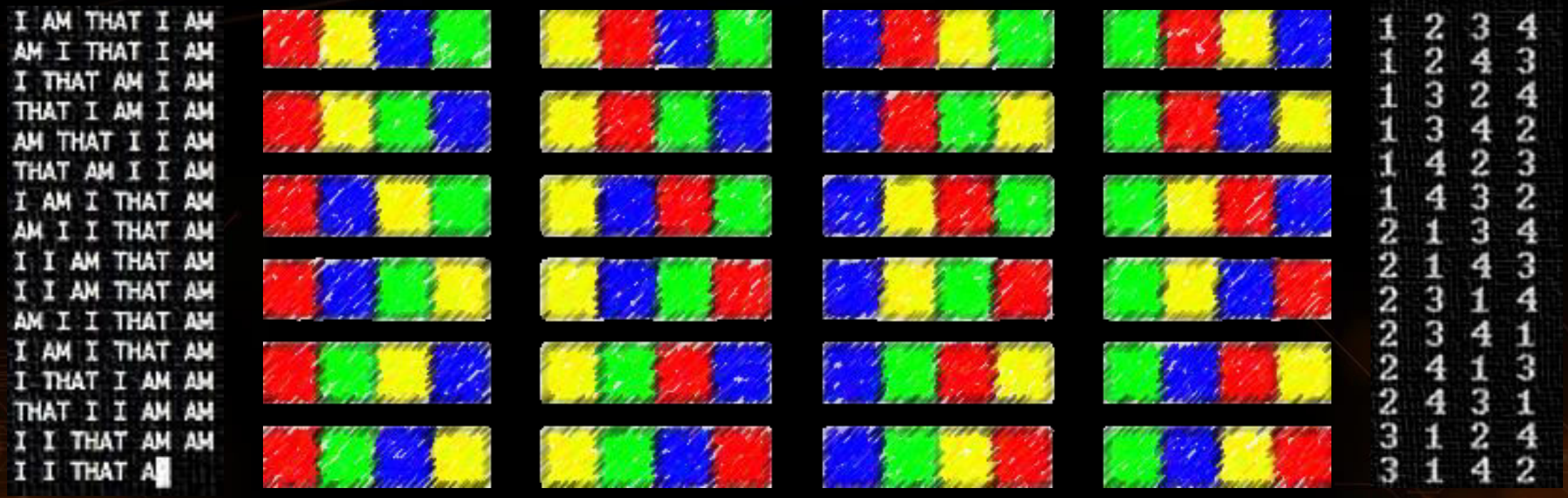
Have a Question?

SOFTWARE UNIVERSITY FOUNDATION

sli.do

#DsAlgo

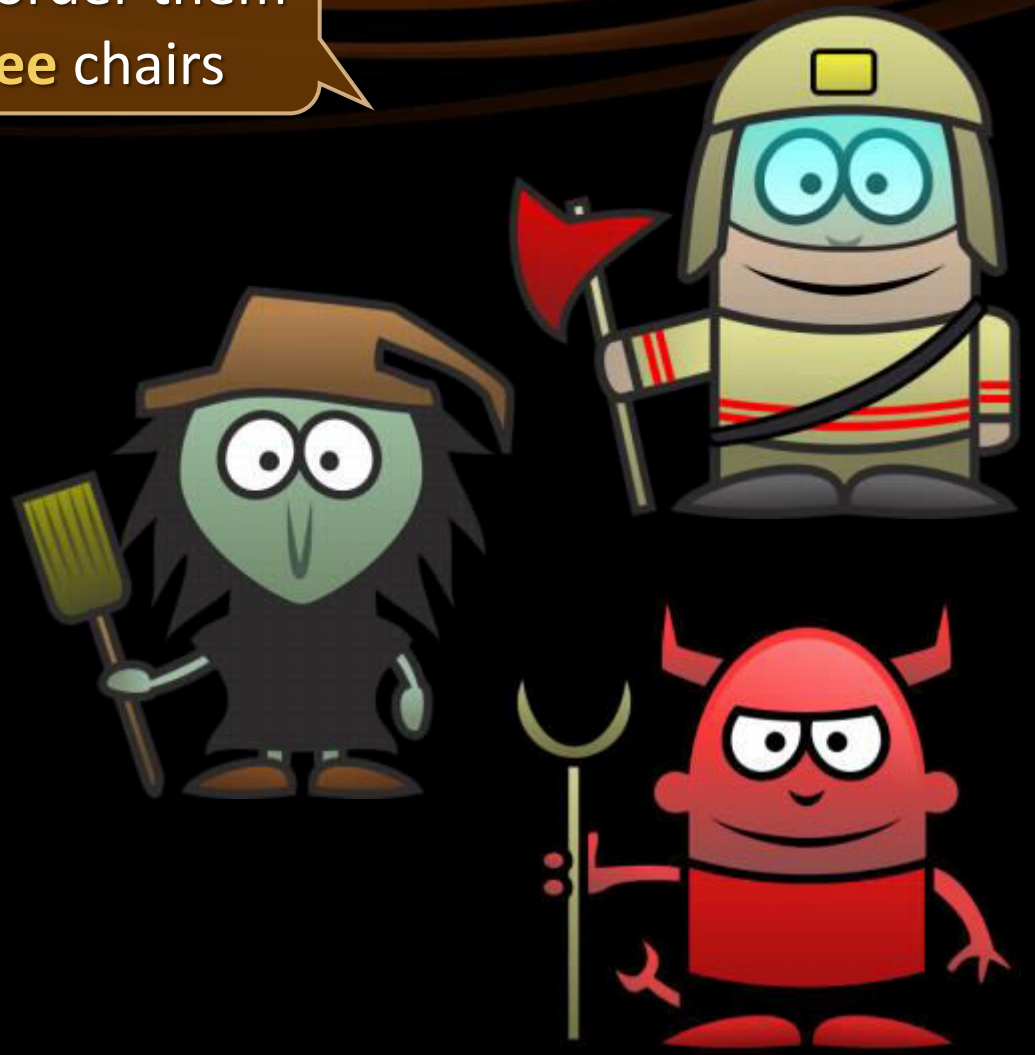
SOFTWARE UNIVERSITY FOUNDATION



Permutations

Permutations

How to order them in **three** chairs



Permutations

How to order them in **two** chairs

Permutations

How to order him
in **one** chairs

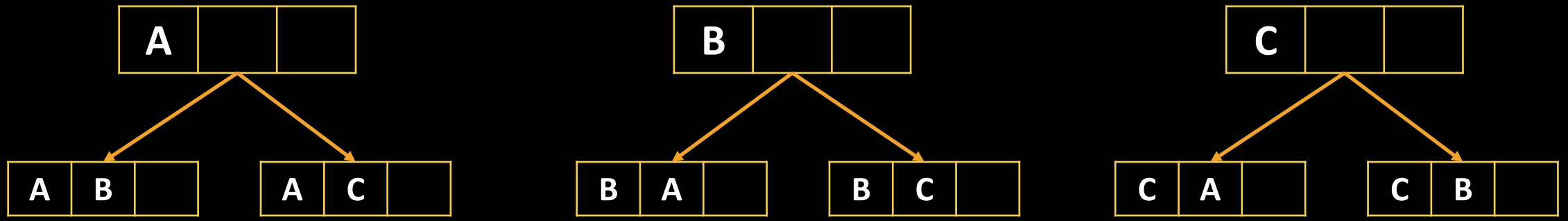
Permutations

Permutations

Order **A**, **B** and **C** in all possible ways

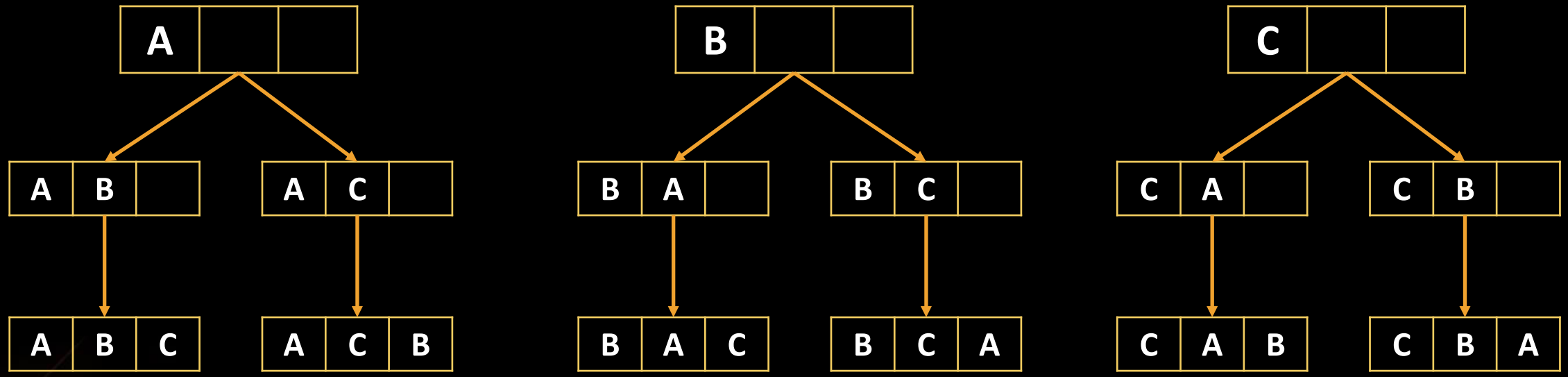
Permutations

Order **A**, **B** and **C** in all possible ways



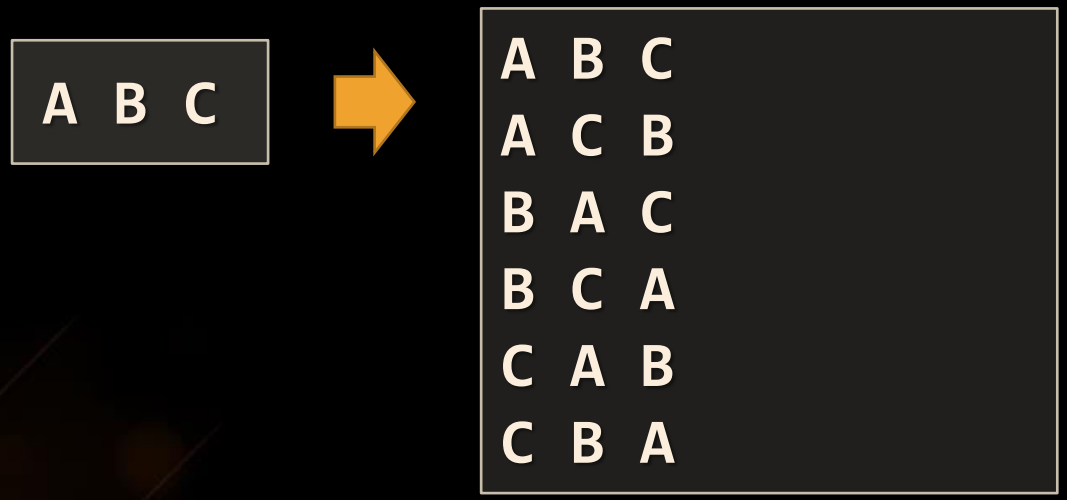
Permutations

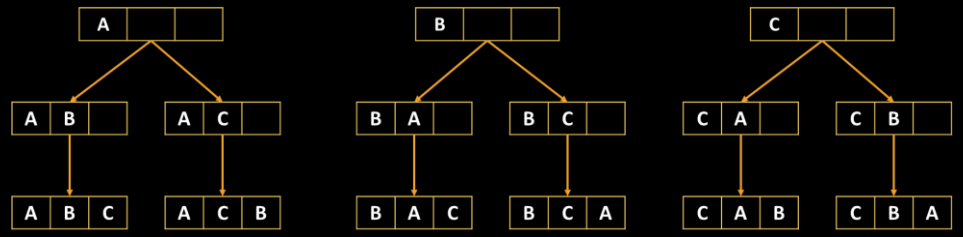
Order **A**, **B** and **C** in all possible ways



