**Диагонали квадратной матрицы**

**Обработка элементов квадратной матрицы, относительно диагоналей**

1. Работа с элементами, расположенными выше главной диагонали
	1. **for** i := 1 **to** n **do**
	2. **for** j := 1 **to** n **do**
	3. **if** i < j **then**
	4. *{работа с элементами}*

**Примечание**: данный способ не является рациональным, так как перебираются все элементы массива (а к нужным приходится обращаться через условный оператор). Рациональный способ решается путем введения зависимости начального или конечного значений параметра внутреннего цикла от значений параметра внешнего цикла. Предыдущая задача решается следующим образом:

* 1. **for** i := 1 **to** n - 1 **do**
	2. **for** j := i + 1 **to** n **do**
	3. *{работа с элементами}*

**Пример.** Заполнить элементы квадратной матрицы лещащих на главной диагонали и выше её единицами:

1 1 1 1 1 1

0 1 1 1 1 1

0 0 1 1 1 1

0 0 0 1 1 1

0 0 0 0 1 1

0 0 0 0 0 1

* 1. **const**
	2. n = 6; *{элементов в матрице}*
	3. **var**
	4. i, j:**byte**;
	5. matrix:**array**[1..n, 1..n] **of** **integer**;
	6. **begin**
	7. **for** i := 1 **to** n **do**
	8. **begin**
	9. **for** j := i **to** n **do**
	10. matrix[i, j] := 1; *{реализация}*
	11. **for** j := 1 **to** n **do**
	12. write(matrix[i, j]:2); *{вывод}*
	13. writeln
	14. **end**
	15. **end**.
1. Работа с элементами находящимися ниже и на главной диагонали
	1. **for** i := 1 **to** n **do**
	2. **for** j := 1 **to** i **do**
2. Работа с элементами находящимися выше и на побочной диагонали
	1. **for** i := 1 **to** n **do**
	2. **for** j := 1 **to** n - i + 1 **do**
3. Работа с элементами находящимися ниже и на побочной диагонали
	1. **for** i := 1 **to** n **do**
	2. **for** j := n - i + 1 **to** n **do**

Рассмотрим пример работы с элементами, находящимися на диагоналях.
**Пример 1.** Заполнить матрицу по образцу:

 1 1 1 1 1 1 1

 0 1 1 1 1 1 0

 0 0 1 1 1 0 0

 0 0 0 1 0 0 0

 0 0 1 1 1 0 0

 0 1 1 1 1 1 0

 1 1 1 1 1 1 1

* 1. **uses**
	2. crt;
	3. **const**
	4. n = 7;
	5. **var**
	6. i, j: **byte**;
	7. matrix: **array**[1..n, 1..n] **of** **byte**;
	8. **begin**
	9. clrscr; *{Очистка экрана}*
	10. **for** i := 1 **to** n **div** 2 **do**
	11. **for** j := i **to** (n + 1) - i **do**
	12. matrix[i,j] := 1;
	13. *{Верх}*
	14. **for** i := n **div** 2 + 1 **to** n **do**
	15. **for** j := (n + 1) - i **to** i **do**
	16. matrix[i,j] := 1;
	17. *{Низ}*
	18. **for** i := 1 **to** n **do** *{Вывод массива}*
	19. **begin**
	20. **for** j := 1 **to** n **do**
	21. write(matrix[i,j]:2);
	22. writeln
	23. **end**;
	24. readkey *{Ожидание нажатия клавиши}*
	25. **end**.

**Пример 2.** Определить суммы элементов, находящихся на диагоналях квадратной матрицы.

* 1. **const**
	2. n = 6;
	3. **var**
	4. i:**byte**;
	5. matrix:**array**[1..n, 1..n] **of** **integer**;
	6. sump, sumg:**integer**;
	7. **begin**
	8. ...
	9. *{Заполнение массива}*
	10. sumg := 0; *{Сумма на главной}*
	11. sump := 0; *{Сумма на побочной}*
	12. **for** i := 1 **to** n **do**
	13. **begin**
	14. sumg := sumg + matrix[i, i];
	15. sump := sump + matrix[i, n - i + 1];
	16. **end**
	17. ...
	18. *{Вывод сумм и прочее}*

**Домашнее задание**

1. **Заполнить матрицу по образцу:**
2. **1 0 0 0 0 0 1**
3. **1 1 0 0 0 1 1**
4. **1 1 1 0 1 1 1**
5. **1 1 1 1 1 1 1**
6. **1 1 1 0 1 1 1**
7. **1 1 0 0 0 1 1**
8. **1 0 0 0 0 0 1**