Рязанский станкостроительный колледж РГРТУ

Основы алгоритмизации и программирования

Тема 3. Базовые конструкции структурного программирования (Язык программирования C#)

Рязань 2022

Оглавление

Основные принципы технологии структурного программирования

Технологией программирования называют совокупность методов и средств, используемых в процессе разработки программного обеспечения.

Исторически в развитии программирования можно выделить несколько технологий программирования принципиально отличающихся методологий.

Изначально понятие технологии как таковой — это 60-е годы прошлого столетия — это период **"стихийного" программирования**. В этот период отсутствовало понятие структуры программы, типов данных и т.д. Вследствие этого код получался запутанным, противоречивым. Программирование тех лет считалось искусством. Для небольших программ это вполне годилось, но с увеличением размера программ это стало большим тормозом программирования.

Структурный подход к программированию представляет собой совокупность рекомендуемых технологических приемов, охватывающих выполнение всех этапов разработки программного обеспечения. В основе структурного подхода лежат три основных подхода:

- **декомпозиция** (разбиение на части) сложных систем с целью последующей реализации в виде отдельных небольших подпрограмм. С появлением других принципов декомпозиции (объектного, логического и т.д.) данный способ получил название процедурной декомпозиции.
- другим базовым принципом структурного программирования является использование при составлении программ только базовых алгоритмических структур (линейные, разветвляющиеся, циклические), запрет на использование оператора GOTO.
- повышения уровня типизации данных. Теория типов данных исходит из того, что каждое используемое в программе данное принадлежит одному и только одному типу данных. Тип данного определяет множество возможных значений данного и набор операций, допустимых над этим данным. Данное конкретного типа в ряде случаев может быть преобразовано в данное другого типа, но такое преобразование должно быть явно представлено в программе. Применение типизированного языка при написании программы позволяет еще при трансляции исходного текста выявить многие ошибки использования данных и этим повысить надежность программы

При применении этих методов появилась возможность читать и проверять программу как последовательный текст, что повысило производительность труда программистов при разработке и отладке программ.

Поддержка принципов структурного программирования была заложена в основу так называемых процедурных языков программирования. Как правило, они включали основные "структурные" операторы передачи управления, поддерживали вложение подпрограмм, локализацию и ограничение области "видимости" данных. Среди наиболее известных языков этой группы стоит назвать Pascal, C, C++.

Оформление кода программы в соответствии со стандартом кодирования

Основные положения

Стандарт является соглашением по оформлению и написанию кода на языке C#. В документе приведены основные правила оформления кода и приемы, используемые при написании программ.

Стандарт является обязательным для любых разработок программных систем, а также разработки учебных программ в рамках учебного процесса РССК РГРТУ.

Далее мы рассмотрим основные положения по оформлению кода на языке C#, более подробно смотрите файл «***Стандарт оформления кода C#***».

Стили наименования

PascalCase – первая буква каждого слова в имени идентификатора начинается с верхнего регистра.

Пример: TheCategory.

CamelCase – первая буква первого слова в идентификаторе в нижнем регистре, все первые буквы последующих слов – в верхнем.

Пример: theCategory.

Общие правила именования идентификаторов

Для названий любых элементов программного кода (модулей, классов и их методов, функций, переменных, констант, последовательностей, представлений), следует использовать содержательные имена, позволяющие сделать вывод о назначении данных элементов. При именовании идентификаторов не используются аббревиатуры или сокращения, если только они не являются общепринятыми.

Пример: GetWindow(), а не GetWin();

Если имя идентификатора включает в себя сокращение – сокращение пишется в PascalCase. Исключение - когда имя идентификатора должно быть указано в CamelCase и сокращение стоит в начале имени идентификатора. В этом случае сокращение пишется в нижнем регистре.

Пример:
SqlAccount для PascalCase, sqlAccount для CamelCase.