Problem: Generate Permutations

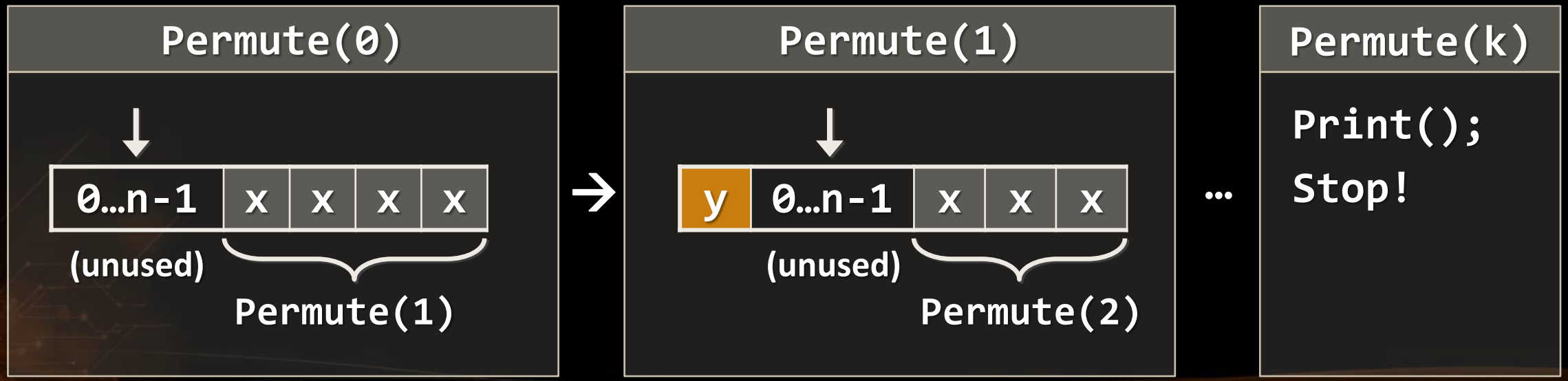
- Generates all possible **permutations** of a given set of elements
- You can **pick** each **item only once**





Algorithm: Permutations

- Algorithm **`Permute(index)`** to generate variations **`P(n)`**
  - Put **unused** elements **`0`** … **`n-1`** at position **`index`**
  - Mark/unmark elements as **being used**
  - Call **`Permute(index + 1)`** to generate the rest of the array



Generating Permutations

SOFTWARE UNIVERSITY FOUNDATION

```
public static void Gen(int index)
{
  if (index >= elements.Length)
    Console.WriteLine(string.Join(" ", perm));
  else
    for (int i = 0; i < elements.Length; i++)
      if (!used[i])
      {
        used[i] = true;
        perm[index] = elements[i];
        Gen(index + 1);
        used[i] = false;
      }
}
```

```
elements = new int[n];
used = new bool[n];
Permute(0);
```

Permutations Count

Order **A**, **B** and **C** in all possible ways
How many ways are there?

A	B	C

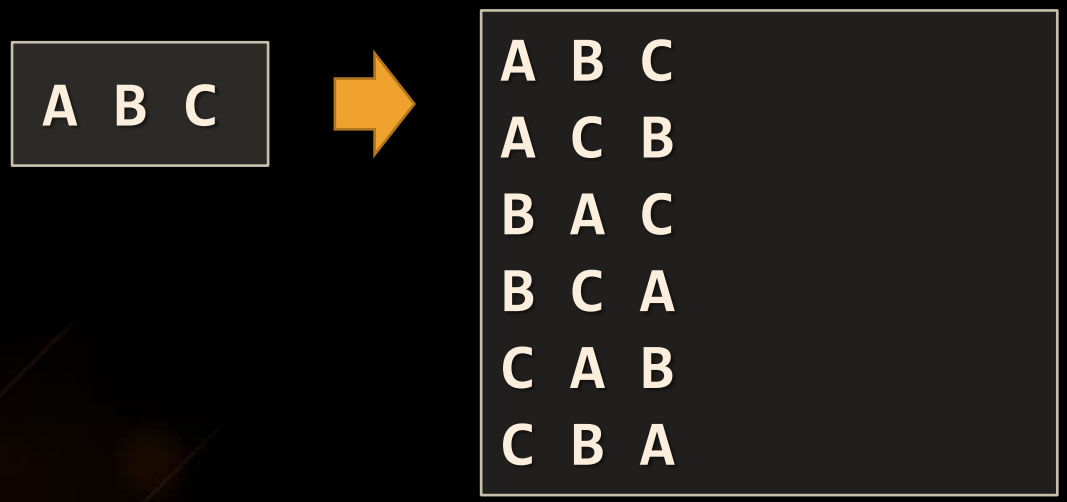
3	2	1

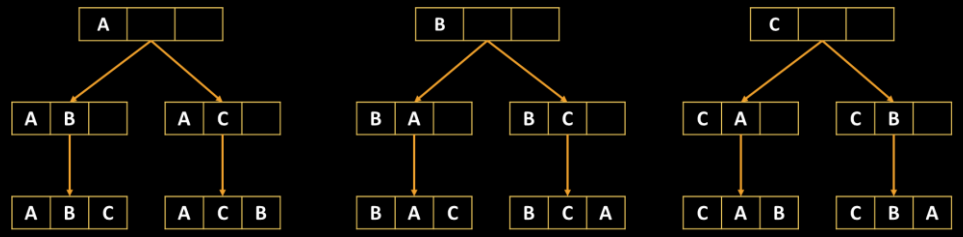
$n! = 3!$

6 possible ways

Problem: Optimize Permutations

- Generates all possible **permutations** of a given set of elements
  - **Without** using **extra memory**

