**Практическое занятие**

**по выполнению преобразований в различных системах счисления,**

**выполнению расчётов с логическими операциями**

**и решению задач программирования на условном языке или с помощью блок-схем.**

1. **Преобразования чисел в различных системах счисления.**
	1. Общие вопросы.

Наиболее простой и понятной нам системой счисления является **десятичная** (десятиричная). На самом деле она ничем не отличается от остальных систем по сложности выполнения расчётов, просто мы к ней привыкли. Тем не менее, эта система счисления является основной для большинства людей и наиболее распространенной по всему миру. Десятичная система счисления относится к так называемым **позиционным** системам счисления – *это системы, в которых значение числа зависит не только от символа (символов), которыми число отображается, но и от позиции этого символа (этих символов) в числе.*

Однако для вычислительной техники предпочтительнее использовать **двоичную** систему счисления, так как информация в компьютере представлена в виде электрических сигналов, которые имеют всего два варианта существования: либо электрический сигнал есть – обозначается 1, либо электрического сигнала нет – обозначается 0. Кроме этого для компактной записи информации в компьютере используется **шестнадцатиричная** система счисления, которая позволяет меньшим количеством символов записать большее количество информации.

Наряду с этими системами могут существовать и восьмиричная, и семеричная, и любая другая система счисления. Если понять принцип организации этих систем, то нет никакой разницы, в какой из систем представлено число, и можно довольно просто произвести преобразование числа из одной системы в другую.

* 1. Двоичная система счисления

Для понимания принципов работы в двоичной системе, как правило, ограничиваются 1 байтом, т.е. двоичным числом, состоящим из 8 разрядов. Такое число позволяет выполнять операции в пределах десятичного числа 255 (25510 – для обозначения системы используется подстрочный индекс).

Рассмотрим структуру одного байта с точки зрения десятичной системы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | Степень числа 2 при наличии значения в разряде |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Значение разряда (если значение равно нулю, то и результат тоже равен нулю) |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | Результат при значении разряда, равного единице, равен числу 2, возведённому в степень указанную в верхней строке |

Суммируя все результаты нижней строки, получаем: 128+64+32+16+8+4+2+1=255.

Если наше число больше, то используются дополнительные разряды, дописываемые слева. Так, например, оперируя 2 байтами, можно работать с десятичными числами в пределах 65 535.

Для решения задачи пересчёта двоичной системы в десятичную существует много вариантов, и можно использовать любой, однако, нам кажется, нагляднее и удобнее следующий вариант:

Перевести двоичное число 100110102 в десятичное. Для этого подставим данное число в приведённую выше таблицу в средний ряд (Значение разрядов). Соответственно в нижнем ряду получим следующие значения и просуммируем их.

Стандартная таблица для пересчета:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

Подставляем в средний ряд вместо единиц наше число 100110102 и получаем следующие значения (если значение разряда равно нулю, следовательно, значение в ячейке нижнего ряда также равно нулю):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 128 | 0 | 0 | 16 | 8 | 0 | 2 | 0 |

Складываем: 128+16+8+2=154.

Обратный перевод: число 23510 перевести в двоичную систему. Логика решения задачи следующая: 1) число 235 не больше 255, следовательно, для записи его в двоичной системе одного байта будет достаточно; 2) смотрим, какое максимальное число из нижнего ряда таблицы не превышает данное (естественно, 128); 3) вычитаем 128 из 235 и в значении разряда этого числа в среднем столбике ставим 1 (если это число не помещается в наше, то ставиться 0);

4) далее анализируем остаток (107), от остатка отнимаем следующее число (64), помещающееся в остаток и в разряде этого числа ставим 1, и т.д. Итак, число 235 равно:

Шаги 2 и 3: 235-128=107 – ставим 1 в разряде, соответствующем 128

Шаг 4: 107-64=43 – ставим 1 в разряде, соответствующем 64; далее повторяем процедуру 4), пока не останется ноль:

43-32=11 - ставим 1 в разряде, соответствующем 32,

11- 16 – не отнимается, ставим 0 в разряде, соответствующем 16,

11-8 = 3- ставим 1 в разряде, соответствующем 8,

3 – 4 – не отнимается, ставим 0 в разряде, соответствующем 4,

3-2 = 1 - ставим 1 в разряде соответствующем 2,

1-1 = 0 - ставим 1 в разряде соответствующем 1.

