**СЕМИНАР 3. Подготовка к рейтингу 1**

**Теоретический материал**

Под магическим квадратом порядка N понимается квадратная матрица размером NxN из последовательных элементов натуральных чисел от 1 до N в квадрате, которые размещены так, что суммы элементов любого столбца, строки или главной диагонали одинаковы. Результат вычисления любой из перечисленных сумм принято называть константой магического квадрата. Порядок магического квадрата определяется числом элементов любого столбца или строки.

Страна, в которой был впервые придуман магический квадрат, точно неизвестна, неизвестен век, даже тысячелетие нельзя установить точно. Первые упоминания о магических квадратах были у древних китайцев.

Составление магических, или волшебных, квадратов – старинный и ещё сейчас весьма распространённый вид математических развлечений. Задача состоит в отыскании такого расположения последовательных чисел по клеткам квадрата, чтобы суммы чисел во всех строках, столбцах и по обеим диагоналям квадрата были одинаковы. Ученые создавали огромные квадраты, например, 43\*43,  содержащие числа от 1 до 1849, причем обладающие помимо указанных свойств магических квадратов, еще многими дополнительными свойствами.

 В XX вв. интерес к магическим квадратам вспыхнул с новой силой. Их стали исследовать с помощью методов высшей алгебры. Полного описания всех возможных магических квадратов не получено и до сего времени.

Самый простой вариант магического квадрата – квадрат порядка 1, который представляет собой одну единственную клетку заполненную значением 1. Магических квадратов 2\*2 не существует. Существует единственный магический квадрат 3\*3, так как остальные магические квадраты 3\*3 получаются из него либо перестановкой строк, или столбцов, либо путем поворота исходного квадрата на 900, или на 1800. С увеличением размеров квадрата быстро растет количество возможных магических квадратов такого размера. Например, существует 880 магических квадратов порядка 4 и 275 000 000 магических квадратов порядка 5.

**Применение магических квадратов**

Одной из возможных областей применения магических квадратов быть, например, защита информации. Шифруемый текст вписывали в магические квадраты нужного размера в соответствии с нумерацией их клеток. Если затем выписать содержимое такой таблицы по строкам, то получится шифртекст, сформированный благодаря перестановке букв исходного сообщения.

Еще один из возможных способов применения магического квадрата – при изготовлении жк телевизоров Toshiba.

Магический квадрат образует группу из 16 пикселей (квадрат 4х4) в котором в каждый момент времени с определенным интервалом зажигается одно и тоже число разных пикселей по горизонтале, вертикале и диагонали. Это обеспечивает планые цветовые переходы незаметные человеческому глазу и дает изображение на экран.

Кроме того возможно не самый полезный, однако одной из самых популярных областей применения магических квадратов является головоломка Судоку.

Магические квадраты также используются и астрологами. Великий Пифагор, например, считал, что всем миром управляют числа и разработал свой магический квадрат который по его мнению по дате рождения может многое рассказать о характере человека. Однако, насколько мне известно проведенные исследования показали, что сведения, полученные о человеке из магического квадрата Пифагора далеко не всегда совпадают с реальностью.

**Способы составления магических квадратов.**

Наибольший практический интерес представляют универсальные методы, которые не зависят от порядка магического квадрата. Однако, общий метод построения квадратов неизвестен, хотя широко применяются различные частные алгоритмы. Некоторые из них мы рассмотрим сегодня. Правила построения магических квадратов делятся на три категории в зависимости от того, каков порядок квадрата:

* нечетный;
* четный, порядок которого равен степени числа 2;
* четный, порядок которого равен удвоенному нечетному;
* четный, порядок которого равен учетверенному нечетному;

**Составление магических квадратов нечетного порядка**

Наиболее наглядный из них удобно рассмотреть на примере составления магического квадрата 5-го порядка из натуральных чисел от 1 до 25. Алгоритм этого метода включает следующие шаги.