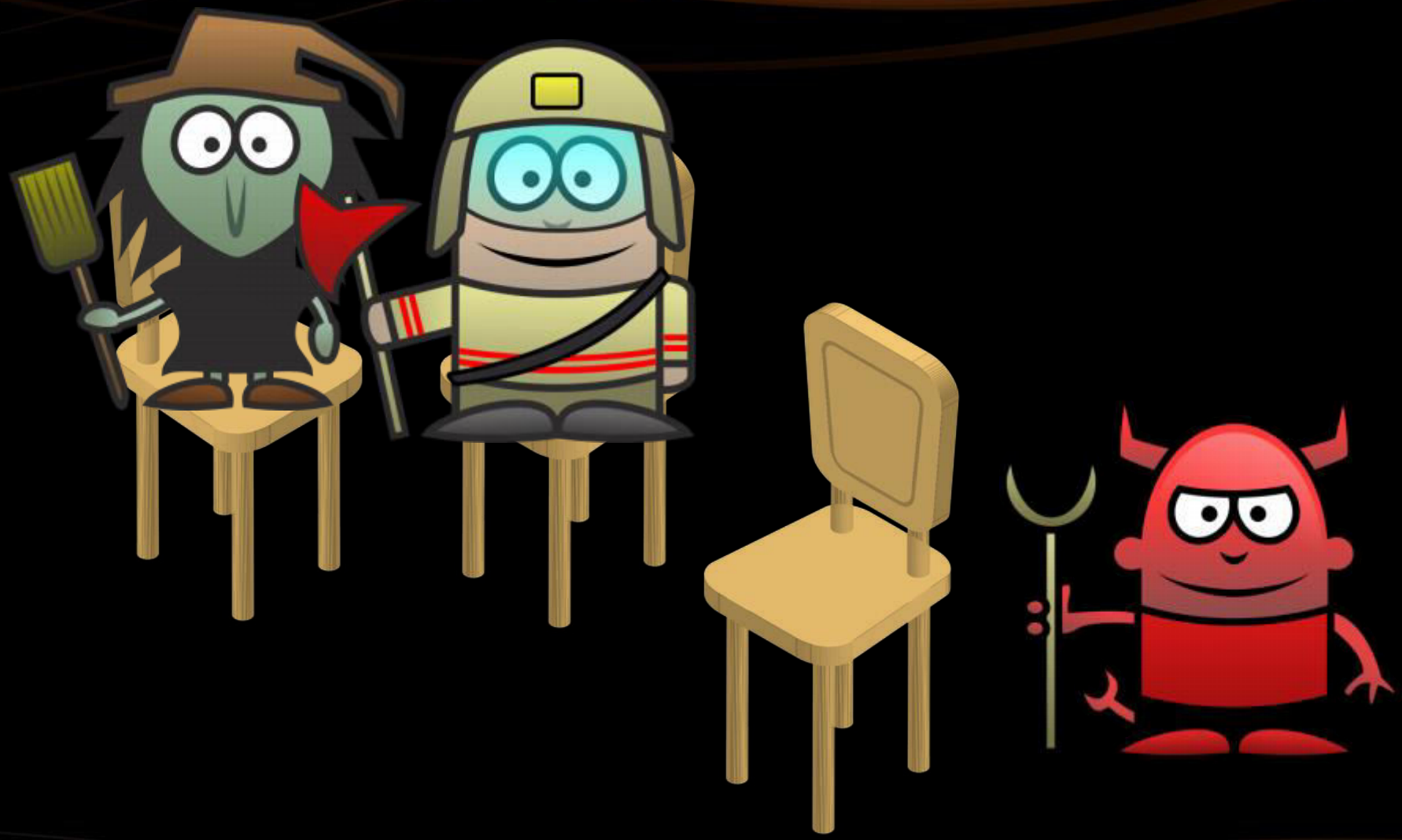




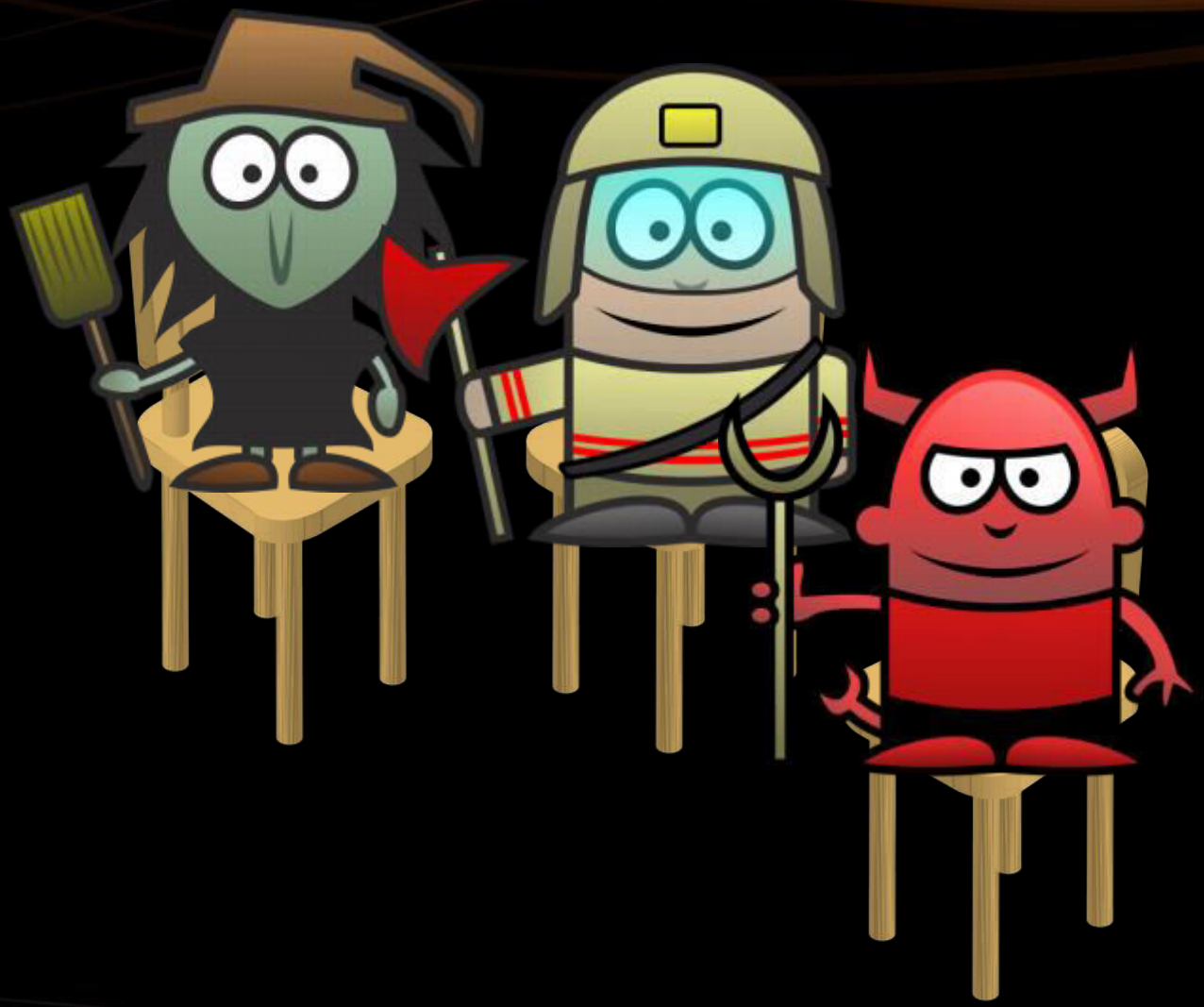
Permutations: Swap Algorithm

Permutations: Swap Algorithm

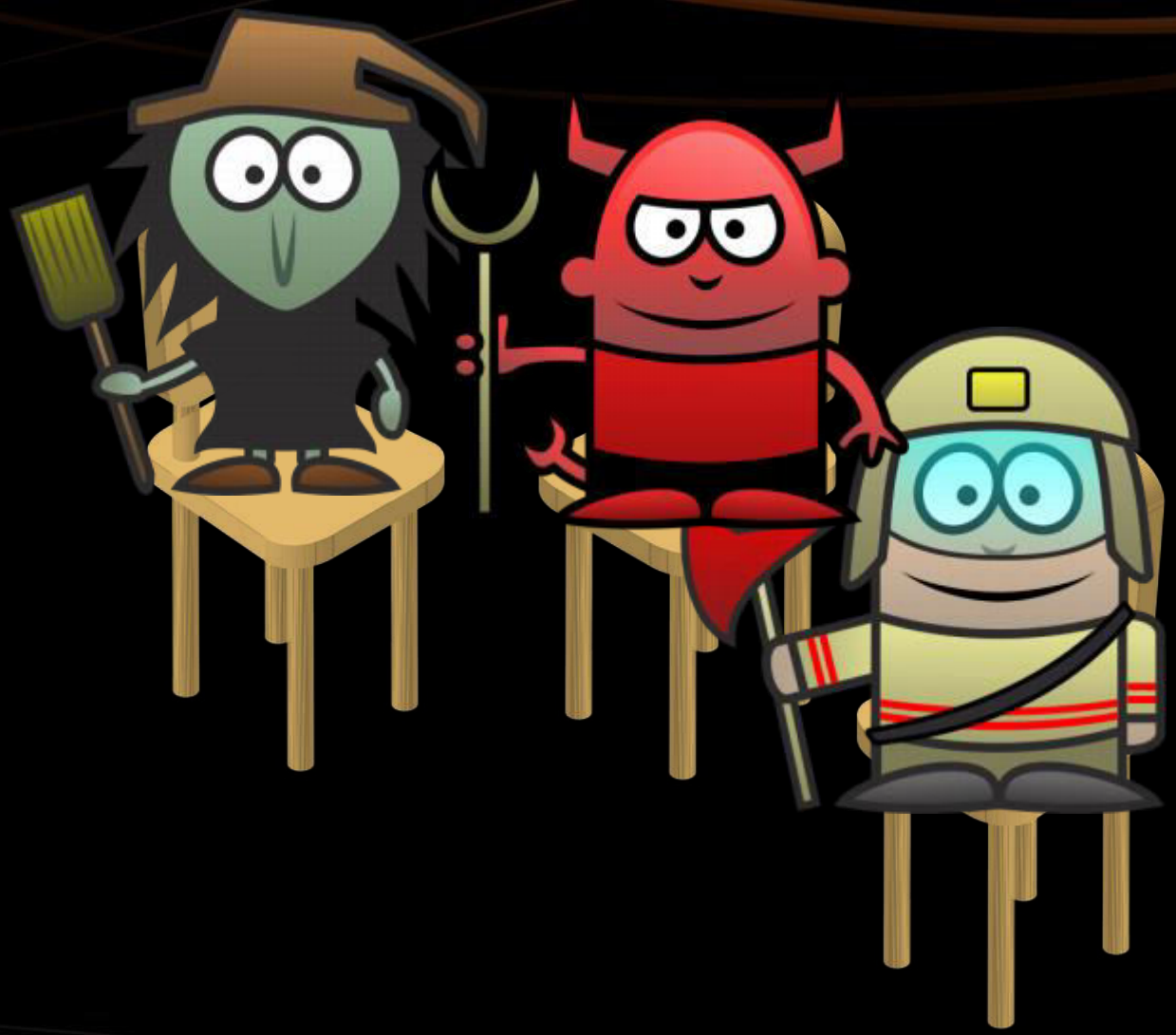
Permutations: Swap Algorithm

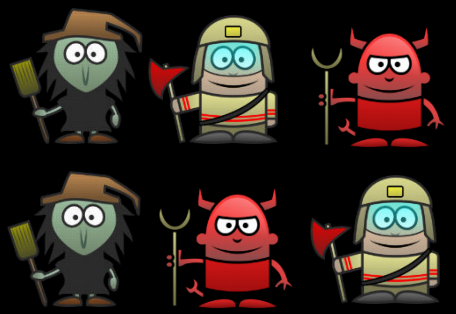


Permutations: Swap Algorithm

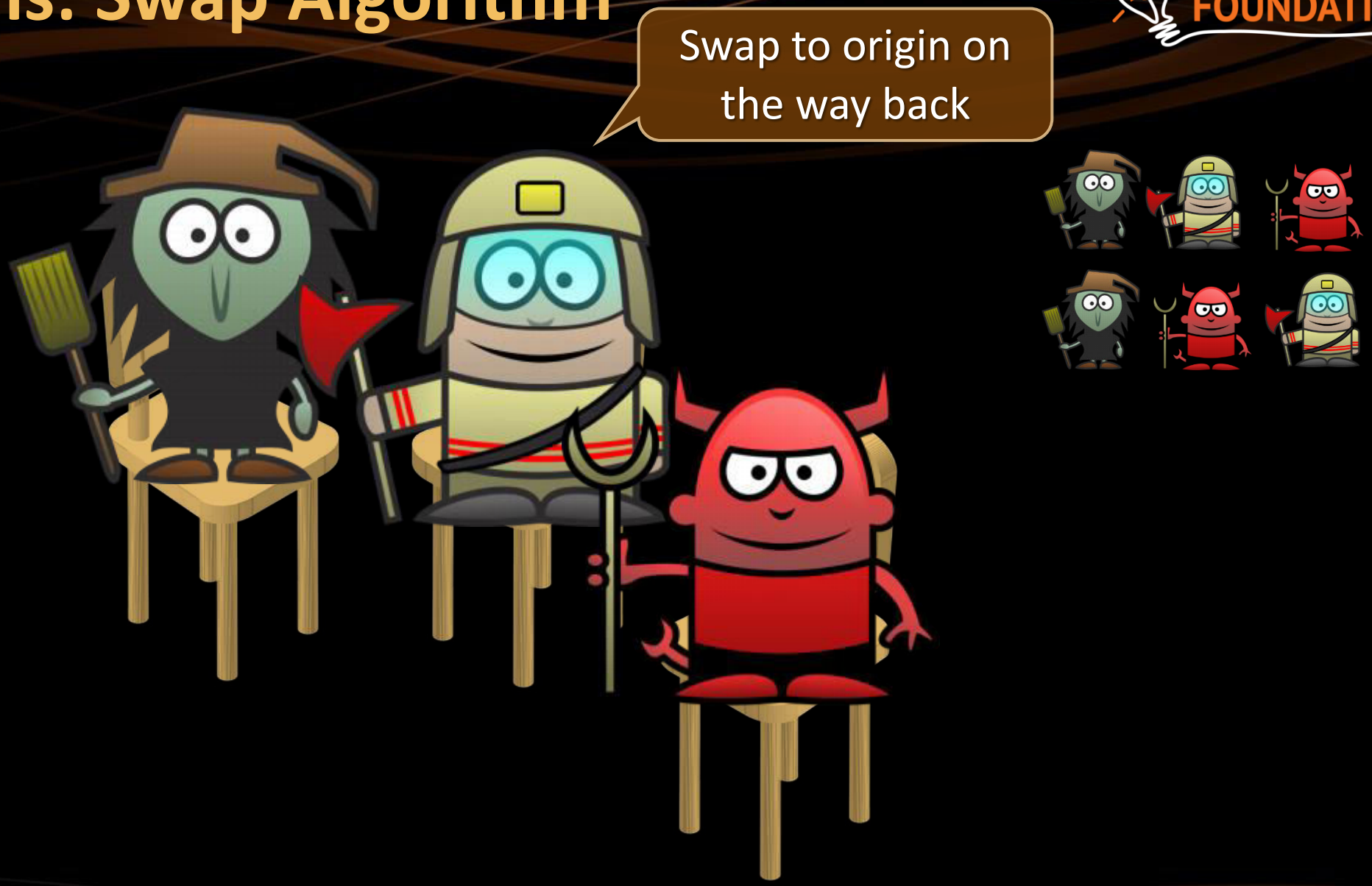


Permutations: Swap Algorithm

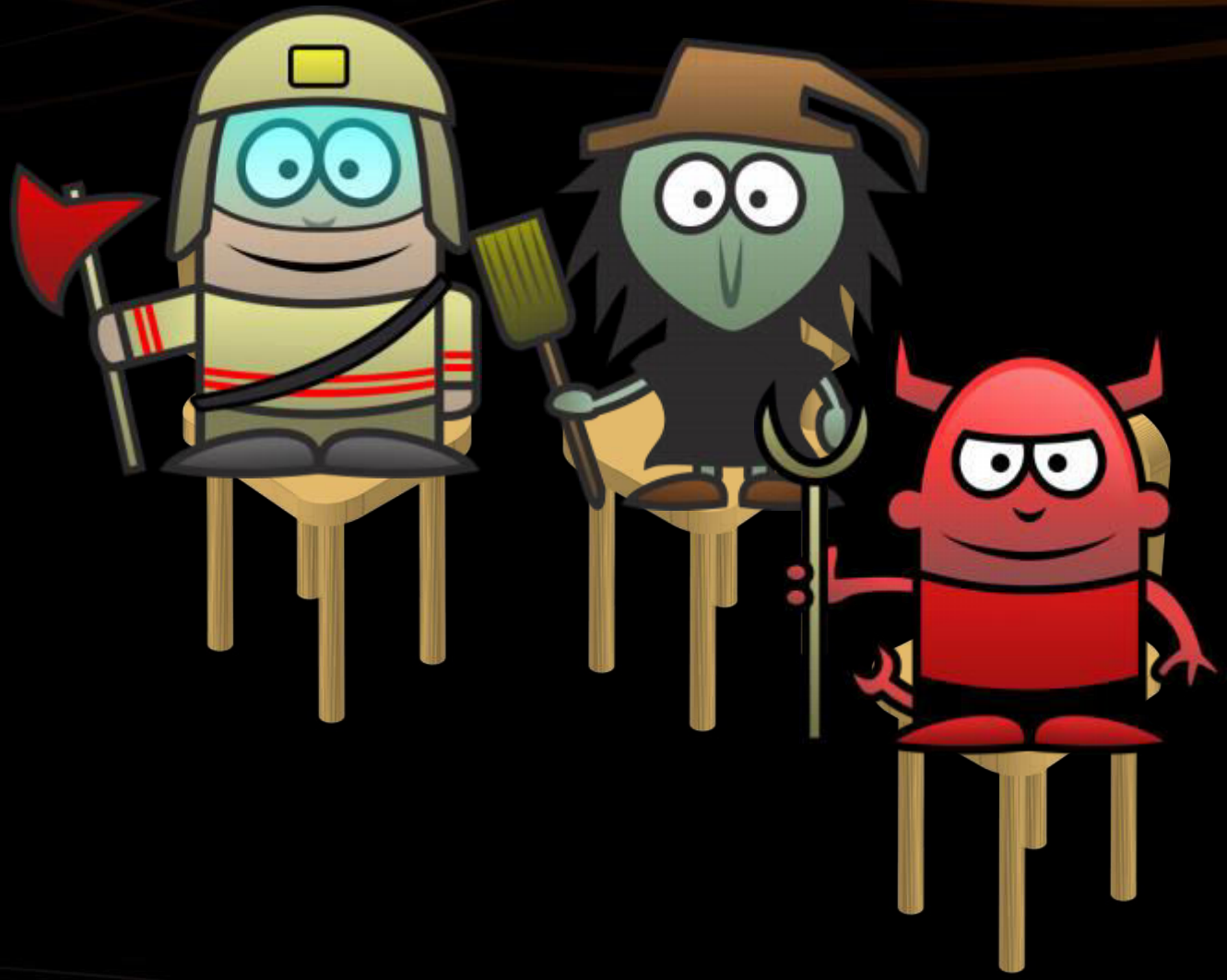


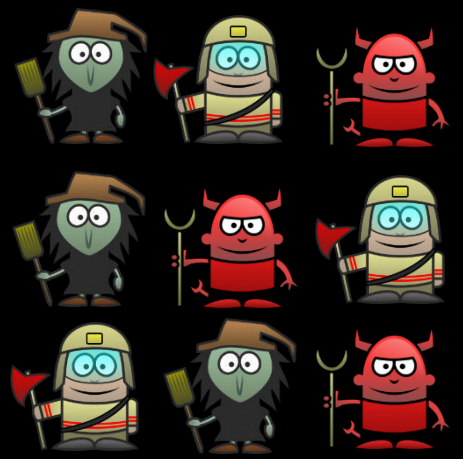


Permutations: Swap Algorithm

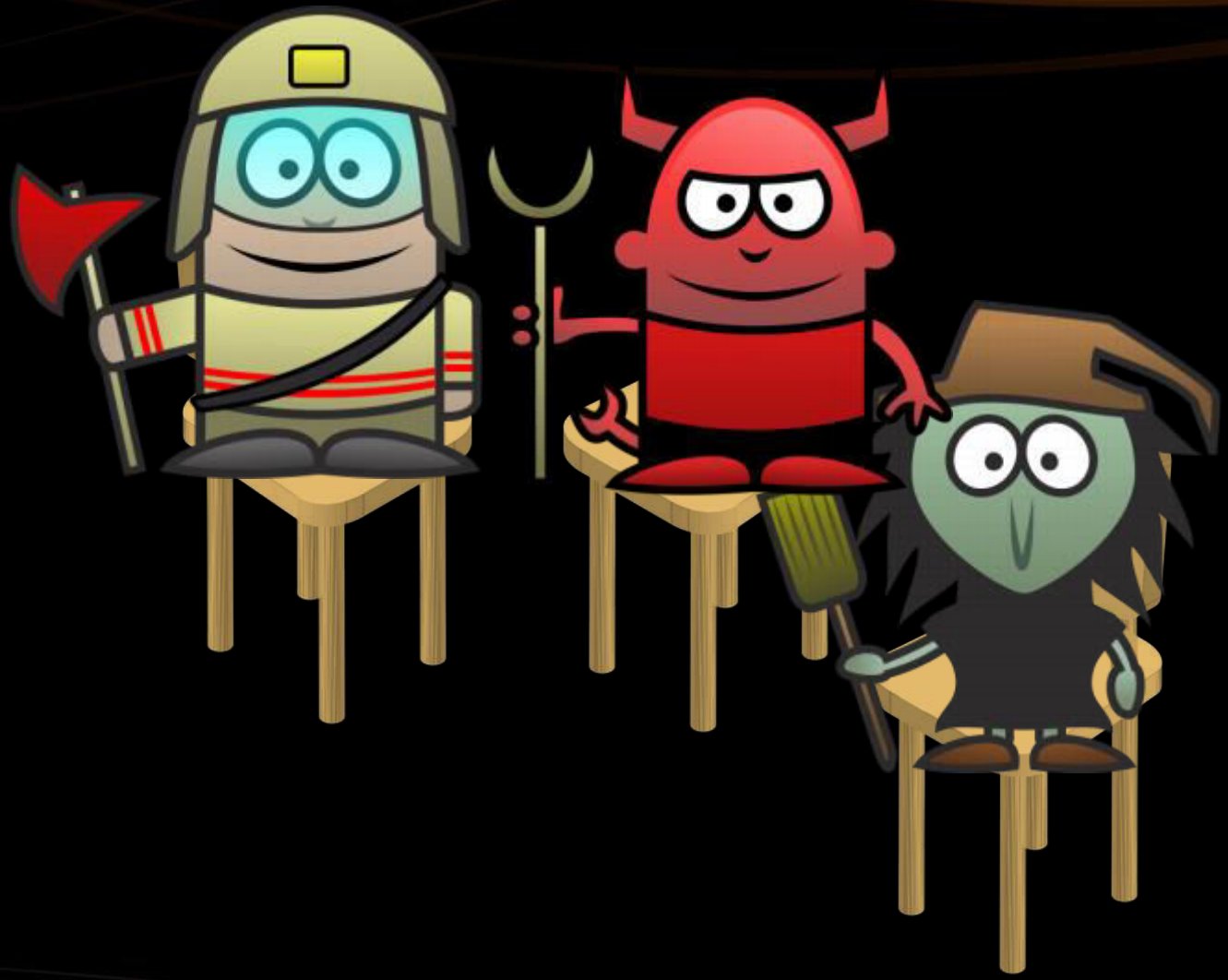


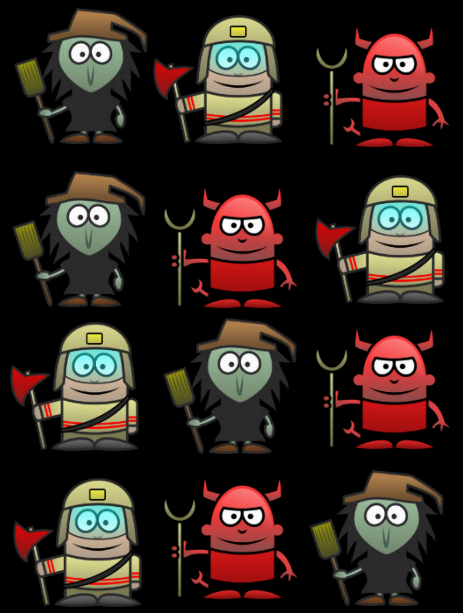
Permutations: Swap Algorithm



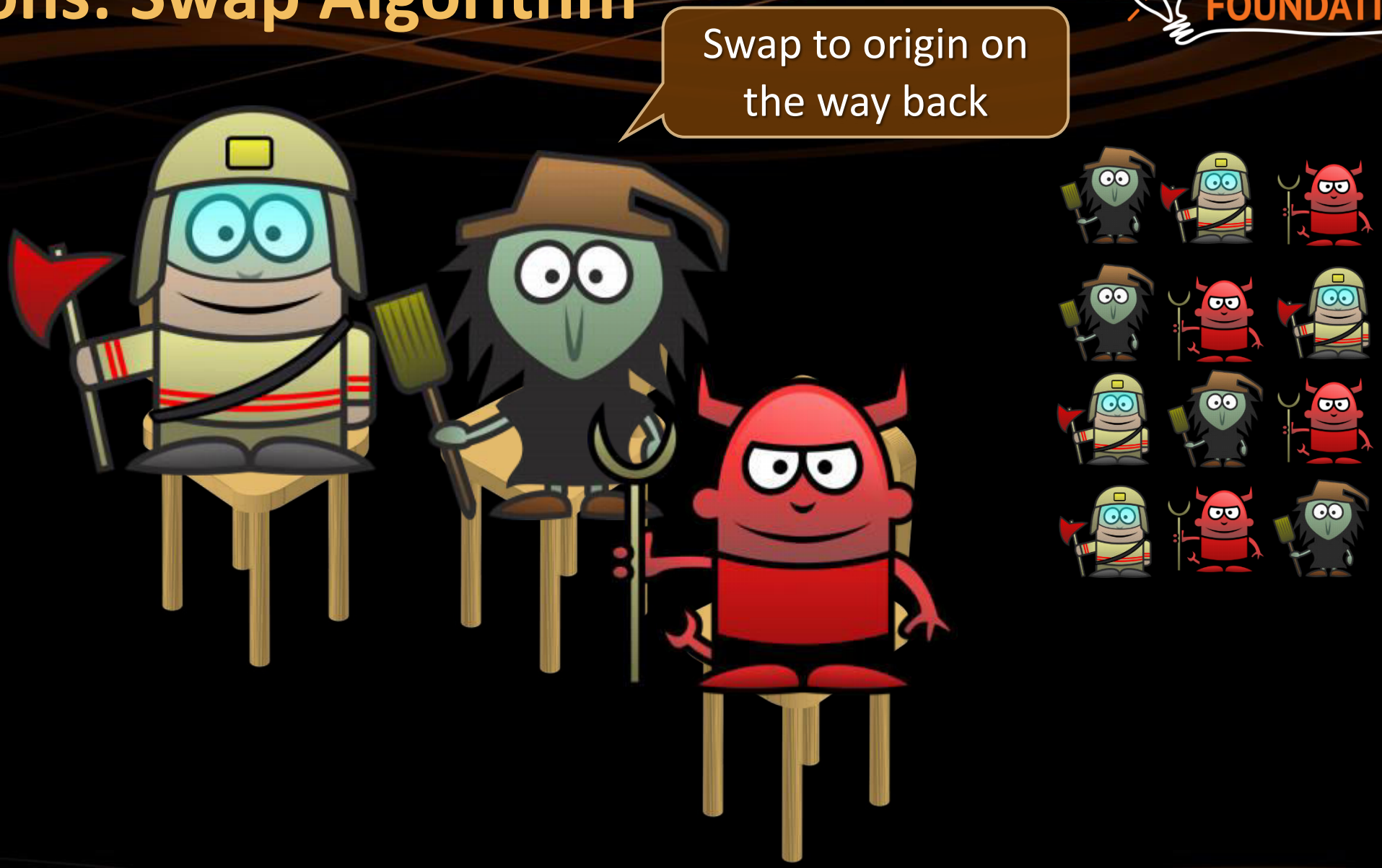


Permutations: Swap Algorithm



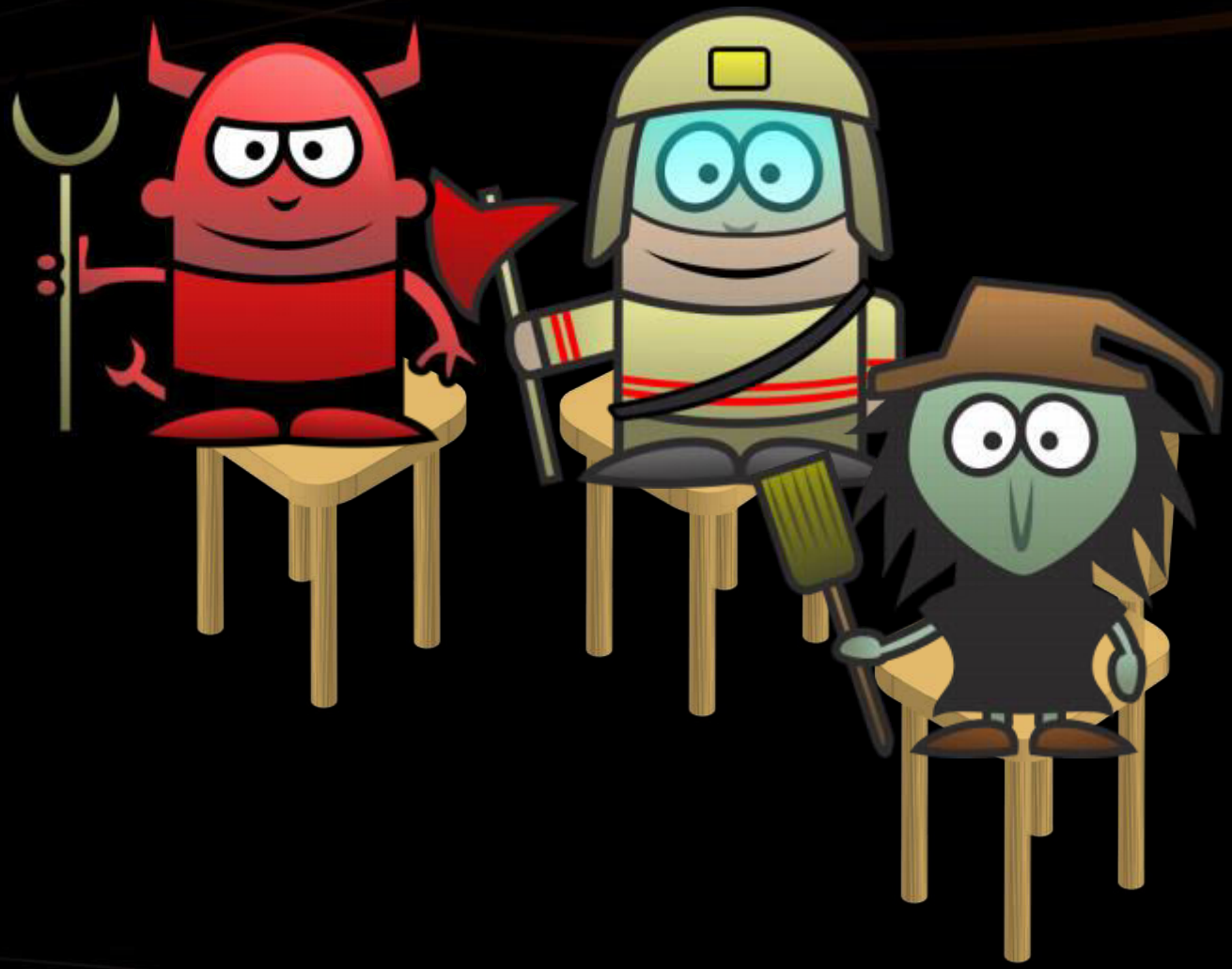


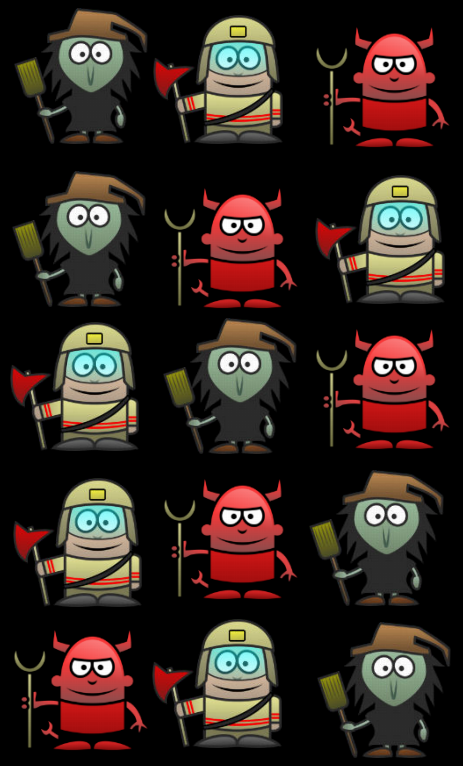
Permutations: Swap Algorithm



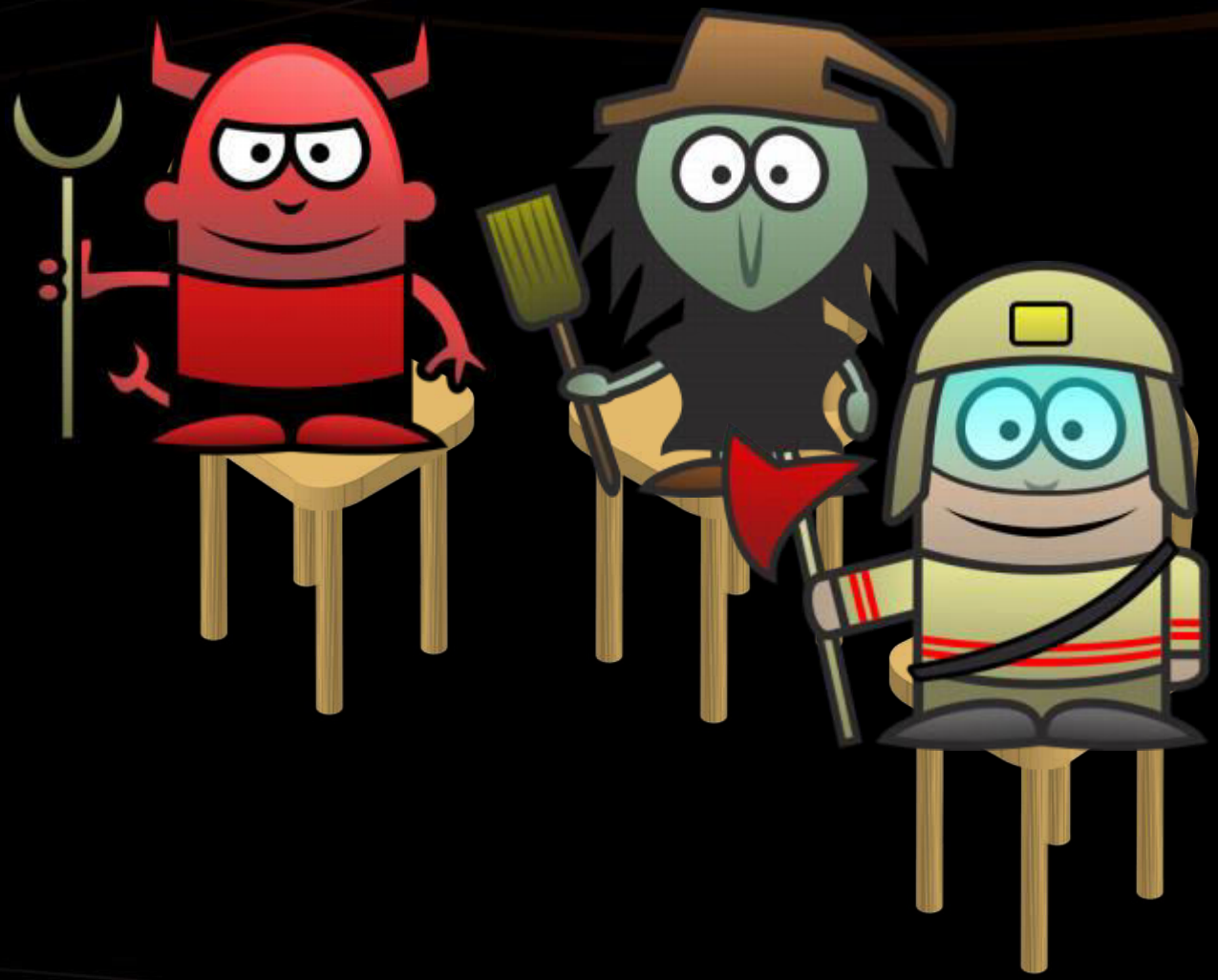
Permutations: Swap Algorithm

Permutations: Swap Algorithm

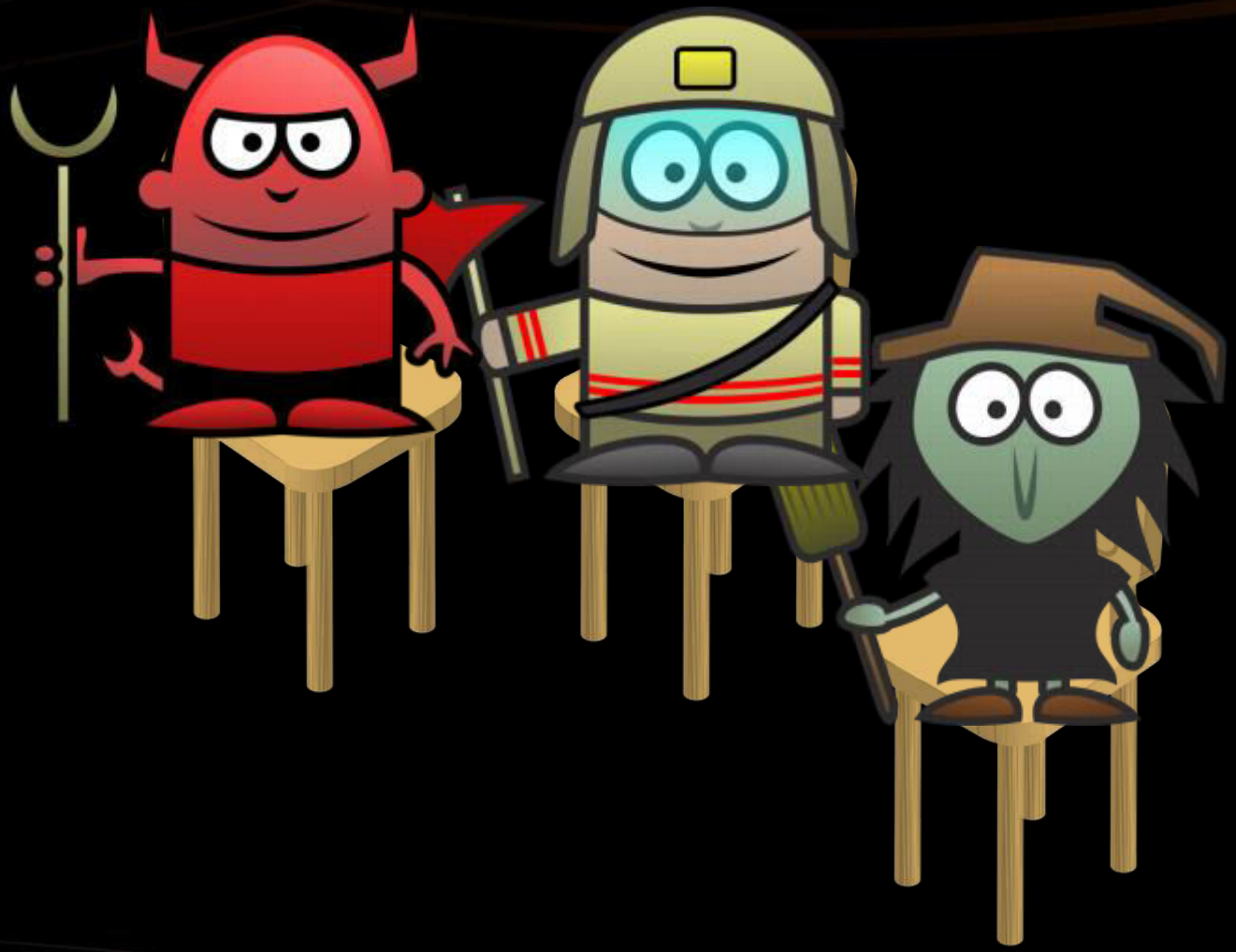




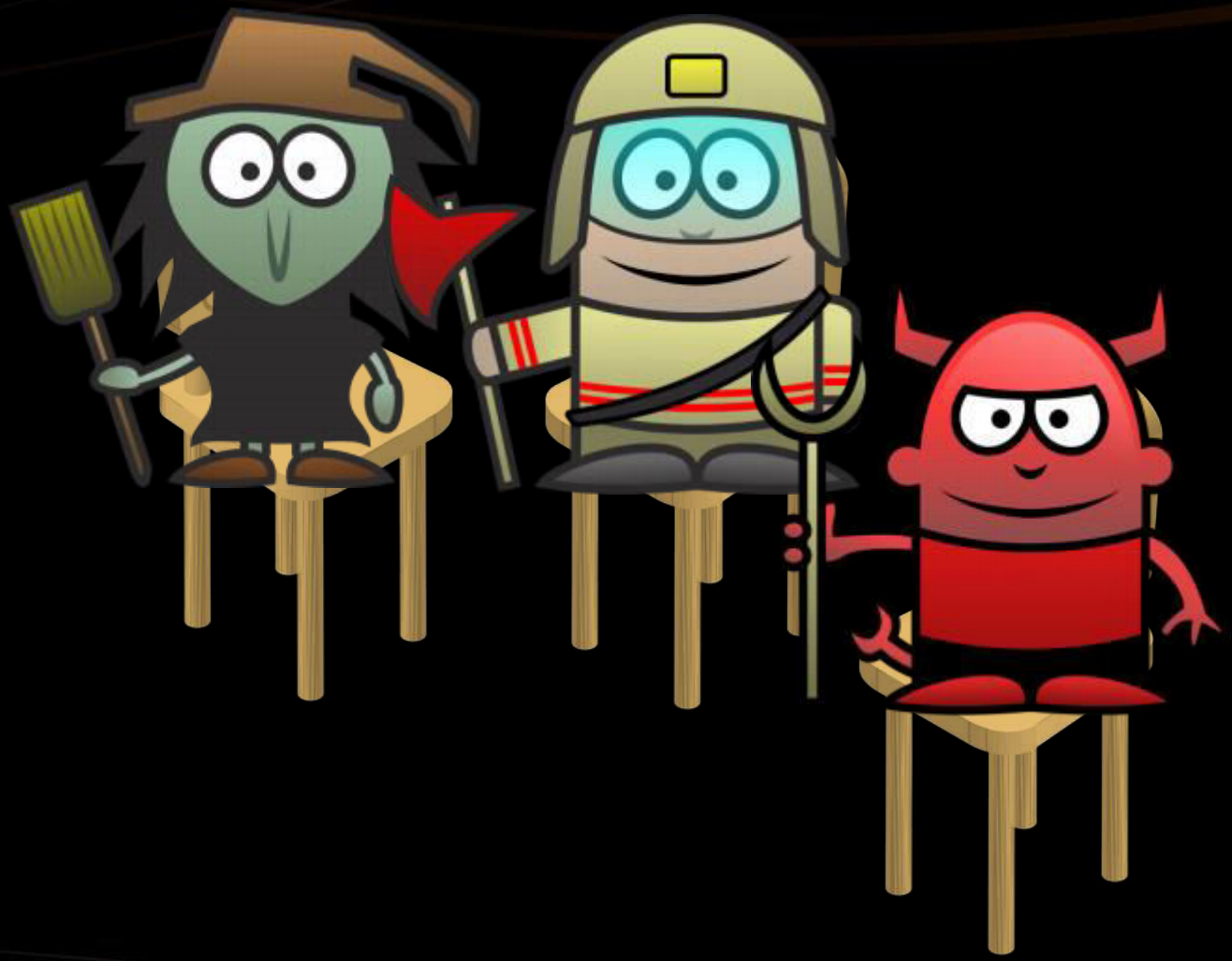
Permutations: Swap Algorithm



Permutations: Swap Algorithm

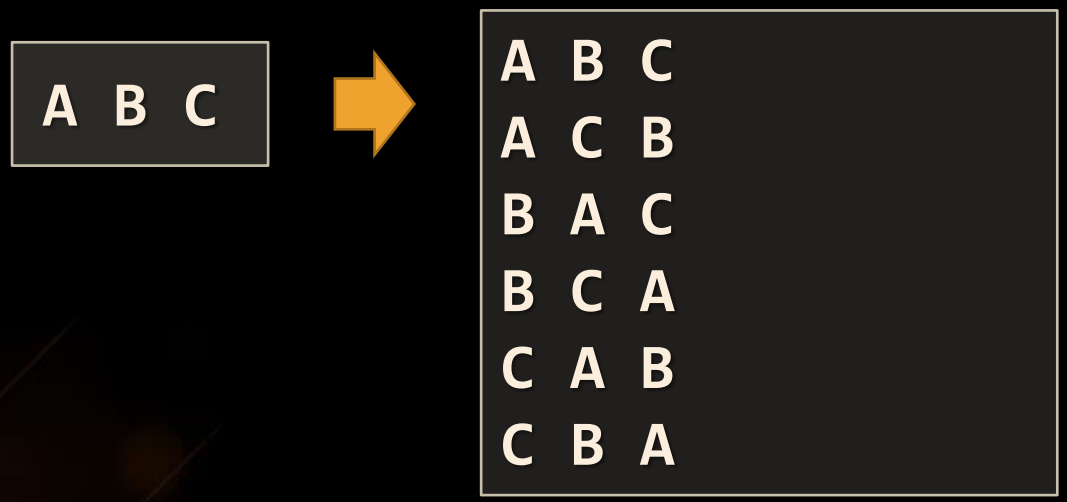


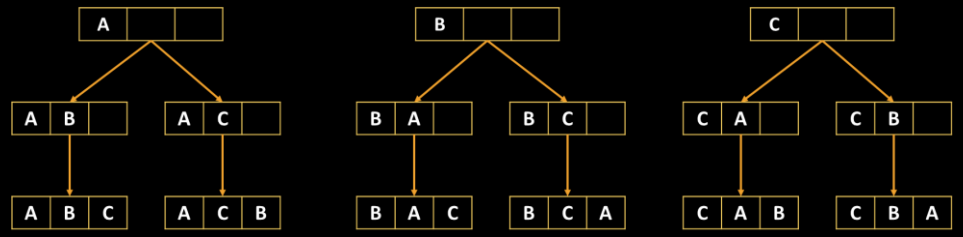
Permutations: Swap Algorithm



Problem: Optimize Permutations

- Generates all possible **permutations** of a given set of elements
  - **Without** using **extra memory**