Сводим все в таблицу:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 128 | 64 | 32 | 0 | 8 | 0 | 2 | 1 |

Проверяем: 128+64+32+8+2+1=235

Получилось двоичное число: 111010112

Может это и не самый быстрый способ, зато простой, наглядный и надёжный.

Если десятичное число достаточно большое, то можно применить следующий вид записи:

Пример перевода по общепринятой методике (перевести десятичное число 218 в двоичное):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Число | 218 | 109 | 54 | 27 | 13 | 6 | 3 | 1 |
| Делитель | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Остаток | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

Соберем остатки от деления в направлении, указанной стрелкой, начиная с последней единицы, и получим число в двоичной системе счисления: 21810 = 110110102

Сложение и вычитание двоичных чисел производится по стандартной классической схеме, например:

Сложение Вычитание

|  |  |
| --- | --- |
| + 0 1 1 0 1 1 1 0 0 0 1 1 1 0 0 1 1 0 1 0 0 1 1 1 | - 1 0 1 0 1 1 1 0 0 1 1 0 0 0 1 0 0 1 0 0 1 1 0 0 |

При сложении двоичных чисел сумма в каждом разряде не может быть больше 2, если число получается равным двум, то в разряде записывается 0, а 1 переходит в следующий (старший) разряд. Если получается больше 2 (например, складывали три числа, все равны 1, получилось 3), то в данном разряде записывается остаток от вычитания из получившегося числа двойки (в нашем примере из трех вычесть два, записываем 1), а 1 переходит в старший разряд.

**Примеры для повторения №1:**

1. 22710 преобразовать в двоичную систему
2. 011010112 преобразовать в десятичную систему
3. 110112 -101112 = ответ в десятичной системе.
	1. Шестнадцатиричная система счисления

Это позиционная система счисления, в которой каждый символ может принимать одно из 16 значений от **0** до **F**:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № п/п | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| значение | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

Для записи 1 байта в шестнадцатиричной системе достаточно двух разрядов и максимальное шестнадцатиричное число, записанное 1 байтом равно – FF.

Пересчёт шестнадцатиричной системы в двоичную удобнее выполнять через ту же таблицу значений разрядов, что и при переводе двоичной системы в десятичную:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

Первый шестнадцатиричный символ кодирует первые 4 разряда байта, а второй - вторые 4 разряда, таким образом, байт можно поделить пополам и представить в следующем виде:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Значение двоичного разряда |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | Пересчёт в десятичную систему |
| 8 | 4 | 2 | 1 | 8 | 4 | 2 | 1 | Пересчёт в шестнадцатиричную систему |
| 8+4+2+1=15=F | 8+4+2+1=15=F | Значение шестнадцатиричного разряда (FF16) |

Таким образом, мы можем довольно легко и наглядно записать любое десятичное число в шестнадцатиричной, двоичной и десятичной форме, например, E916 разделим на 2 части (Е в первой части байта и 9 во второй). Смотрим по таблице: E соответствует 14, а 9 соответствует 9, запишем это: Е16=1410 и 916=910

Теперь представим десятичные числа 14 и 9 в виде суммы чисел (как для записи в двоичной системе):

|  |  |
| --- | --- |
| E16=1410=8+4+2==11102 | 916=910=8+1=10012 |
| 8 | 4 | 2 | 0 | 8 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |

Теперь добавляем стандартную таблицу для перерасчета и находим десятичное число:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 128 | 64 | 32 | 0 | 8 | 0 | 0 | 1 |
| 128+64+32+8+1=23310 |

Таким образом: E916→111010012→23310

**Примеры для повторения №2:**

2.1. 13710→шестнадцатиричную→двоичную

2.2. 100101112→шестнадцатиричную→десятичную

2.3. CA16→двоичную→десятичную

* 1. Восьмиричная система счисления

Также является позиционной системой счисления с основанием, равным 8.

