* + - 1. Что такое комбинаторика?

Комбинаторика - это раздел математики, который изучает комбинаторные структуры и проблемы, связанные с их анализом и перечислением. Она занимается подсчетом, классификацией, анализом и определением закономерностей множеств объектов, учитывая их порядок или отсутствие порядка, повторяемость или неповторяемость элементов. В комбинаторике рассматриваются такие вопросы, как перестановки, сочетания, размещения, разбиения, графы, деревья и другие комбинаторные объекты. Комбинаторика находит применение в различных областях, включая математическую статистику, теорию информации, теорию кодирования, компьютерную науку, теорию игр, экономику, физику и другие науки.

* + - 1. Что такое генератор?

Генератор - это функция в программировании, которая возвращает итератор, при каждом вызове которого генерируется новое значение. Он позволяет генерировать значения на лету вместо того, чтобы генерировать все значения сразу и сохранять их в памяти. Генераторы могут быть полезны в различных сценариях, таких как обработка больших объемов данных, где необходимо сократить использование памяти.

* + - 1. Чем размещения отличаются от сочетаний?

1. Размещения и сочетания - это два понятия комбинаторики, связанные с выбором и перестановкой элементов из некоторого множества.
2. Размещения - это комбинации элементов, где учитывается порядок элементов. То есть, размещения - это все возможные упорядоченные наборы элементов. Например, если у нас есть множество {A, B, C}, то размещениями из двух элементов будут {AB, AC, BA, BC, CA, CB}. Отметим, что порядок здесь имеет значение, то есть AB и BA считаются разными размещениями.
3. Сочетания - это комбинации элементов, где порядок не учитывается. То есть, сочетания - это все возможные наборы элементов без учета их порядка. Например, если у нас есть множество {A, B, C}, то сочетаниями из двух элементов будут {AB, AC, BC}. Отметим, что порядок здесь не имеет значения, то есть AB и BA считаются одним и тем же сочетанием.
4. Таким образом, основное отличие между размещениями и сочетаниями заключается в том, что при размещениях учитывается порядок элементов, а при сочетаниях порядок не имеет значения.
   * + 1. Каких подмножеств множества любой размерности всегда по одному?
5. Если множество содержит n элементов, то существует 2^n подмножеств, включая пустое множество и само множество.
6. Если рассмотреть подмножества множества любой размерности, которые содержат k элементов, то количество таких подмножеств можно выразить следующей формулой сочетаний: C(n,k) = n! / (k! \* (n-k)!)
7. Однако, если рассмотреть подмножества множества любой размерности, которые содержат по одному элементу, то таких подмножеств будет n, где n - количество элементов в множестве. Например, если множество состоит из элементов {A, B, C, D}, то подмножествами с одним элементом будут {A}, {B}, {C}, {D}.
8. Таким образом, подмножества множества любой размерности, которые содержат по одному элементу, всегда будут по одному на каждый элемент множества.
   * + 1. С помощью какого генератора решается задача о загрузке судна?

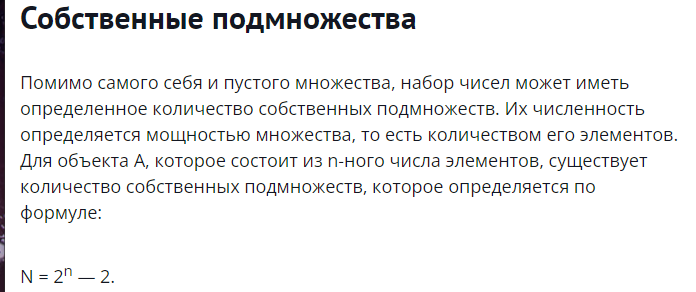
С помощью генератора размещений и генератора сочетаний

* + - 1. С помощью какого генератора решается задача о рюкзаке?

С помощью генератора множества всех подмножеств

* + - 1. С помощью какого генератора решается задача о коммивояжере?

С помощью генератора перестановок



Генерация подмножеств заданного множества

|  |
| --- |
| setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " - Генератор множества всех подмножеств -";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация всех подмножеств ";  combi::subset s1(sizeof(AA) / 2); // создание генератора  int n = s1.getfirst(); // первое (пустое) подмножество  while (n >= 0) // пока есть подмножества  {  std::cout << std::endl << "{ ";  for (int i = 0; i < n; i++)  std::cout << AA[s1.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = s1.getnext(); // cледующее подмножество  };  std::cout << std::endl << "всего: " << s1.count() << std::endl;  system("pause");  return 0; |