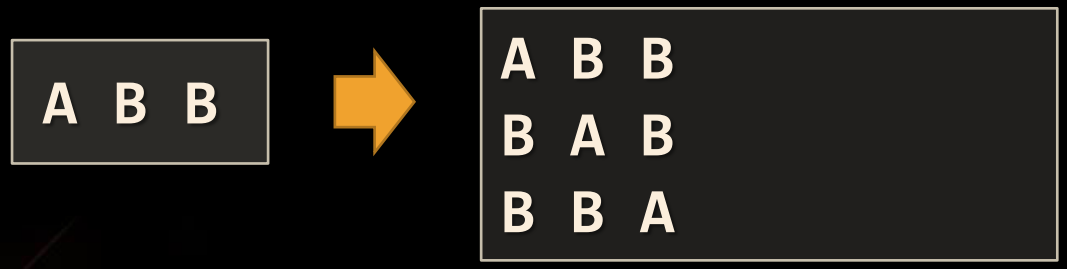
Generating Permutations

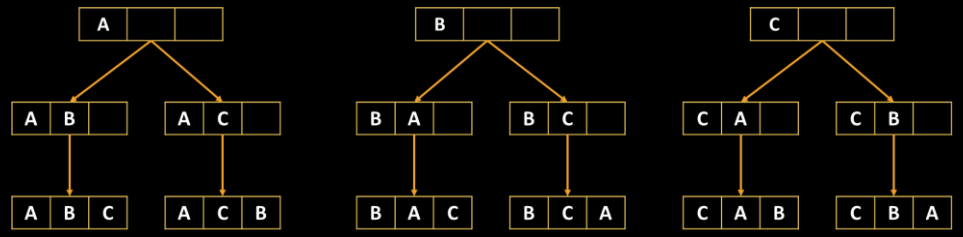
```
public static void Gen(int index)
  if (index >= set.Length)
    Console.WriteLine(string.Join(" ", set));
  else
    Gen(index + 1);
    for (int i = index + 1; i < elements.Length; i++)
    {
      Swap(index, i);
      Gen(index + 1);
      Swap(index, i);
    }
}
```

```
elements = new int[n];
Permute(0);
```

Problem: Permutations with Repetition

- What about `array = new int[] { A, B, B }`
- By definition: permutations { A, B', B'' } == { A, B'', B' }
- Generate all permutations from a **multi-set**





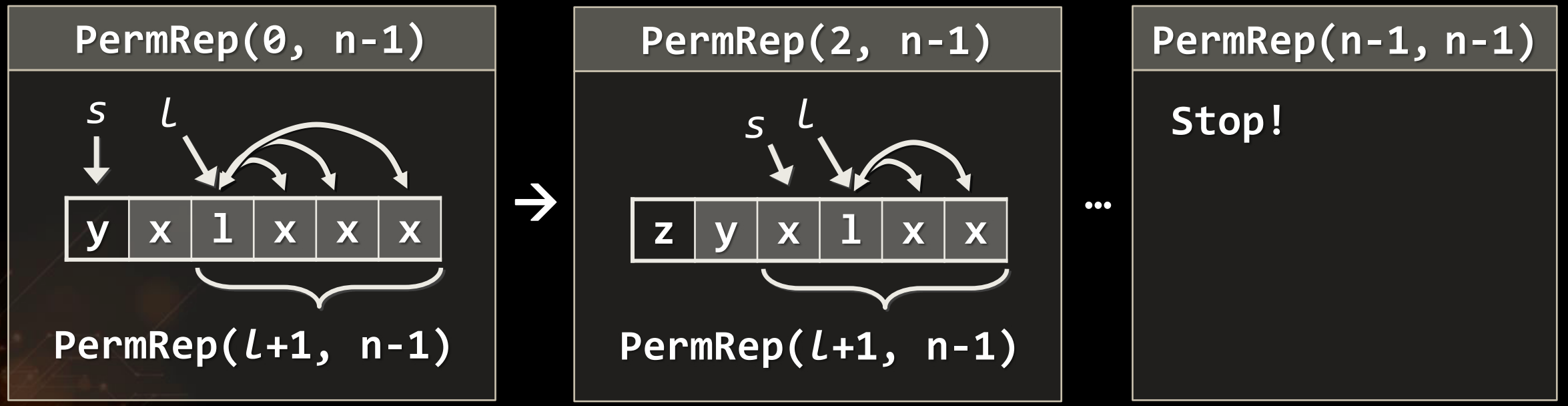
Solution: Permutations with Repetition


```
public static void Gen(int index)
  if (index >= set.Length)
    Console.WriteLine(string.Join(" ", set));
  else
    HashSet<int> swapped = new HashSet<int>();
    for (int i = index; i < elements.Length; i++) {
      if (swapped.Contains(elements[i])) {
        Swap(index, i);
        Gen(index + 1);
        Swap(index, i);
        swapped.Add(elements[i]);
      }
    }
}
```

```
elements = new int[n];
Permute(0);
```

Optimized: Permutations with Repetition

- Algorithm `PermRep(s, e)` permutes the items [`s` … `e`]
  - Exchange the item at positions `l` = `k` … `n-2` with items at `l` … `n-1`