Для пересчёта восьмиричной системы в десятичную, используем стандартный, классический вариант. Суть его заключается в том, что для перевода чисел из любой системы счисления в десятичную можно использовать следующую схему:

A\*Xk, где

А – значение числа в разряде,

Х – значение основания системы счисления,

K – значение позиции разряда

Например, если число четырехзначное в восьмиричной системе, то его перевод будет выглядеть так:

А4\*83+ А3\*82+ А2\*81+ А1\*80

например:

3778→десятичную =3\*82+7\*81+7\*80= 192+56+7=25510

Обратный перевод немного более сложный (так как уже сложно выполнять вычисления «в уме» - см.стандартную таблицу), но также выполним с помощью простого подбора, если речь идёт о небольших числах, будем исходить максимум из трёхзначного числа в восьмиричной системе, например:

24110→ восьмиричную.

Предполагаем, что более трёх разрядов в восьмиричной системе для обозначения этого числа не понадобиться, третий разряд в восьмиричной системе равен значению числа А\*82=А\*64. Считаем, сколько раз по 64 может уложиться в 24110 целиком, получается 3\*64=192. Таким образом, значение первого разряда числа 24110 в восьмиричной системе будет равно 3. Находим остаток для определения второго разряда восьмиричной системы: 241-192=49. Смотрим, сколько раз по 81=8 может поместиться в 49, получается 6\*8=48, таким образом, значение второго разряда восьмиричной системы будет равно 6. У нас осталось значение первого разряда восьмиричной системы, равное 49-48=1. Таким образом: 24110→3618

Стандартная таблица для пересчета (подбора) в восьмиричной системе:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 87 | 86 | 85 | 84 | 83 | 82 | 81 | 80 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 097 152 | 262 144 | 32 768 | 4 096 | 512 | 64 | 8 | 1 |

Сложение и вычитание в восьмиричной системе также выполняются по классическим правилам, например:

|  |  |
| --- | --- |
| Сложение | Вычитание |
| +2 3 48 1 6 78 4 2 38 | - 2 3 48 1 6 78 4 58 |

При сложении восьмиричных чисел сумма в каждом разряде не может быть больше 8, если число получается равным восьми, то в разряде записывается 0, а 1 переходит в следующий разряд, если больше 8, то в данном разряде записывается остаток от вычитания из получившегося числа 8, а 1 переходит в следующий разряд.

Рассмотрим приведённый выше пример: складываем последние разряды 4+7=11, значит, в последнем разряде будет записано 11-8=3, а 1 переходит в старший разряд. Складываем вторые разряды: 1 (перешло из предыдущего разряда)+3+6=10, значит, во втором разряде запишем 10-8=2, и 1 переходит в старший разряд. И в третьем разряде запишем: 1(перешло из предыдущего разряда)+2+1=4. Таким образом, результат = 4238.

Вычитание выполняется аналогичным образом. Из 4 вычесть 7 нельзя. Берём 1 из старшего разряда и получаем в последнем разряде значение 8+4=12. Из 12-7=5, таким образом, значение последнего разряда равно 5. В старшем разряде осталось значение 3-1(заимствованное значение) =2. Из 2 вычесть 6 нельзя. Повторяем процедуру аналогичную для первого разряда: 8+2=10 и из 10-6=4, таким образом, во втором разряде будет число 4. В старшем разряде остаётся: 2-1(заимствованный разряд)=1, вычитаем 1-1=0. Результат: 458

**Примеры для повторения №3:**

* 1. 21810→восьмиричную→семиричную
	2. 2558→десятичную→шестнадцатиричную
	3. 011011112→восьмиричную
	4. 1678+2558
	5. 4558-2678

**Коварные вопросы по системам счисления:**

1. Сколько вариантов комбинаций можно закодировать 1 байтом?
2. Какое максимальное десятичное число можно записать 1 байтом?
3. Если десятичное число чётное, то в двоичной системе последним будет 0 или 1?
4. Какому разряду в двоичной системе соответствует 25?
5. Чему будет равно максимальное десятичное число, записанное одним байтом в шестнадцатиричной системе?
6. Чему будет равно десятичное число 256 в шестнадцатиричной системе?
7. Какое значение примет число последнего разряда, если в нём складываются 5 и 7 в восьмиричной системе?
8. **Логические операции.**

