# **ЧИСЛА КАТАЛАНА**

Числами ***Каталана*** *c n* ( *n* = 0, 1, 2, …) називаються числа виду

*c n* =

Першими числами послідовності будуть 1, 1, 2, 5, 14, 42, 132, 429, 1430, ….

Числа Каталана є рішенням рекурентного рівняння

*з* 0 = 1,

*з n* = *c* 0 *c n* -1 + *c* 1 *c n* -2 + *c* 2 *c n* -3 + ... + *c n* -1 *c* 0 = при *n* > 0

Наприклад, *з* 1 = *c* 0 *c* 0 = 1, *з* 2 = *c* 0 *c* 1 + *c* 1 *c* 0 = 1 + 1 = 2 тощо.

Заповнити масив cat[10] числами Каталана можна за допомогою вкладеного циклу:

cat[0] = cat[1] = 1;

for(i=2;i<10;i++)

{

for(j=0;j<i;j++)

cat[i] += cat[j]\*cat[ij-1];

}

**Приклад.** Розміщення дужок.Позначимо через *f* ( *n* ) кількість повних розстановок дужок у добутку *n* чисел. Останній добуток матиме місце між *k* - им і ( *k* + 1) - им числом. І тому необхідно окремо обчислити добуток перших *k* чисел і ( *n* – *k* ) останніх. Тому отримаємо рекурентне співвідношення:

*f* ( *n* ) =

Це співвідношення задає послідовність чисел Каталана: *f* ( *n* ) = cn *-* 1 . Зауважимо, що добуток *n* чисел містить *n* – 1 операцію множення.

**ЧИСЛА СТИРЛІНГА**

***Число Стірлінга другого роду*** S( *n* , *k* ) дорівнює кількості способів розбиття множини з *n* елементів на *k* непустих підмножин і позначається (читається " *k* підмножин з *n* "). Наприклад, існує 7 способів розбиття чотириелементної множини на дві частини:

{1, 2, 3} {4}, {1, 2, 4} {3}, {1, 3, 4} {2}, {2, 3, 4} {1},

{1, 2} {3, 4}, {1, 3} {2, 4}, {1, 4} {2, 3}

Тому = 7.

Розглянемо значення чисел Стірлінга для малих значень *k* :

1. *k* = 0. Існує лише один спосіб розбиття порожньої множини на нульове число непустих частин, тому = 1. Для непорожньої множини потрібна принаймні хоча б одна частина, тому = 0 при *n* > 0.

2. *k* = 1. Існує тільки один спосіб поміщення *n* елементів в одну-єдину непорожню множину, тому = 1 при *n* > 0. Однак = 0, так як 0-елементна множина порожня.

3. *k* = 2. Очевидно, що = 0. Якщо множина з *n* > 0 об'єктів розділено на дві непорожні частини, одна з цих частин містить останній об'єкт і деяке підмножина з *n* – 1 об'єктів. Є 2 *n* -1 підмножин із *n* – 1 об'єктів. Оскільки всі об'єкти не можна помістити в одну частину, то .

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *n* |  |  |  |  |  |  |  |  |  |  |
| 0 | 1 |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 |  |  |  |  |  |  |  |  |
| 2 | 0 | 1 | 1 |  |  |  |  |  |  |  |
| 3 | 0 | 1 | 3 | 1 |  |  |  |  |  |  |
| 4 | 0 | 1 | 7 | 6 | 1 |  |  |  |  |  |
| 5 | 0 | 1 | 15 | 25 | 10 | 1 |  |  |  |  |
| 6 | 0 | 1 | 31 | 90 | 65 | 15 | 1 |  |  |  |
| 7 | 0 | 1 | 63 | 301 | 350 | 140 | 21 | 1 |  |  |
| 8 | 0 | 1 | 127 | 966 | 1701 | 1050 | 266 | 28 | 1 |  |
| 9 | 0 | 1 | 255 | 3025 | 7770 | 6951 | 2646 | 462 | 36 | 1 |