  - Call `PermRep(l + 1)` to permute the rest of the array



More info: http://hardprogrammer.blogspot.bg/2006/11/permutaciones-con-repeticin.html

Generating Permutations with Repetition


```
static void PermuteRep(int[] arr, int start, int end)
{
  Print(arr);
  for (int left = end - 1; left >= start; left--)
  {
    for (int right = left + 1; right <= end; right++)
      if (arr[left] != arr[right])
      {
        Swap(ref arr[left], ref arr[right]);
        PermuteRep(arr, left + 1, end);
      }
    var firstElement = arr[left];
    for (int i = left; i <= end - 1; i++)
      arr[i] = arr[i + 1];
    arr[end] = firstElement;
  }
}
```

```
1, 3, 5, 5, 5
1, 5, 3, 5, 5
1, 5, 5, 3, 5
1, 5, 5, 5, 3
3, 1, 5, 5, 5
3, 5, 1, 5, 5
3, 5, 5, 1, 5
3, 5, 5, 5, 1
5, 1, 3, 5, 5
5, 1, 5, 3, 5
5, 1, 5, 5, 3
5, 3, 1, 5, 5
5, 3, 5, 1, 5
5, 3, 5, 5, 1
5, 5, 1, 3, 5
5, 5, 1, 5, 3
5, 5, 3, 1, 5
5, 5, 3, 5, 1
5, 5, 5, 1, 3
5, 5, 5, 3, 1
```

```
var arr = new int[] {3, 5, 1, 5, 5};
Array.Sort(arr);
PermuteRep(arr, 0, arr.Length-1);
```

Permutations with Repetition Count

Order **A**, **B** and **B** in all possible ways
How many ways are there?

A	B	B

$$\frac{n!}{s1!s2!..sk!} = \frac{3!}{2!1!}$$

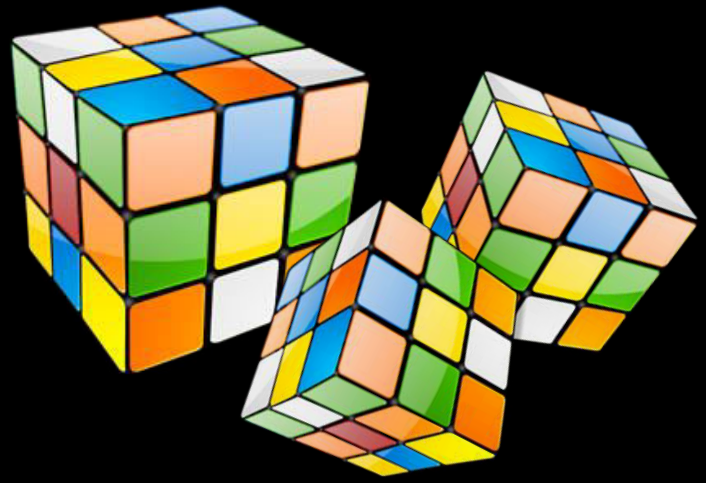
3 different ways