В основе работы вычислительной техники лежат логические операции, выполняемые над двоичными значениями. Это связано с тем, что электронные устройства (логические микросхемы) обрабатывают электрические сигналы, которые либо существуют – логическое обозначение - «1», либо не существуют - логическое обозначение - «0». В качестве основных выделяют три логические операции:

1. Логическое *отрицание* – «NOT» или «НЕ», либо знак отрицания «$¬$» или $\overbar{X}$ (чёрточка вверху обозначает – «НЕ» X)
2. Логическое *умножение* или *конъюнкция* – «AND» или «И», либо знак конъюнкции «^». В языках программирования также могут встречаться следующие обозначения: А && В, А & В, А · В, А AND В
3. Логическое *сложение* или *дизъюнкция* – «OR» или «ИЛИ», либо знак дизъюнкции «˅». В языках программирования также могут встречаться следующие обозначения: А||В, А|В, А+В, А OR В

Логическая операция отрицания унарная, т.е. может выполняться над одним значением. А логические операции конъюнкции и дизъюнкции бинарные, т.е. для их выполнения необходимо как минимум 2 значения.

Суть логических операций (функций) следующая:

1. Функция отрицания «НЕ» возвращает противоположное значение исходного числа, например: если А=1, то $\overbar{А}$=0 и наоборот: если А=0, то $\overbar{А}$=1. Двойное отрицание числа$ А:[¬¬$А], возвращает само число А.
2. Функция конъюнкции «И» принимает значение 1, если и только тогда, когда все переменные (или обе переменные) равны 1, (и, соответственно, равна 0, если хотя бы одна из этих переменных равна 0), например: если А=1, В=1, то А ˄ В=1; если А=1, а В=0, то А ˄ В=0, а также если А=0, а В=0, то А ˄ В=0
3. Функция дизъюнкции «ИЛИ» принимает значение 1, если хотя бы одна переменная (как минимум из двух) равна 1, (и, соответственно, равна 0, если все переменные равны 0), например: если А=1, В=1, то А ˅ В=1; также А=1, В=0, то А ˅ В=1, и только если А=0, В=0, то А ˅ В=0

Графическое обозначение логических операций:

|  |  |  |
| --- | --- | --- |
| Отрицание (инвентор) | Умножение (коньюнкция) | Сложение (дизъюнкция) |
| http://upload.wikimedia.org/wikipedia/commons/thumb/6/66/NOT_gate_RU.svg/80px-NOT_gate_RU.svg.png | http://upload.wikimedia.org/wikipedia/commons/thumb/d/df/AND_gate_RU.svg/100px-AND_gate_RU.svg.png | http://upload.wikimedia.org/wikipedia/commons/thumb/3/3f/OR_gate_RU.svg/100px-OR_gate_RU.svg.png |

Таблица истинности.

Основные операции можно представить в единой таблице, которую чаще всего называют таблицей истинности:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | Y | not X | XandY | XorY | XxorY |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

**XxorY** – логическая операция, исключающая ИЛИ, фактически проверяет на соответствие два значения, если они равны, то возвращается 0, если не равны, то возвращается 1.

Примеры выполнения расчётов с логическими операциями.

Логическое умножение одного байта на другой выполняется побитно, например:

|  |  |
| --- | --- |
| X= | 010111012 |
| Y= | 001101012 |
| X AND Y | 000101012 |

Аналогично выполняется и логическое сложение:

|  |  |
| --- | --- |
| X= | 010111012 |
| Y= | 001101012 |
| X OR Y | 011111012 |

Пример 1.

Решения задачи следующего вида: ($¬А $ ˄ В)˅(А ˄ В)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | В | $$¬А$$ | $¬А $ ˄ В | А ˄ В | ($¬А $ ˄ В)˅(А ˄ В) |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 |

Данный пример решается следующим образом:

Вначале, исходя из приоритета логических операций, находится значение$ ¬А$, затем находится значение $¬А $ ˄ В, затем находится значения А ˄ В и на последнем этапе определяется значение ($¬А $ ˄ В)˅(А ˄ В). Таким образом мы нашли ответы для заданного примера для всех возможных исходных вариантов А и В.

Пример 2.