Использование верхнего и нижнего регистра в именах

Запрещается создавать два различных имени, функции или свойства с одинаковыми именами, отличающиеся только регистром. Запрещается создавать функции с именами параметров, отличающимися только регистром. Ниже приведены примеры НЕправильных названий.

Пример:
KeywordManager и Keywordmanager;
findByID(int id) и FindByID(int id);

Правила именования переменных

Для именования переменных используется стиль CamelCase;

Пример: int userId = 20;

Правила именования констант

Для именования констант используется стиль PascalCase.

Правила именования функций

Для именования функций используется стиль PascalCase;

Имена функций должны давать четкое представление о том, какое действие эта функция выполняет. Имя функции начинается с глагола, указывающего на то, какое действие она выполняет.

Пример: public void CreateItem();

Правила именования параметров функций

Для именования параметров используется стиль CamelCase;

Имена параметров должны давать четкое представление о том для чего используется параметр, и какое значение следует передать при вызове функции.

Пример:

public void EncodeString(string sourceString, ref string encodedString),

а не public void EncodeString(string string1, ref string string2).

Форматирование кода программы

Используются стандартные настройки форматирования Visual Studio.

Для улучшения наглядности структуры программного кода следует использовать пробелы и отступы:

- Пробелы – для лучшего обозначения отдельных элементов кода в строке.
- Горизонтальные отступы – для лучшего обозначения вложенности функциональных блоков многострочного кода. При формировании горизонтальных отступов рекомендуется использовать tab - 4 пробела.
- Вертикальные отступы – для отделения логических блоков программного кода между собой.

Фигурные скобки размещаются всегда на отдельной строке.

В условии if-else всегда используются фигурные скобки.

Использование строк длиннее 100 символов не желательно. При необходимости инструкция переносится на другую строку. При переносе части кода на другую строку вторая и последующая строки сдвигаются вправо на один символ табуляции.

Каждая переменная объявляется на отдельной строке.

Комментирование кода программы

Все комментарии должны быть на русском языке.

В комментариях не допускается использование слов, не относящихся к техническому или деловому лексикону.

В комментариях не допускается наличие информации, не несущей смысловую нагрузку к пояснению сути алгоритма программного кода. Например, недопустимы

комментарии вида: «это гениальный код», «таблица1», «!№;%:?*», «Это комментарий», «Эту фигню все равно никто не читает» и подобные им.

Если для языка программирования применяется система контроля версий, то комментарии вида: дата создания; дата модификации; автор создания; автор модификации - допускается вести в ней.

Комментарии к программному модулю, представленному в виде отдельного файла, необходимо указывать всегда, при этом обязательны следующие комментарии:

- ***Описание назначения.***
- Дата создания.
- Автор создания.
- Дата модификации (при необходимости).
- Автор модификации (при необходимости).
- Цель модификации (при необходимости).

Комментарии к переменным, заголовкам свойств, полей, интерфейсов и прочего необходимо указывать всегда, при этом обязательны следующие комментарии:

- ***Описание назначения.***

Комментарии к заголовкам функций необходимо указывать всегда, при этом обязательны следующие комментарии:

- ***Описание назначения.***
- ***Описание входных и выходных параметров (при их наличии).***

Для функций, выполняющих сложные алгоритмы, не очевидные для восприятия, необходимо указывать подробные комментарии не только к заголовку функции, но и самому алгоритму с пояснением каждого шага выполнения алгоритма.

Для удобства работы с большой глубиной вложенности программных структур рекомендуется добавлять следующие комментарии:

- Метка окончания каждого ветвления алгоритма с указанием типа ветвления и краткого текста с условием.
- Метка окончания каждого цикла с указанием типа цикла и названия переменной-счетчика.
- Метка окончания модуля, класса, метода класса, события класса, функции, процедуры, триггера, пакета, с указанием названия закончившегося элемента программного кода, если указание названия не предусмотрено синтаксисом используемого языка.