SOFTWARE UNIVERSITY FOUNDATION

Permutations

Live Exercises in Class (Lab)

SOFTWARE UNIVERSITY FOUNDATION



Variations

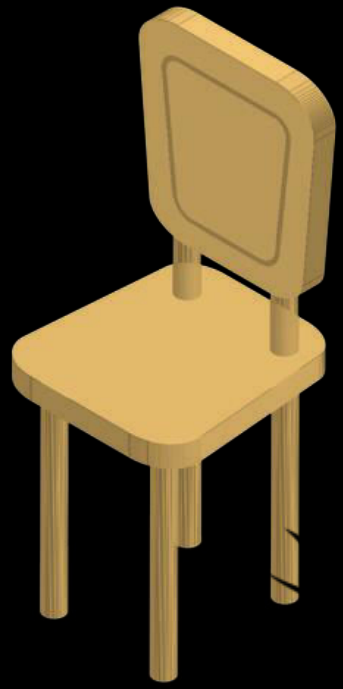
Variations

Variations

Variations

Variations

Variations



Variations

Variations

Variations

Variations

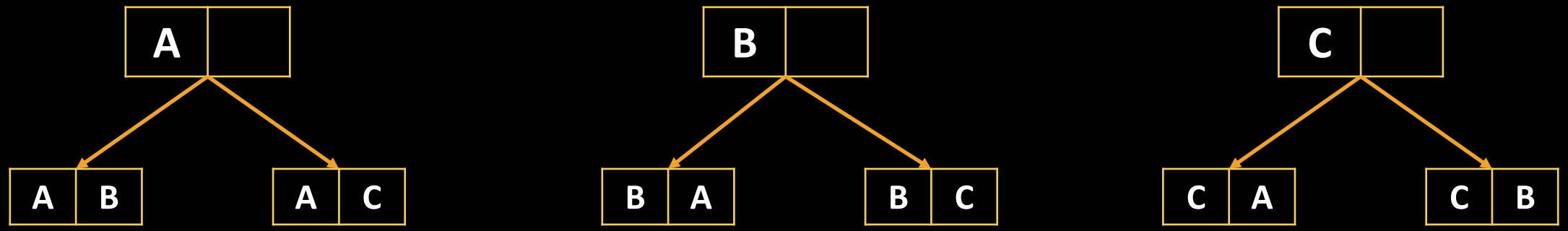
Variations

Variations

Order **A**, **B** and **C** in all possible ways int **k slots**
Pick each **item** only **once**

Variations

Order **A**, **B** and **C** in all possible ways int **k slots**
Pick each **item** only **once**

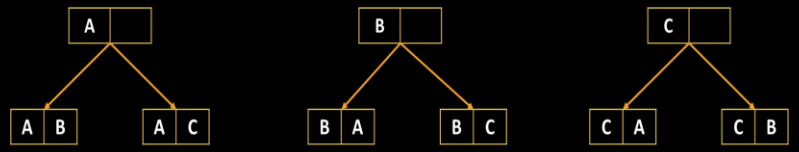


Problem: Generate Variations

- Generates all possible **variations of k** from a set of elements
- You can **pick** an **item once**

```
A B C
2
```

```
A B
A C
B A
B C
C A
C B
```



Generating Variations

```
public static void Gen(int index)
{
  if (index >= k)
    Console.WriteLine(string.Join(" ", vari));
  else
    for (int i = 0; i < elements.Length; i++)
      if (!used[i])
      {
        used[i] = true;
        vari[index] = elements[i];
        Gen(index + 1);
        used[i] = false;
      }
}
```

```
elements = new int[n];
vari = new int[k]
used = new bool[n];
Permute(0);
```

Variations Count

Order two from **A**, **B**, **C** and **D** in all possible ways
How many ways are there?