Заданы логические выражения:

1. x<y OR x<z
2. x<y AND y<z
3. y>x AND z>x

Если число x имеет минимальное значение среди попарно неравных трёх чисел x,y и z , то значение ИСТИНА принимают выражения?

Выражение [а.] будет истинно так как x<y и x<z т.е. обе части уравнения будут истины.

Выражение [b.] может быть как истинным, так и ложным, так как x<y, а вот у<z мы не знаем, а уравнение будет истинным, если обе части будут истинными.

Выражение [с.] будет истинным, так как и y и z больше x, т.е. обе части уравнения будут истинными, и в результате получим истину.

Таким образом, ответ будет: a,c

**Примеры для повторения №4:**

1. с=а ˄в; а= 11011011 и в=00111010, чему будет равно с ?
2. (А ˅ В) ˄ ($\overbar{А}˅\overbar{ В}$), если А=1, а В=0

|  |  |  |
| --- | --- | --- |
| А | В | (НЕ А ИЛИ НЕ В) И А |
| 0 | 0 |  |
| 0 | 1 |  |
| 1 | 0 |  |
| 1 | 1 |  |

1. **Решение задач программирования на условном языке или с помощью блок-схем.**

Для понимания общих основ программирования, а также в учебных целях очень часто используются различные элементы программирования в виде небольших программ или части программ, выполненных на условном языке программирования, или в виде блок-схемы. Для решения таких задач не обязательно владеть каким-либо языком программирования и писать на нём программы. Достаточно понимать суть (смысл) выполняемых действий, понимать логику последовательности выполнения действий, и понимать обозначения элементов такой программы.

Рассмотрим несколько примеров:

Пример 1.

Задан одномерный массив X1, X2, ,XN . Что определяет фрагмент алгоритма?



Фрагмент алгоритма программы будет выполняться последовательно в следующем порядке:

1. R=1 – операция присваивания, заносит в переменную R число 1.
2. i=2 - операция присваивания, заносит в переменную i число 2.
3. запись в ромбе- логическое условие, проверяет значение переменной i на равенство или неравенство (< -меньше) переменной N. Переменная N является индексом массива и определяет количество элементов в исходном массиве. Пока значение i будет меньше или равно N программа будет работать, как только значение i превысит значение N, работа программы закончится. Чаще всего i является либо программным счётчиком, либо счётчиком элементов. В нашем случае i является номером элементов массива (в следующем условии записано Xi) и выбирает из массива элемент под номером (индексом), равным i. Таким образом, первым из массива будет выбран элемент под №2 (т.е. чётным номером – X2).

i<=N

1. Следующим действием фрагмента алгоритма будет проверка значения элемента массива X2 на условие [<0]. Далее программа разветвляется на два различных варианта дальнейшей работы в зависимости от результата проверки этого условия. Если этот элемент будет меньше 0, т.е. отрицательным (условие верно), то в этом случае начнёт выполняться пункт 6, а если положительным (условие не верно), то будет выполняться пункт 7.

Xi <0

1. R =R\* X2 - в этом действии исходное значение R=1 будет умножено на значение элемента X2 и записано в переменную R.
2. i=i+2 -следующим действием будет изменение индекса массива на [+2], т.е. i=4. После переходим (возвращаемся на пункт 3) на проверку условия проверки индекса массива на наличие такого номера элемента в массиве. Если такой номер элемента есть, программа будет выполняться по кругу (в цикле) пункты 4, 5, 6, 7, пока не закончатся чётные элементы массива. Как только чётные элементы массива закончатся работа программы прекратиться.

***В результате работы этого фрагмента алгоритма получим произведение чётных элементов массива X(i)c отрицательными значениями***

Пример 2.

Задан двумерный массив А$\left(\genfrac{}{}{0pt}{}{6,9}{8,7}\right)$

После выполнения фрагмента алгоритма переменная В примет значение?