1).  Сначала исходный (пустой) квадрат достраивается до симметричной ступенчатой ромбовидной фигуры как показано на рисунке, где ячейки для элементов квадрата обозначены символом 0, а достроенные ячейки - символом \*.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | \* |  |  |  |  |
|  |  |  | \* | \* | \* |  |  |  |
|  |  | 0 | 0 | 0 | 0 | 0 |  |  |
|  | \* | 0 | 0 | 0 | 0 | 0 | \* |  |
| \* | \* | 0 | 0 | 0 | 0 | 0 | \* | \* |
|  | \* | 0 | 0 | 0 | 0 | 0 | \* |  |
|  |  | 0 | 0 | 0 | 0 | 0 |  |  |
|  |  |  | \* | \* | \* |  |  |  |
|  |  |  |  | \* |  |  |  |  |

2). Полученная на шаге 1 фигура заполняется по косым рядам сверху-вниз-направо целыми числами от 1 до 25 в натуральном порядке. Результат заполнения показан на следующем рисунке:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **1** |  |  |  |  |
|  |  |  | **6** | \* | **2** |  |  |  |
|  |  | **11** | 0 | **7** | 0 | **3** |  |  |
|  | **16** | 0 | **12** | 0 | **8** | 0 | **4** |  |
| **21** | \* | **17** | 0 | **13** | 0 | **9** | \* | **5** |
|  | **22** | 0 | **18** | 0 | **14** | 0 | **10** |  |
|  |  | **23** | 0 | **19** | 0 | **15** |  |  |
|  |  |  | **24** | \* | **20** |  |  |  |
|  |  |  |  | **25** |  |  |  |  |

3). Каждое число, расположенное в фигуре шага 2 вне исходного квадрата, переносится по вертикали или горизонтали внутрь исходного квадрата на число клеток, равное порядку квадрата – в данном случае на 5 клеток.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **11** | **24** | **7** | **20** | **3** |
| **4** | **12** | **25** | **8** | **16** |
| **17** | **5** | **13** | **21** | **9** |
| **10** | **18** | **1** | **14** | **22** |
| **23** | **6** | **19** | **2** | **15** |

4). Освободившиеся  ячейки,  заполненные символом  \*,  должны  быть  исключены.

Оставшиеся  внутренние  ячейки, заполненные натуральными  числами, образуют магический квадрат. Сумма чисел в столбцах, строках, диагоналях равна 65.

Это далеко не единственный способ заполнения магических квадратов нечетного порядка. В лабораторных работах в каждом варианте задания приведен один из таких алгоритмов.

**Составление магических квадратов четного порядка**

Как было сказано ранее не существует универсальных алгоритмов составления магических квадратов четного порядка. Однако существуют индивидуальные подходы к решению различных задач.

Ниже рассмотрен метод составления магических квадратов, порядок которых является экспонентой 2. Этот метод удобно рассмотреть на примере магического квадрата 8-го порядка из натуральных чисел от 1 до 64. Метод включает следующую последовательность шагов.

1. Исходный квадрат делится на соответствующее число квадратов порядка 4. В данном случае таких квадратов будет 4. В каждом подквадрате отмечаются диагональные элементы (например, символом #). Остальные элементы построчно заполняются порядковыми целыми числами в направлении слева-направо и сверху-вниз. Числа, приходящиеся на выделенные диагональные элементы должны быть пропущены. Результат заполнения недиагональных элементов квадрата 8-го порядка показан на следующем рисунке:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | 2 | 3 | # | # | 6 | 7 | | # | |
| 9 | # | # | 12 | 13 | # | # | 16 | |
| 17 | # | # | 20 | 21 | # | # | 24 | |
| # | 26 | 27 | # | # | 30 | 31 | # | |
| # | 34 | 35 | # | # | 38 | 39 | # | |
| 41 | # | # | 44 | 45 | # | # | 48 | |
| 49 | # | # | 52 | 53 | # | # | 56 | |
| # | 58 | 59 | # | # | 62 | 63 | # | |

2. Отмеченные на шаге 1 диагональные элементы квадрата заполняют пропущенными целыми числами в порядке возрастания в направлении справа-налево и снизу-вверх. Недиагональные элементы в каждом подквадрате должны быть отмечены (например, символом $), а числа, приходящиеся на них должны быть пропущены. Результат заполнения диагональных элементов для квадрата 8-го порядка показан на следующем рисунке:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 64 | $ | $ | 61 | 60 | $ | $ | 57 |
| $ | 55 | 54 | $ | $ | 51 | 50 | $ |
| $ | 47 | 46 | $ | $ | 43 | 42 | $ |
| 40 | $ | $ | 37 | 36 | $ | $ | 33 |
| 32 | $ | $ | 29 | 28 | $ | $ | 25 |
| $ | 23 | 22 | $ | $ | 19 | 18 | $ |
| $ | 15 | 14 | $ | $ | 11 | 10 | $ |
| 8 | $ | $ | 5 | 4 | $ | $ | 1 |