A	B	C	D

4	3

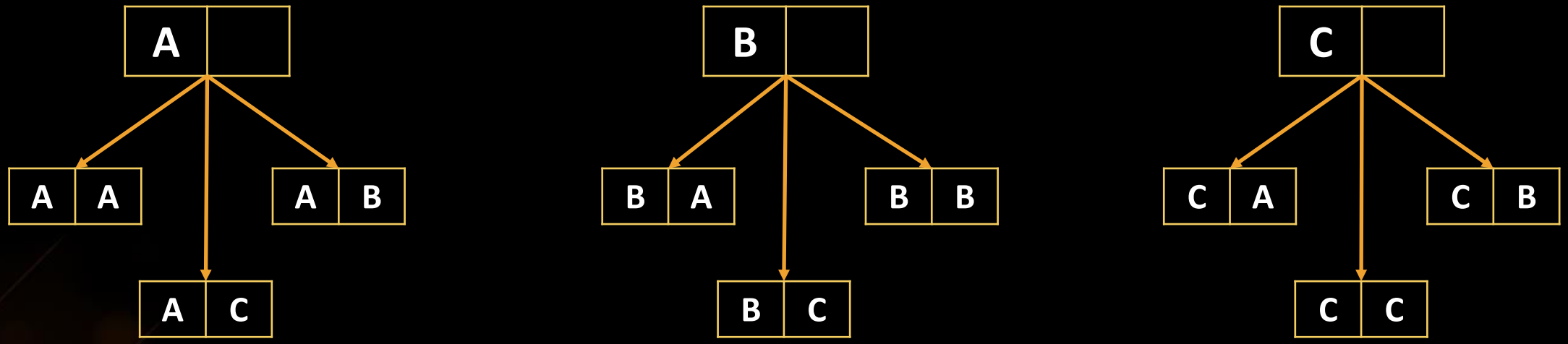
Multiply

$$\frac{n!}{(n-k)!} = \frac{4!}{2!}$$

12 different ways

Variations with Repetitions

Order **A**, **B** and **C** in all possible ways into k slots
You can **pick** an item **multiple times**

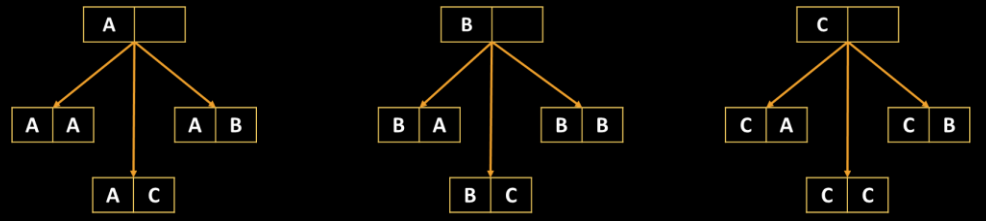


Problem: Generate Variations with Reps

- Generates all possible **variations** of a given elements
  - You can **pick** an **item multiple times**

```
A B C
2
```

```
A A
A B
A C
B A
B B
B C
C A
C B
C C
```



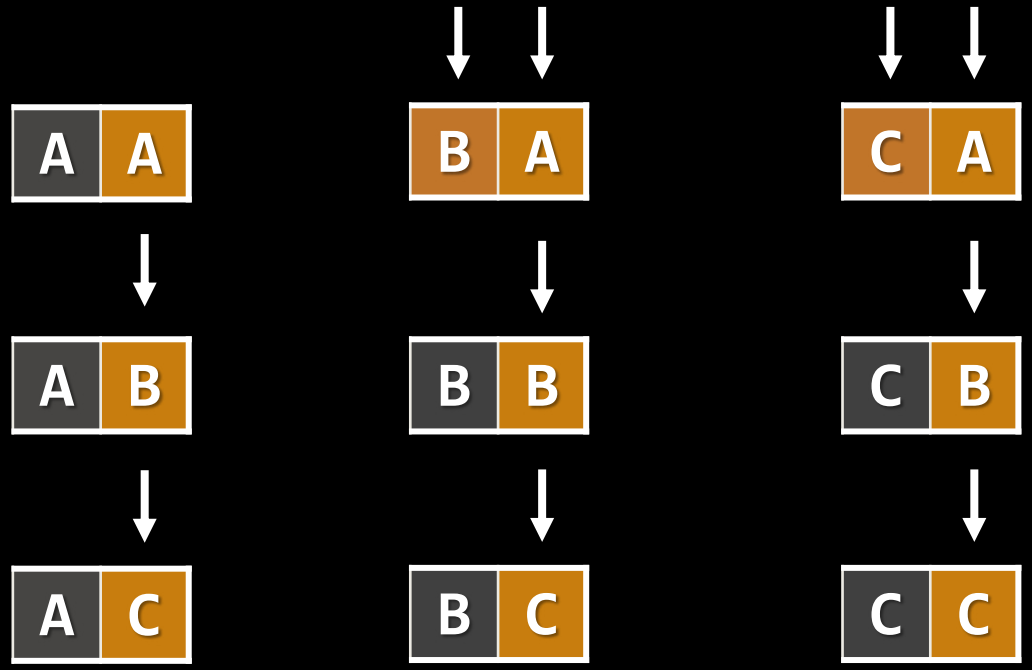
Generating Permutations

```
public static void Gen(int index)
{
  if (index >= k)
    Console.WriteLine(string.Join(" ", variation));
  else
  {
    for (int i = 0; i < n; i++)
    {
      variation[index] = elements[i];
      Gen(index + 1);
    }
}
```

```
elements = new int[n];
variation = new int[k]
Permute(0);
```

Variations with Reps: Iterative Algorithm

- Generating the variations for $n = 3$ and $k = 2$



Variations with Reps: Iterative Algorithm


```
while (true)
{
  Print(arr);
  int index = k - 1;
  while (index >= 0 && arr[index] == n-1)
    index--;
  if (index < 0)
    break;
  arr[index]++;
  for (int i = index + 1; i < k; i++)
    arr[i] = 0;
}
```

```
int n = 5;
int k = 3;
int[] arr = new int[k];
```

```
(0, 0, 0)
(0, 0, 1)
   …
(4, 4, 2)
(4, 4, 3)
(4, 4, 4)
```

Variations Count

Order two from **A**, **B**, **C** and **D** in all possible ways
How many ways are there?