В:=10

**нц для** I **от** 1 **до** 2

**нц для** J **от** 1 **до** 2

**если** А(I,J)<B

**то** В= А(I,J)

**всё**

**кц**

**кц**

Обозначения:

* **нц для** I **от** 1 **до** 2 –это организация цикла (т.е. повторения части программы от слова **нц** (начало цикла) до слова **кц**- конец цикла) связанного с именно с этим началом цикла. Как правило, они выделяются по уровням отступа от левой границы абзаца, или же нужно просто искать по логике открытие и закрытие циклов; **кц** всегда закрывает ближайшее к нему **нц,** стоящее выше.
* переменная I и J – это переменные, управляющие циклом, в которые, в нашем случае, при первом выполнении цикла заносится значение 1 (I **от** 1 **до** 2), а при каждом новом проходе цикла значение переменной увеличивается на 1 (если не задано другое условие), и так до тех пор, пока значение переменной не примет конечное значение – в нашем случае – 2. Т.о. в данной записи цикл по переменной I выполнится 2 раза: при первом проходе I=1 и при втором I=2. После этого цикл закончиться, и программа будет выполняться дальше.

**Если** А(I,J)<B

**то** В= А(I,J)

**иначе**

**всё**

элемент программы – *условие* состоит из нескольких элементов:

* собственно условие **если,** в котором записывается логическое выражение, т.е. проверяется какое-либо равенство или неравенство.
* следующий элемент условия **то** – действие, которое будет выполнено, если записанное выше условие верно (выполняется)
* следующий элемент условия **иначе** – действие, которое будет выполнено, если условие ложно (не выполняется). В некоторых языках программирования вариантов **иначе** может быть несколько.
* элемент **всё** – завершение условия.

Теперь рассмотрим, как будет работать (выполняться) фрагмент приведённой выше программы:

1. **нц для** I **от** 1 **до** 2 – этой записью (строкой) организован цикл по переменной I, которая при первом проходе (выполнении) цикла принимает значение 1, цикл будет работать 2 раза, так как окончанием цикла является 2, или I=2
2. **нц для** J **от** 1 **до** 2 - этой записью организован ещё один, аналогичный предыдущему, цикл по переменной J, принимающей такие же значения (1 и 2). Циклы вложенные, это значит, что внутри первого цикла будет выполняться второй цикл. Аналогично, при первом проходе (выполнении) второго цикла J=1, а при втором J=2.
3. Третьей строкой фрагмента программы проверяется условие **если** А(I,J)<B, т.е. проверяется значение элемента массива АI,J, т.е. А1,1 (так как I=1 и J=1). Значение элемента массива А1,1=6 , а В=10, т.е. условие будет истинным (т.е. выполняется, ведь 6<10).
4. Следующей строкой фрагмента программы выполняется действие **то** В= А(I,J) (или операция присваивания), но только в том случае, если условие, записанное в предыдущей строке, будет истинным, в противном случае эта строка выполняться не будет (игнорируется), а программа сразу переходит (перескакивает) на следующую строку. Так как в нашем случае условие выполняется, то и данная строка выполниться, т.е. в переменную **В** будет записано значение элемента массива А1,1 =6, таким образом В тоже будет равно 6 (В=6).
5. Следующей строкой программы записано окончание условия **всё**.
6. Следующей строкой программы записано окончание второго цикла по J - **кц** .
7. Следующей строкой программы записано окончание первого цикла по I - **кц** .

Теперь рассмотрим работу всего фрагмента программы полностью, и определим значения, которые примут переменные, описанные в этом фрагменте.

1. проход: АI,J, т.е. А1,1 =6; В = 6
2. проход: АI,J, т.е. А1,2 =8; В = 6
3. проход: АI,J, т.е. А2,1 =9; В = 6
4. проход: АI,J, т.е. А2,2 =7; В = 6

На этом работа программы будет закончена. Значение переменной В=6, т.е. будет найден самый маленький элемент массива.

Пример 3.

В результате выполнения фрагмента блок-схемы алгоритма значения переменных X и Y примут значения?