Комментарии к тексту программного кода должны быть лаконичными и пояснять суть и назначение программного кода, а не его техническую составляющую. Например, вместо комментария «увеличиваем счетчик на 1» нужно писать «переходим к обработке следующего факультета». Текст комментариев не должен иметь сокращений, понимание которых может быть неоднозначным или затруднительным.

Пример программного кода с соблюдением правил оформления кода (язык C++):

```
// 14.01.2020  Сидоров С.С. ИСП-21
// Практическая работа №12 Сортировка
// Сортировка строк двумерного массива методом пузырька
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <time.h>
using namespace std; //Установка стандартного пространства имен

//печать массива
void printMatrix
(int[][10], // Сортируемый массив
    const int // Размерность массива
);
int main()
{
    const int Size = 10; // Размерность массива
    int matrix[Size][Size]; // Определяем массив
    int temp; // Временная переменная
    srand(time(NULL)); // Определяем исходную точку для генератора случайных чисел

    // Заполняем массив случайным образом
    for (int i = 0; i < Size; i++)
        for (int j = 0; j < Size; j++)
            matrix[i][j] = 1 + rand() % 100;

    // Сортируем пузырьком
    for (int n = 1; n < Size * Size; n++)
    {
        for (int i = 0; i < Size; i++)
        {
            for (int j = 0; j < Size - 1; j++)
            {
                // Если нужно скорректировать порядок сортировки
                if (matrix[i][j + 1] < matrix[i][j])
                {
                    // Делаем обмен элементов массива местами
                    temp = matrix[i][j + 1];
                    matrix[i][j + 1] = matrix[i][j];
                    matrix[i][j] = temp;
                }// for (int j = 0; j < Size - 1; j++)
            }// for (int i = 0; i < Size; i++)
        }// for (int n = 1; n < Size * Size; n++)

        //Выводим массив на печать
        printMatrix(matrix, Size);

        return 0;
    }
}// end of main
```

```
// Печать массива
// Входные знвчения:
// varMX[][10] - сортируемый массив
// Size - размер массива
void printMatrix(int varMX[][10], const int size)
{
    for (int i = 0; i < size; i++)
    {
        cout << endl;
        for (int j = 0; j < size; j++) // Печать j-массива на одной строке
            cout << setw(4) << varMX[i][j];
        cout << endl; // Каждый j-массив на отдельной строке
    }
}// end of printMatrix
```

Программирование алгоритмов разветвляющей структуры

Алгоритм ветвящейся структуры это такой алгоритм, в котором выбирается один из нескольких возможных путей (вариантов) вычислительного процесса.

Каждый подобный путь называется **ветвью алгоритма.**

Признаком разветвляющегося алгоритма является наличие операций условного перехода, когда происходит проверка истинности некоторого логического выражения (проверяемое условие) и в зависимости от истинности или ложности проверяемого условия для выполнения выбирается та или иная ветвь алгоритма.

Для программной реализации этого типа алгоритмов в языке программирования Паскаль можно использовать три оператора:

1. Оператор безусловного перехода (использовать не рекомендуется).
2. Условный оператор (оператор условного перехода).
3. Оператор выбора (оператор варианта).

Рассмотрим все эти операторы подробнее.

Оператор безусловного перехода

Общий формат записи оператора **goto**:

goto *метка*;

.

.

***метка*: заданные действия.**

GOTO - зарезервированное слово (перейти на метку). Действие оператора GOTO состоит в передаче управления соответствующему отмеченному оператору.

Метка это идентификатор с последующим двоеточием, позволяющий именовать некоторый оператор программы и таким образом ссылаться на него.

```
int i, a = 0;
goto m1;
for (i = 1; i < 1000000000; i = i + 1) { a = a + 1; };
m1:
Console.ReadKey();
```

Рисунок 1 - Использование оператора goto

Условные выражения

Условные выражение позволяет проверить некоторое условие и в зависимости от результатов проверки выполнить то или иное действие. Для проверки условия используются различные логические операции и операции сравнения.