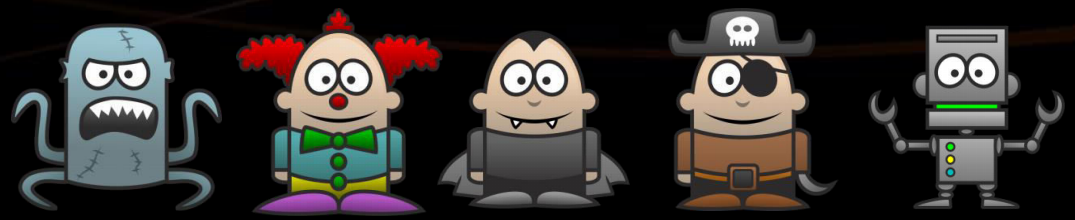
$$n^k = 4^2$$

16 different ways

SOFTWARE UNIVERSITY FOUNDATION

Combinations

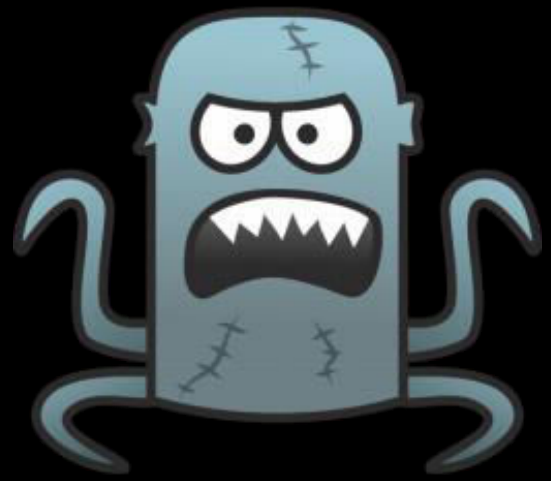
Combinations



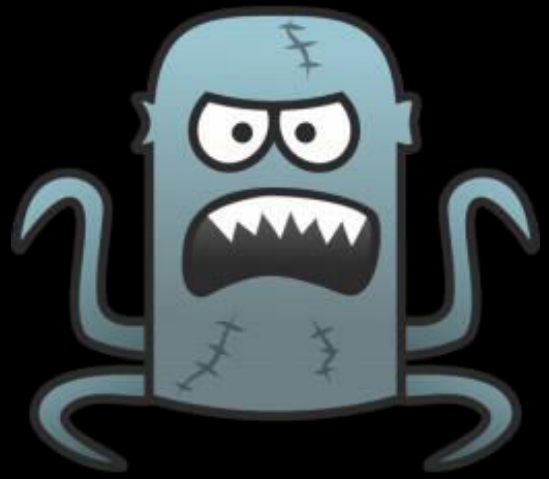
Pick **two** to hang out

Combinations

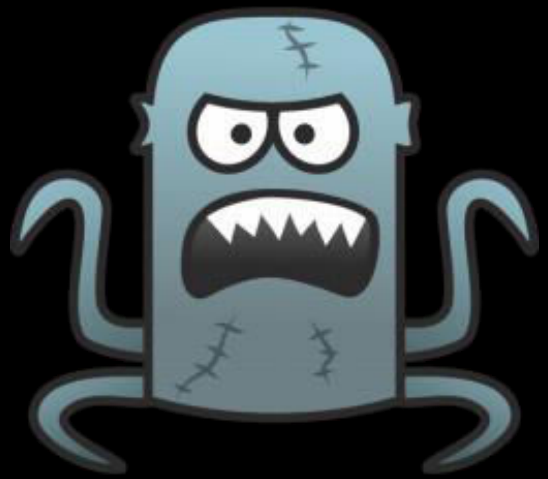
Combinations



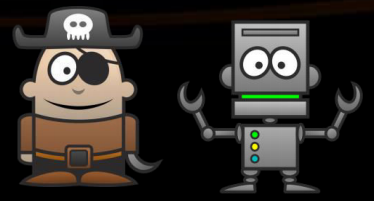
Combinations



Combinations

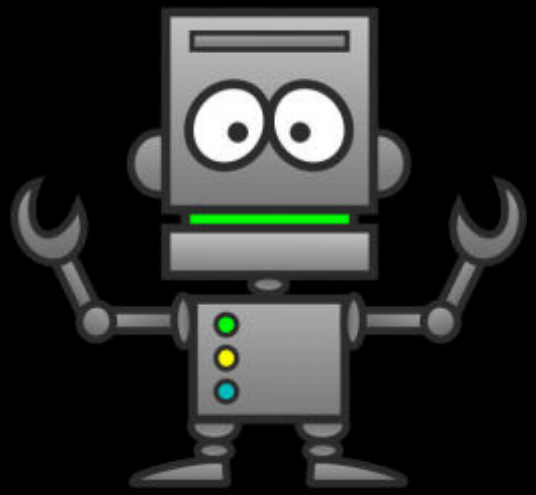


Combinations



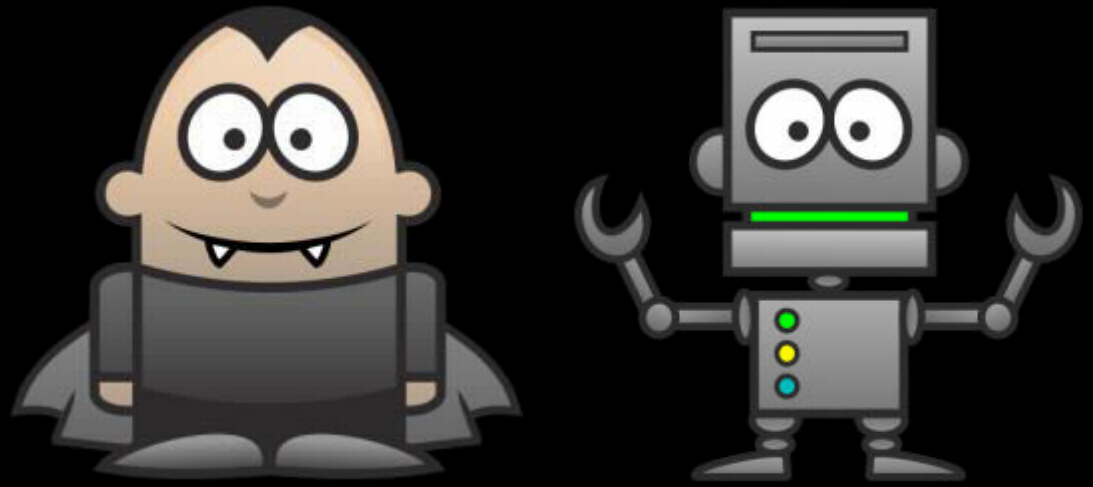
Combinations

Combinations



Combinations

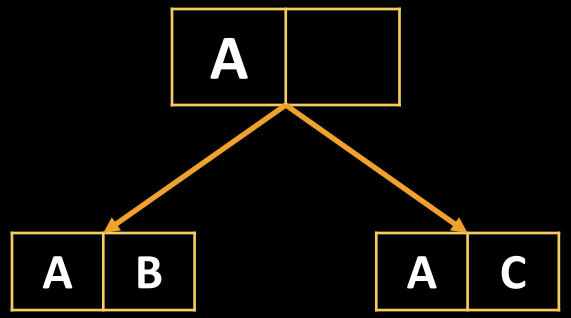
Combinations

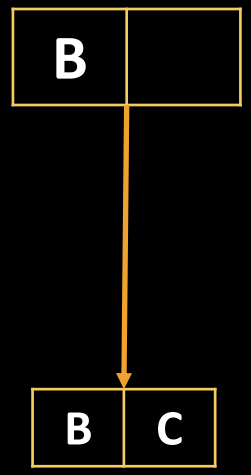


Combinations

Combinations

Pick two form **A**, **B** and **C**
Order does not matter



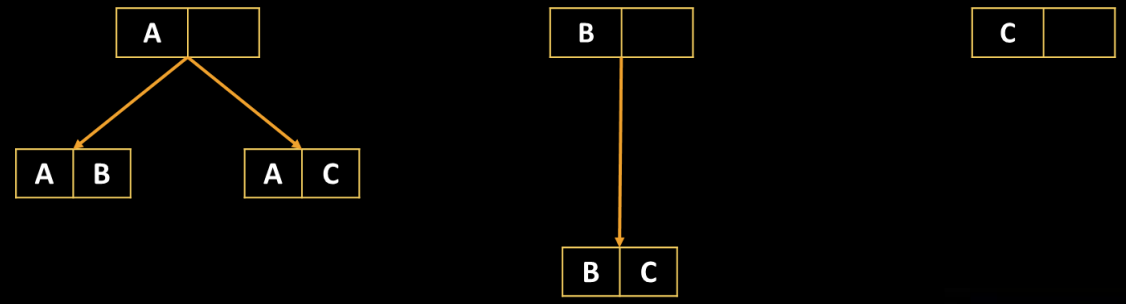


Problem: Generate Combinations

- Generates all possible combinations from a given elements
  - You can **pick** each **item** only **once**

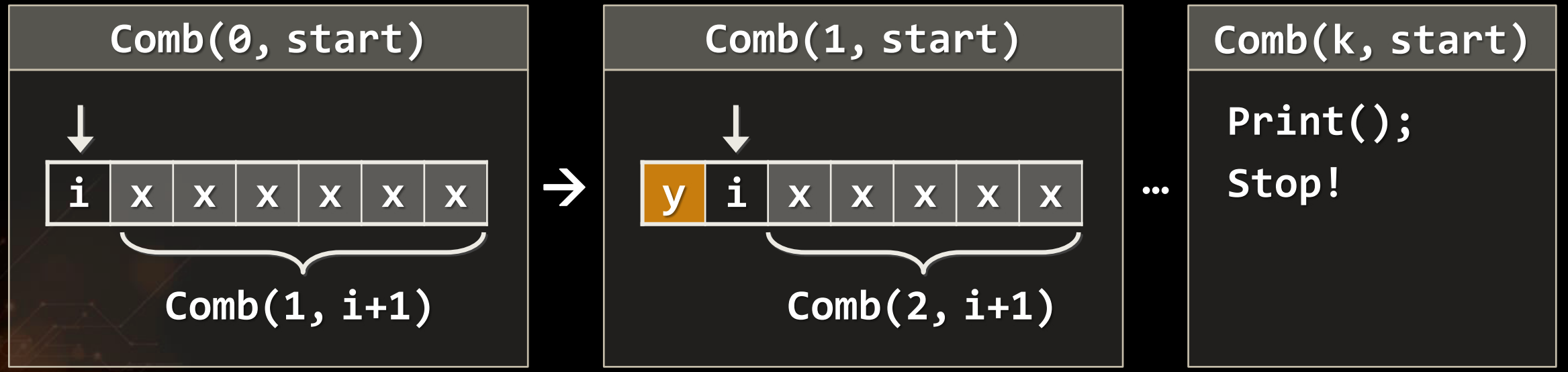
```
A B C
2
```

```
A B
A C
B C
```



Algorithm: Combinations without Repetition

- Algorithm `Comb(index, start)`
  - Put the numbers `i` = `start` … `n-1` at position `index`
  - Call `Comb(index + 1, i + 1)` to generate the rest of the array



Combinations without Repetition

```
void Comb(int index, int start)
{
  if (index >= k)
  {
    PrintCombinations();
  }
  else
    for (int i = start; i < n; i++)
    {
      arr[index] = i;
      Comb(index + 1, i + 1);
    }
}
```

```
int k = 3;
int n = 5;
int[] arr = new int[k];
Comb(0, 0);
```

Combinations Count

Pick two from **A**, **B**, **C** and **D** in all possible ways, **order does not matter**
How many ways are there?

A	B	C	D

4	3

Variations n = 4, k = 2

2	1

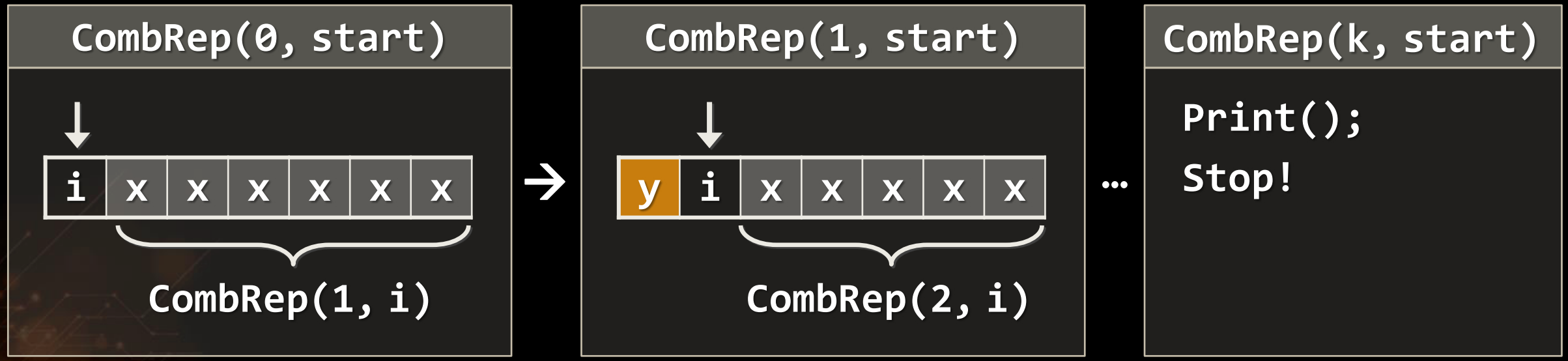
Permutations of n = 2

$$\frac{n!}{k!(n-k)!} = \frac{4!}{2!2!}$$

6 different ways

Algorithm: Combinations with Repetition

- Algorithm **Comb(index, start)**
  - Put the numbers **i** = **start** … **n-1** at position **index**
  - Call **Comb(index + 1, i)** to generate the rest of the array



Generate Combinations with Repetition

```
void Comb(int index, int start)
{
  if (index >= k)
  {
    PrintCombinations();
  }
  else
    for (int i = start; i < n; i++)
    {
      arr[index] = i;
      Comb(index + 1, i);
    }
}
```

```
int k = 3;
int n = 4;
int[] arr = new int[k];
Comb(0, 0);
```

SOFTWARE UNIVERSITY FOUNDATION

Variations and Combinations

Live Exercises in Class (Lab)

SOFTWARE UNIVERSITY FOUNDATION

$$C_n^k = \binom{n}{k} = \frac{n!}{(n-k)!\,k!}$$

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

N Choose K Count

Problem: Combinations Count

- How many **combinations** are there for **n = 16, k = 15**
- **Solution:**
  - How many ways to pick 15 items?

$$C_n^k = \binom{n}{k} = \frac{n!}{(n-k)!\,k!}$$

16 * 15 * 14 * … * 2

  - Divide by the number of ways in which you can arrange 15 numbers

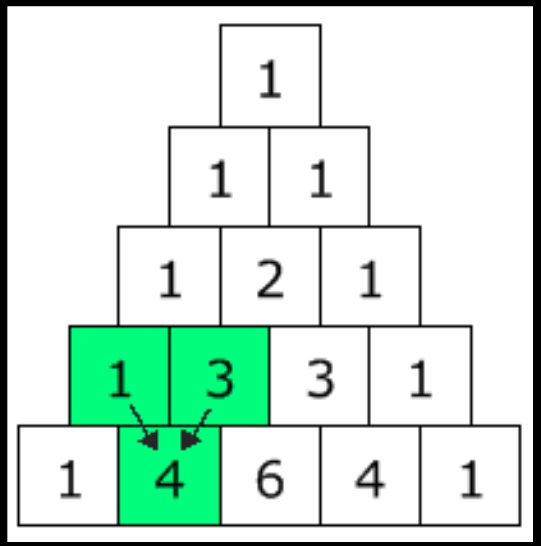
15 * 14 * 13 * … * 1

- Possible combinations → **16**

Pascal's Triangle

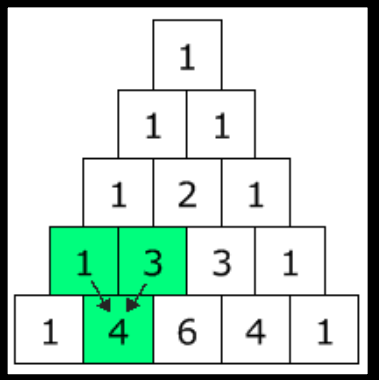
- Each node → how many ways are there to reach it?
- Quickly find **N choose K** count
  - Go down to row *n* (the top row is 0)
  - Move along *k* places to the right
  - The value there is your answer

$$C_n^k = \binom{n}{k} = \frac{n!}{(n-k)!\,k!}$$



Binomial Coefficients: Calculation

$$C_n^k = \binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$




```
Base cases:
if k > n → 0
if k == 0 → 1
if k == n → 1
```

Binomial Coefficients: Calculation

```
decimal Binom(int n, int k)
{
  if (k > n)
    return 0;
  if (k == 0 || k == n)
    return 1;
  return Binom(n - 1, k - 1) + Binom(n - 1, k);
}
```

$$C_n^k = \binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

This is slow
(stay tuned for DP)

Summary

- **Permutations** – Ways to order n elements
- **Variations** – Ways to **order k** of **n** elements
- **Combinations** – Ways to **choose k** of **n** elements
- Pascal's Triangle
  - Binomial Coefficients () – **N choose K Count**

Combinatorial Algorithms

Questions?

https://softuni.bg/opencourses/algorithms

License

- This course (slides, examples, labs, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

- Attribution: this work may contain portions from
  - "Fundamentals of Computer Programming with C#" book by Svetlin Nakov & Co. under CC-BY-SA license
  - "Data Structures and Algorithms" course by Telerik Academy under CC-BY-NC-SA license

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University Foundation
  - softuni.org
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

SOFTWARE UNIVERSITY

SOFTWARE UNIVERSITY FOUNDATION

FORUM