1. Первым действием фрагмента алгоритма будет операция присваивания Х=0, т.е. в переменную с именем Х будет записано значение 0.
2. Вторым действием фрагмента алгоритма будет также операция присваивания Y=0, т.е. в переменную с именем Y будет записано значение 0.
3. Третьим действием фрагмента алгоритма будет проверка условия X>=Y. Если условие будет выполнено, т.е. значение, записанное в переменной Х, будет больше или равно значению, записанному в переменной Y, то в этом случае произойдёт переход к следующему действию. Если условие не выполниться, то будут выведены значения, записанные в переменных X и Y, и работа программы будет завершена.
4. Четвёртым действием будет проверка ещё одного условия Х<=4. Если условие будет выполнено, т.е. значение, записанное в переменной Х, будет меньше или рано 4, то в этом случае произойдёт переход к следующему действию, если условие не выполниться, то будут выведены значения, записанные в переменных X и Y, и работа программы будет завершена.
5. Пятым действием будет выполнено сложение значения записанного в переменной Х и 1 и результат будет записан в переменную Х (Х=Х+1). Т.о. значение переменной Х будет увеличено на 1.
6. Шестым действием будет расчёт значения Y = X\*X/2, т.е. в переменную Y будет записано значение Х, умноженное на это же значение и делёное на 2.
7. Последним действием будет переход (возврат) программы с новыми (рассчитанными в пунктах 5 и 6) значениями Х и Y к третьему действию (пункту) фрагмента алгоритма. Т.е. будет выполнен фрагмент алгоритма с пункта 3 повторно. И так будет происходить до тех пор, пока условия в пункте 3 и 4 будут выполняться (истинными). Как только условие в пункте 3 или 4 станет ложным, то будут выведены значения, записанные в переменных X и Y, и работа программы будет завершена.

Теперь, зная схему решения алгоритма, решим его с имеющимися у нас исходными значениями Х=0 и Y=0.

При первом проходе:

1. Х=0;
2. Y=0;
3. X>=Y или 0=0 → условие выполняется;
4. Х<=4 или 0<4 → условие выполняется;
5. Х=Х+1 или X=0+1=1, т.е. Х=1;
6. Y=X\*X/2 или Y=1\*1/2, т.е. У=0,5
7. Возврат к пункту 3

Второй проход:

1. X>=Y или 1>0,5 → условие выполняется;
2. Х<=4 или 1<4 → условие выполняется;
3. Х=Х+1 или X=1+1=2, т.е. Х=2;
4. Y=X\*X/2 – Y=2\*2/2, т.е. У=2
5. Возврат к пункту 3

Третий проход:

1. X>=Y или 2=2 → условие выполняется;
2. Х<=4 или 2<4 → условие выполняется;
3. Х=Х+1 или X=2+1=3, т.е. Х=3;
4. Y=X\*X/2 или Y=3\*3/2, т.е. У=4,5
5. Возврат к пункту 3

Четвёртый проход:

1. X>=Y или 3<4,5 → условие не выполняется;
2. Вывод значений Х=3 и Y=4,5 и завершение программы

Пример 4.

Если элементы массива D[1…5] равны соответственно 4,1,5,3,2, то значение выражения D[D[4]]-D[D[5]] равно :

1. В квадратных скобках записывается индекс (номер) элемента массива, под каждым номером элемента массива записываются различные значения. Таким образом, получаем:

D[1]=4

D[2]=1

D[3]=5

D[4]=3

D[5]=2

1. Подставим найденные значения D[4] и D[5] в формулу:

D[3]-D[2]

1. Теперь подставим в формулу D[3]-D[2] значения элементов массива 5-1=4

**Примеры для повторения №5:**

5.1. Укажите пропущенный фрагмент в алгоритме, определяющем количество чётных элементов в массиве А[1:N]. Операторы: DIV – деление нацело, MOD – остаток от деления.

S:=0; K:=0

**нц для**J**от** 1 **до**N

**если** \_\_\_\_\_\_\_\_\_\_

**то** S:=S+1

**всё**

**кц**

1. A[J] DIV 2 = 0
2. A[J] MOD 2
3. A[J] MOD 2=K
4. A[K] MOD 2=0

Ответ:

5.2. Дан фрагмент алгоритма, в котором обрабатывается массив А:(2,12,0,-3,0). Укажите значение переменной Y после окончания работы фрагмента алгоритма.

**целтаб** А[1:5]

Y:=0

**нцдля** К**от** 1 **до** 5

**если** А[k] >=0

**то**Y=Y+1

**всё**

**кц**

**вывод**Y

Укажите значение Y:

5.3. В результате выполнения фрагмента алгоритма значение переменной b будет равно?



b=

5.4. Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных: A=7; B=8; C=9