Трикутник Стірлінга для числа підмножин

Виведемо рекурентне співвідношення, з якого можна буде обчислити значення . Якщо задано множина з *n* > 0 об'єктів, яке має бути розбита на *k* непустих частин, то ми або поміщаємо останній об'єкт в окремий клас, а *n* - 1 об'єкт, що залишилися, розбиваємо на *k* - 1 частина способами, або поміщаємо його в будь-яку з *k* частин , куди розбиті *n* – 1 об'єкт. На разі є *k* \* можливих варіантів, оскільки кожен із способів розподілу перших *n* – 1 об'єктів по *k* непустим частинам дає *k* підмножин, із якими можна об'єднати *n* -ий об'єкт. Маємо рекурентну формулу:

= + *k* \* , *n* > 0

**Приклад 1.** Розглянемо програму, яка обчислює числа Стірлінга другого роду, причому значення S( *n* , *k* ) буде обчислюватися в комірці s[ *n* ] [ *k* ]. Виведемо отримані значення як форматованої таблиці на екран.

#include <stdio.h>

#include <memory.h>

int n, k, s[10][10];

void main( void )

{

memset(s,0, sizeof (s));

s[0][0] = 1;

for (n = 1; n < 10; n++)

for (k = 1; k <= n; k++)

s[n][k] = s[n-1][k-1] + k \* s[n-1][k];

for (n = 0; n < 10; n++)

{

for (k = 0; k <= n; k++)

printf( "%4d" , s[n][k]);

printf( "\n" );

}

}

***Число Стірлінга першого роду*** s( *n* , *k* ) дорівнює кількості способів представлення *n* об'єктів у вигляді *k* циклів і позначається (читається " *k* циклів з *n* ").

Наприклад, цикли з 4 елементів [ *a* , *b* , *c* , *d* ], [ *b* , *c* , *d* , *a* ], [ *c* , *d* , *a* , *b* ] і [ *d* , *a* , *b* , *c* ] є однаковими. Цикл можна прокручувати, оскільки його кінець з'єднаний із початком.

Наприклад, існує 11 різних способів скласти два цикли з чотирьох елементів:

[1, 2, 3][4], [1, 2, 4][3],[1, 3, 4][2],[2, 3, 4][1],

[1, 3, 2][4], [1, 4, 2][3],[1, 4, 3][2],[2, 4, 3][1],

[1, 2] [3, 4], [1, 3] [2, 4], [1, 4] [2, 3]

Тому = 11.

Одиничний цикл (цикл, що складається з одного елемента) - це те саме, що і одинична множина. 2-цикл подібний до 2-множини, оскільки [A, B] = [B, A] як і {A, B} = {B, A}. Однак вже існує два різні 3-цикли: [A, B, C] та [A, C, B]. З будь-якої *n* -елементної множини можуть бути складені *n* ! / *n* = ( *n* - 1)! циклів, якщо *n* > 0 (всього є *n* ! перестановок, а кожен цикл відповідає відразу *n* їх, оскільки відлік циклу може бути розпочато з будь-якого з його елементів). Тому

= ( *n* - 1)!.

Якщо всі цикли є одиничними чи подвійними, то = . Наприклад,

= = 1, = =

Виведемо рекурентну формулу для обчислення чисел Стірлінга першого роду. Кожне представдення *n* об'єктів як *k* циклів або поміщає останній об'єкт в окремий цикл s( *n* – 1, *k* – 1) способами, або вставляє цей об'єкт у одне з s( *n* – 1, *k* ) циклічних уявлень перших *n* – 1 об'єктів. На разі існує *n* – 1 різних методів подібної вставки. Наприклад, при вставці елемента *d* в цикл [ *a* , *b* , *c* ] можна отримати тільки 3 різних цикли: [ *a* , *b* , *c* , *d* ], [a, *b* , *d* , *c* ], [ *a* , *d* , *b* , *c* ]. Таким чином, рекурентність має вигляд:

= + ( *n* - 1) \* , *n* > 0

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *N* |  |  |  |  |  |  |  |  |  |  |
| 0 | 1 |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 |  |  |  |  |  |  |  |  |
| 2 | 0 | 1 | 1 |  |  |  |  |  |  |  |
| 3 | 0 | 2 | 3 | 1 |  |  |  |  |  |  |
| 4 | 0 | 6 | 11 | 6 | 1 |  |  |  |  |  |
| 5 | 0 | 24 | 50 | 35 | 10 | 1 |  |  |  |  |
| 6 | 0 | 120 | 274 | 225 | 85 | 15 | 1 |  |  |  |
| 7 | 0 | 720 | 1764 | 1624 | 735 | 175 | 21 | 1 |  |  |
| 8 | 0 | 5040 | 13068 | 13132 | 6769 | 1960 | 322 | 28 | 1 |  |
| 9 | 0 | 40320 | 109584 | 118124 | 67284 | 22449 | 4536 | 546 | 36 | 1 |

Трикутник Стірлінга для числа циклів

**Теорема.** Має місце співвідношення:



**Доведення.** позначає кількість перестановок *n* об'єктів, що містить до *k* циклів. Якщо підсумувати по всіх *k* , то має вийти загальна кількість перестановок, що дорівнює *n* !.

**Приклад 2.** Розглянемо програму, яка обчислює числа Стирлінг першого роду. Значення s( *n* , *k* ) обчислюється в комірці s[ *n* ] [ *k* ]. Виводимо отримані значення як форматованої таблиці на екран.

#include <stdio.h>

#include <memory.h>

int n, k, s[10][10];

void main( void )

{

memset(s,0, sizeof (s));

s[0][0] = 1;

for (n = 1; n < 10; n++)

for (k = 1; k <= n; k++)

s[n][k] = s[n-1][k-1] + (n-1) \* s[n-1][k];

for (n = 0; n < 10; n++)

{

for (k = 0; k <= n; k++)

printf( "%6d" , s[n][k]);

printf( "\n" );

}

}

**ЧИСЛА БЕЛЛА**

***Число Белла*** B *n* дорівнює кількості розбиття множини з *n* елементів на довільну кількість непустих підмножин. Очевидно, що B 0 = 1, оскільки існує тільки одне розбиття порожньої множини. Наприклад, B 3 = 5, оскільки існує 5 можливих розбиття множини { *a* , *b* , *c* } з трьох елементів:

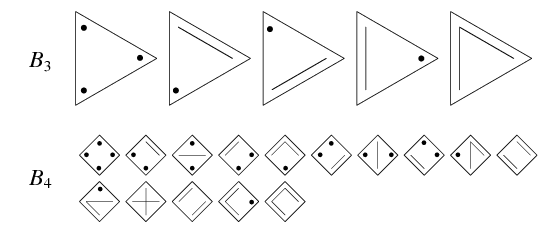
{{a}, {b}, {c}}, {{a, b}, {c}}, {{ a, c }, {b}}, {{a}, {b, c}}, {{a, b, c}}

Зауважимо, що *n* елементів можна розбити на *i* множин (1  *i*  *n* ). При цьому кількість розбиття *n* - елементної множини на *k* підмножин дорівнює числу Стірлінга 2 роду S( *n* , *k* ). Звідки одержуємо формулу:

B *n* =

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *n* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| B *n* | 1 | 1 | 2 | 5 | 15 | 52 | 203 | 877 | 4140 | 21147 | 115975 |

Розглянемо такі конструкції, у яких точки позначають одноелементні множини, а сегменти поєднують елементи, що належать одному множині. З елементів *n* можна побудувати B *n*різних таких конструкцій.



