Простейшие переборные задачи: генерация подмножеств и перестановок

Лабораторная работа №1

Раздел: комбинаторика

Цель занятия:

Изучение простейших переборных задач: генерация подмножеств и перестановок.

Генерация множества перестановок

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2, \ldots, n$ в лексикографическом порядке, например, для $n = 4$:

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

Генерация множества перестановок

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2,\ldots,n$ в лексикографическом порядке, например, для $n = 4$:

Генерация множества перестановок

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2,\ldots,n$ в лексикографическом порядке, например, для $n = 4$:

Генерация множества перестановок

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2,\ldots,n$ в лексикографическом порядке, например, для $n = 4$:

Генерация множества перестановок

- Реализуем этот способ
- Необходима функция, принимающая на вход вектор и множество
  - Множество реализуем бинарным вектором
  - Также будем передавать число элементов, которые еще необходимо добавить к вектору: если оно равно нулю, значит, перестановка построена

Генерация множества перестановок

```
void GenPermut(size_t elems, vector<size_t>& cur, vector< bool>& used) {
    if (elems == cur.size()) {
        for (size_t i = 0; i < cur.size() - 1; ++i) {
            cout << cur[i] + 1 <<  " ";
        }
        cout << cur[cur.size() - 1] + 1 <<  "\n";
    }
    for (size_t next = 0; next < elems; ++next) {
        if (!used[next]) {
            cur.push_back(next);
            used[next] =  true;

            GenPermut(elems, cur, used);

            cur.pop_back();
            used[next] =  false;
        }
    }
}
```

Построение перестановки по ее номеру

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2, \ldots, n$ в лексикографическом порядке, например, для $n = 4$:

$m = 1$	1 2 3 4
$m = 24$	4 3 2 1
$m = 7$	2 1 3 4
$m = 13$	2 4 1 3

Построение перестановки по ее номеру

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2,\ldots,n$ в лексикографическом порядке, например, для $n = 4$:

Построение перестановки по ее номеру

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2,\ldots,n$ в лексикографическом порядке, например, для $n = 4$:

Построение перестановки по ее номеру

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2, \ldots, n$ в лексикографическом порядке, например, для $n = 4$:

Построение перестановки по ее номеру

```
vector<size_t> Permutation(size_t elemCount, size_t permNumber) {
    vector<size_t> numbers;
    for (size_t i = 0; i < elemCount; ++i) {
        numbers.push_back(i);
    }

    int64 currentElementsCount = elemCount;

    vector<size_t> ans;
    while (currentElementsCount > 0) {
        int64 k = 0;
        int64 L = fact(currentElementsCount - 1);

        while ((k + 1) * L < permNumber) {
            ++k;
        }

        size_t curNumber = -1;

        for (size_t j = 0; j < elemCount; ++j) {
            if (numbers[j] != -1) {
                ++curNumber;
            }
            if (curNumber == k) {
                ans.push_back(numbers[j] + 1);
                numbers[j] = -1;
                break;
            }
        }

        permNumber -= L*k;
        --currentElementsCount;
    }
    return ans;
}
```

Задания

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2, \ldots, n$ в лексикографическом порядке, например, для $n = 4$:

Пример входных данных	Пример выходных данных
2	1 2 2 1
3	1 2 3 1 3 2 2 1 3 2 3 1 3 1 2 3 2 1

Задания

- Дано число $n \geq 1$. Требуется сгенерировать все перестановки чисел $1,2, \ldots, n$ в лексикографическом порядке, например, для $n = 4$:

Пример входных данных	Пример выходных данных
10 3628800	10 9 8 7 6 5 4 3 2 1
10 1	1 2 3 4 5 6 7 8 9 10