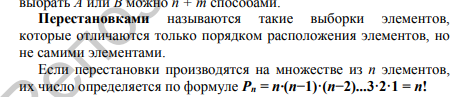
Листинг 1



Генерация сочетаний

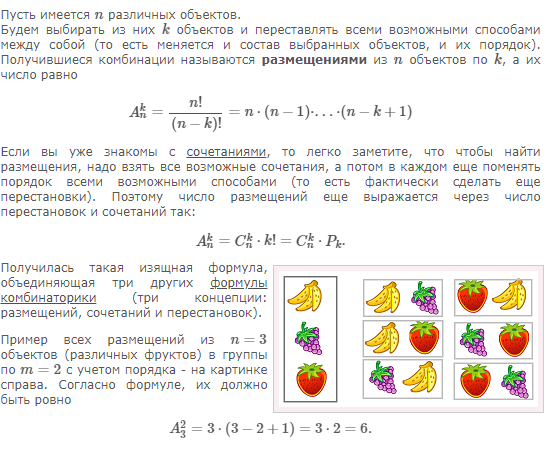
|  |
| --- |
| setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D", "E" };  std::cout << std::endl << " --- Генератор сочетаний ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация сочетаний ";  combi::xcombination xc(sizeof(AA) / 2, 3);  std::cout << "из " << xc.n << " по " << xc.m;  int n = xc.getfirst();  while (n >= 0)  {  std::cout << std::endl << xc.nc << ": { ";  for (int i = 0; i < n; i++)  std::cout << AA[xc.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = xc.getnext();  };  std::cout << std::endl << "всего: " << xc.count() << std::endl;  system("pause");  return 0; |

Листинг 2

  
  
Генерация перестановок

|  |
| --- |
| setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " --- Генератор перестановок ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация перестановок ";  combi::permutation p(sizeof(AA) / 2);  \_\_int64 n = p.getfirst();  while (n >= 0)  {  std::cout << std::endl << std::setw(4) << p.np << ": { ";  for (int i = 0; i < p.n; i++)  std::cout << AA[p.ntx(i)] << ((i < p.n - 1) ? ", " : " ");  std::cout << "}";  n = p.getnext();  };  std::cout << std::endl << "всего: " << p.count() << std::endl;  system("pause");  return 0; |

Листинг 3



Генерация размещений

|  |
| --- |
| setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " --- Генератор размещений ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < N; i++)  std::cout << AA[i] << ((i < N - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация размещений из " << N << " по " << M;  combi::accomodation s(N, M);  int n = s.getfirst();  while (n >= 0)  {  std::cout << std::endl << std::setw(2) << s.na << ": { ";  for (int i = 0; i < 3; i++)  std::cout << AA[s.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = s.getnext();  };  std::cout << std::endl << "всего: " << s.count() << std::endl;  system("pause");  return 0; |

Листинг 4

Коммивояжёр с помощью перестановок

|  |
| --- |
| setlocale(LC\_ALL, "rus");  int d[N5][N5] = { //0 1 2 3 4 5 6 7 8 9  { 0, 45, INF, 25, 50, 40, 45, INF, 25, 50,}, // 0  { 45, 0, 55, 20, 100, 45, 20, 55, 20, 100}, // 1  { 70, 20, 0, 10, 30, 70, 20, 10, 10, 30}, // 2  { 80, 10, 40, 0, 10, 70, 20, 40, 10, 30}, // 3  { 30, 50, 20, 10, 0, 30, 50, 20, 10, 90}, // 4  { 30, 45, INF, 25, 50, 0, 45, 100, 25, 50,}, // 5  { 45, 50, 55, 20, 100, 45, 0, 55, 20, 100}, // 6  { 70, 20, 60, 10, 30, 70, 20, 0, 10, 30}, // 7  { 80, 10, 40, 70, 10, 70, 20, 80, 0, 30}, // 8  { 30, 50, 20, 10, 80, 30, 50, 20, 10, 0} // 9  };  int r[N5]; // результат  int s = salesman(  N5, // [in] количество городов  (int\*)d, // [in] массив [n\*n] расстояний  r // [out] массив [n] маршрут 0 x x x x  );  std::cout << std::endl << "-- Задача коммивояжера -- ";  std::cout << std::endl << "-- количество городов: " << N5;  std::cout << std::endl << "-- матрица расстояний : ";  for (int i = 0; i < N5; i++)  {  std::cout << std::endl;  for (int j = 0; j < N5; j++)  if (d[i][j] != INF) std::cout << std::setw(3) << d[i][j] << " ";  else std::cout << std::setw(3) << "INF" << " ";  }  std::cout << std::endl << "-- оптимальный маршрут: ";  for (int i = 0; i < N5; i++) std::cout << r[i] << "-->"; std::cout << 0;  std::cout << std::endl << "-- длина маршрута : " << s;  std::cout << std::endl;  system("pause");  return 0; |

Листинг 5