**Теорема.** Числа Белла задовольняють наступному рекурентному співвідношенню:



**Доведення.** Розглянемо розбиття *n* +1 елемента в залежності від величини блоку, в якому знаходиться ( *n* +1) - ий елемент. Нехай розмір цього блоку дорівнює *j* (1  *j*  *n* +1). Тоді існує спосіб вибрати в нього крім ( *n* + 1) - ого ще *j* - 1 елемент. Інші *n* + 1 - *j* елементів можна розбити B *n* + 1 - *j* способами. Таким чином:



Наприклад, B 4 = B 0 + B 1 + B 2 + B 3 = 1 \* 1 + 3 \* 1 + 3 \* 2 + 1 \* 5 = 15.

Числа Белла задовольняють таку властивість:



Для значень *n* = 0, 1, 2, … отримаємо такі значення детермінанту:

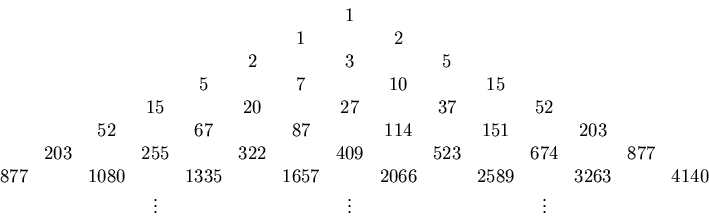
1, 1, 2, 12, 288, 34560, 24883200, 125411328000, 5056584744960000,

1834933472251084800000, 6658606584104736522240000000, …

При розкладанні функції ряд Маклорена коефіцієнти ряду утворюють числа Белла:



Числа Bn *можуть* бути побудовані за допомогою ***трикутника Белла*** . Перший рядок містить 1. Кожен наступний рядок починається числом, що стоїть наприкінці попереднього рядка. Кожне наступне число в рядку дорівнює сумі чиел, що стоять ліворуч та зверху від нього. Числа Белла утворюють останні числа у рядках.



**Приклад 3.** Наступна програма обчислює числа Белла та виводить їх на екран.

#include <stdio.h>

#include <memory.h>

#define MAX 11

int n,k,i;

int bell[MAX], s[MAX][MAX];

void Strirling ( void )

{

memset(s,0, sizeof (s));

s[0][0] = 1;

for (n = 1; n < MAX; n++)

for (k = 0; k <= n; k++)

s[n][k] = s[n-1][k-1] + k \* s[n-1][k];

}

void Bell( void )

{

Strirling();

memset(bell,0, sizeof (bell));

bell[0] = 1;

for (n = 1; n < MAX; n++)

for (k = 1; k <= n; k++)

bell [n] + = s [n] [k];

}

void main( void )

{

Bell();

for (i = 0; i < MAX; i++)

printf( "%d" , bell[i]);

printf( "\n" );

}

**Приклад 4.** Скільки способів можна розкласти *n* помітних об'єктів в одну або кілька коробок?

При розкладанні *n* об'єктів одночасно задіяно може бути максимум *n* коробок, а кожному допустимому розкладанню відповідає деяке розбиття множини з *n* об'єктів. Кількість таких розбиття дорівнює числу Белла B *n* .

**ЧИСЛА ЕЙЛЕРА**

***Числом Ейлера першого роду*** E( *n* , *k* ) називається кількість перестановок порядку *n* з *k* підйомами. Тобто таких перестановок X = ( *x* 1 , *x* 2 , …, *x n* ) , що існує рівно *k* індексів *j* , для яких *x j* < *x j* +1 .

Для заданого *n* існує єдина перестановка без підйомів ( *n* , *n* – 1, …, 2, 1). Також існує єдина перестановка, що має *n* - 1 підйом ( 1, 2, ..., *n* - 1, *n* ) . Значить

E( *n* , 0) = E( *n* , *n* – 1) = 1

Дзеркальним відображенням перестановки з *k* підйомами є перестановка із *n* – *k* – 1 підйомом. Таким чином

E( *n* , *k* ) = E( *n* , *n* – *k* – 1)