3. Квадраты с пропусками диагональных и недиагональных элементов, полученные на шагах 1 и 2, объединяются в общий квадрат, где целочисленные элементы подавляют метки # или $. Результат объединения для квадрата 8-го порядка показан на следующем рисунке:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 64 | 02 | 03 | 61 | 60 | 06 | 07 | 57 |
| 09 | 55 | 54 | 12 | 13 | 51 | 50 | 16 |
| 17 | 47 | 46 | 20 | 21 | 43 | 42 | 24 |
| 40 | 26 | 27 | 37 | 36 | 30 | 31 | 33 |
| 32 | 34 | 35 | 29 | 28 | 38 | 39 | 25 |
| 41 | 23 | 22 | 44 | 45 | 19 | 18 | 48 |
| 49 | 15 | 14 | 52 | 53 | 11 | 10 | 56 |
| 8 | 58 | 59 | 5 | 4 | 62 | 63 | 1 |

Константа этого магического квадрата равна 260, что подтверждается вычислением контрольных сумм элементов по строкам, столбцам и главным диагоналям.

**Разбор программы**

Разработать ООП для построения магического квадрата из последовательных натуральных чисел по методу коня в следующем варианте. Начальное значение 1 нужно записать в среднюю клетку нижней строки квадрата. Запись чисел в остальные клетки должна происходить по ходу шахматного коня на 2 клетки вверх и на 1 клетку вправо. Если эта клетка уже занята, то следующее число нужно записать на 1 клетку ниже, чем предыдущее. Когда число оказывается за границей квадрата, его необходимо перенести внутрь квадрата, изменив заграничную координату на порядок квадрата. Результат построения квадрата должен быть отображен через стандартный вывод. Значение порядка квадрата должно передаваться программе через аргумент командной строки. При разработке программы нужно реализовать класс магического квадрата с приватными полями для матрицы и порядка квадрата, а также публичными методами его заполнения и отображения. Конструктор и деструктор класса должны обеспечивать динамическое распределение памяти квадрата.

1: #include <stdlib.h>

2: #include <stdio.h>

3:

4: class Magic

5: {

6: private:

7: unsigned\*\* tab;

8: int row;

9: int col;

10: int degree;

11: public:

12: Magic(int);

13: ~Magic();

14: void print();

15: void horse3();

16: int reflect(int);

17: };

18:

19: Magic::Magic(int n)

20: {

21: degree = n;

22: tab = new unsigned\* [degree];

23: for(row=0; row < degree; row++)

24: tab[row] = new unsigned [degree];

25: for(row=0; row < degree; row++)

26: for(col=0; col < degree; col++)

27: tab[row][col] = 0;

28: }

29:

30: Magic::~Magic()

31: {

32: for(row=0; row < degree; row++)

33: delete [] tab[row];

34: delete []tab;

35: }

36:

37: void Magic::print()

38: {

39: int degree2;

40: int len=0;

41: degree2 = (degree \* degree);

42: while(degree2 > 0)

43: {

44: degree2 /= 10;

45: len++;

46: }

47: for(row=0; row < degree; row++)

48: {

49: for(col=0; col < degree; col++)

50: printf("%0\*d ", len, tab[row][col]);

51: putchar('\n');

52: }

53: putchar('\n');

54: return;

55: }

56:

57: int Magic::reflect(int k)

58: {

59: if(k < 0)

60: return(k + degree);

61: if(k > (degree - 1))

62: return(k - degree);

63: return(k);

64: }

65:

66: void Magic::horse3()

67: {

68: int i;

69: int j;

70: int degree2 = (degree\*degree);

71: int z = 1;

72: row = (degree - 1);

73: col = (degree / 2);

74: tab[row][col] = z;

75: while(z < degree2)

76: {

77: i = reflect(row - 2);

78: j = reflect(col + 1);

79: if(tab[i][j] > 0)

80: {

81: i = reflect(row + 1);

82: j = reflect(col);

83: }

84: row = i;

85: col = j;

86: ++z;

87: tab[row][col] = z;

88: }

89: return;

90: }

91:

92: int main(int argc, char\* argv[])

93: {

94: if(argc != 2)

95: return(puts("Usage: magic degree"));

96: int n = atoi(argv[1]);

97: if((n % 2) == 0)

98: {

99: puts("Usage: magic 5 (or 7, 11, 17, 19, 23, ...)");

100: return(n);

101: }

102: Magic mag(n);

103: mag.horse3();

104: mag.print();

105: return(n);

106: }