Таблица 1. Логические операции и операции сравнения.

Название	Знак операции	Пример записи	Пояснение
Логическое умножение (**И**)	**&&**	**X>5 && Y<2**	Результат — истина, если истинны оба операнда
Логическое сложение (**ИЛИ**)	**||**	**X>5 || Y<2**	Результат — истина, если истинен хотя один операнд
Логическое отрицание (**НЕ**)	**!**	**!X>2**	Если **X** — истина, то результат — ложь и наоборот
Исключающеюся (**ИЛИ**)	^	**X>5 ^ Y<2**	Результат — истина, если либо первый, либо второй операнд (но не одновременно) истинен
Меньше	<	**X < Y**	Результат — истина, если X меньше Y
Меньше или равно	<=	**X <= Y**	Результат — истина, если X меньше или равен Y
Больше	>	**X >Y**	Результат — истина, если X больше Y
Больше или равно	>=	**X >= Y**	Результат — истина, если X больше или равен Y
Проверка на равенство	==	**X == Y**	Результат — истина, если X равен Y (Обычно целые числа)
Проверка на неравенство	**!=**	**X != Y**	Результат — истина, если X неравен Y (Обычно целые числа)

Таблица 2. Приоритеты операции отношений в убывающей последовательности.

№ п/п	Оператор
1	**!**
2	> >= < <=
3	== **!=**
4	**&&**
5	**||**

Как и в арифметических выражениях, для изменения порядка выполнения операций сравнения и логических операций можно использовать круглые скобки.

Оператор if

Условный оператор if (в переводе с англ. — «если») реализует выполнение определённых команд при условии, что некоторое логическое выражение (условие) принимает значение «истина» true.

Общая форма записи оператора **if**:

if (условное выражение) однострочное действие;

В операторе **if** используется результат вычисления условия, заключенного в круглые скобки, на основе которого принимается решение. Результат вычисления условного выражения может быть арифметическим или логическим. Если результат выполнения условно выражения будет истинным (true), то выполняется действие, указанное в операторе.

Если нужно выполнить несколько действий для этого следует использовать фигурные скобки, например:

```
if (условное выражение)
{
Действие 1;
Действие 2;
…
}
```

Конструкция if – else

Условный оператор if - else реализует выполнение одной последовательности команд при условии, что некоторое логическое выражение (условие) принимает значение «истина» true и другой последовательности команд если условие принимает значение «ложь» false.

Общая форма записи конструкции **if – else**:

```
if (условное выражение)  Действие 1;
else Действие 2;
```

Если истинно условное выражение, то будет выполняться фрагмент программы **Действие 1**, в противном случае произойдет переход к **Действию 2**.

Если нужно выполнить несколько действий для этого следует использовать фигурные скобки, например:

```
if (условное выражение)
{
   Действие 1-1;
   Действие 1-2;
   …
}
else
{
   Действие 2-1;
   Действие 2-2;
   …
}
```

Пример 1.

Формулировка задачи:

Написать программу расчёта функции:

$$y = \begin{cases} \cos^2 x, \text{при } 0 < x < 2 \\ 1 - \sin x^2, \text{иначе.} \end{cases}$$

Выбор метода решения:

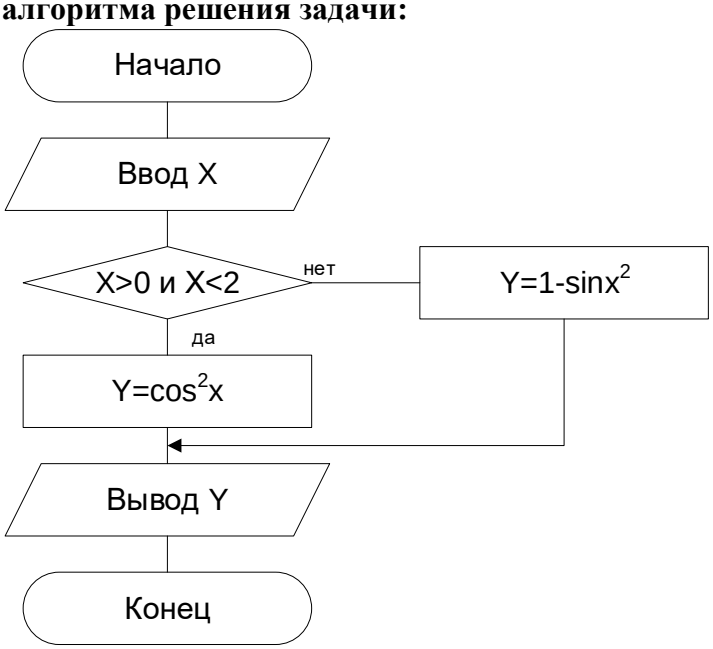
Метод решения прост, если значение ячейки X больше 0 и меньше 2, то рассчитываем значение У по первой формуле, иначе по второй формуле.

Решение:

Исходные данные: X.

Результат: Y.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример1. Программа расчёта функции

double x, y;
Console.Write("Введите X= ");//Подсказка о вводе
x = Convert.ToInt32(Console.ReadLine());//Ввод X
if (x > 0 && x < 2)
{
    y = Math.Cos(x) * Math.Cos(x);
}
else
{
    y = 1 - Math.Sin(x * x);
}
Console.WriteLine("Ответ Y= " + y);//Вывод ответа
Console.ReadKey();//Пауза
```

Пример 2.

Формулировка задачи:

Написать программу, которая в переменную max записывает наибольшее из двух чисел x и y.

$$\max = \begin{cases} x, если\ x > y \\ y, если\ y \geq x \end{cases}$$

Выбор метода решения:

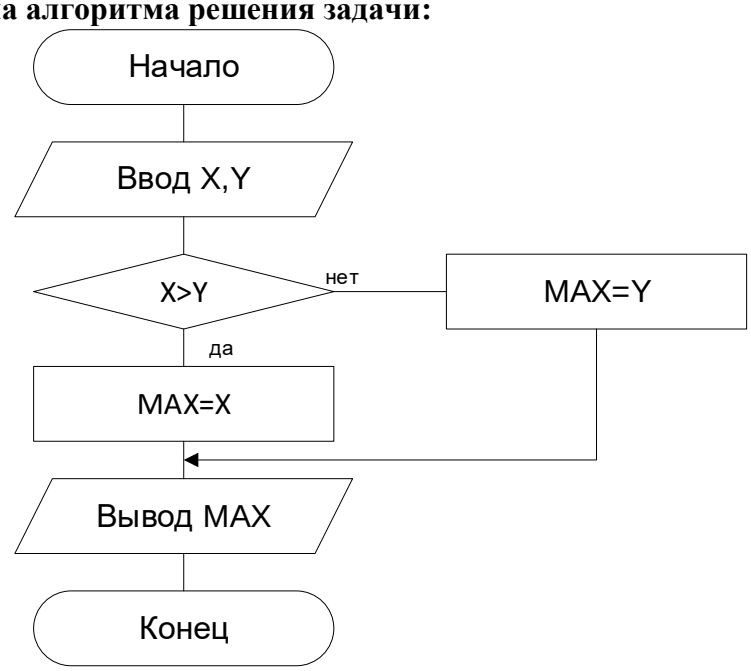
Сравнить два числа и сделать вывод.

Решение.

Исходные данные: X и Y.

Результат: Max.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример2. Найти наибольшее из двух чисел x и y

int x, // Значение x
    y, // Значение н
    Max;// Максимальное
Console.Write("Введите X=");//Подсказка о вводе
x = Convert.ToInt32(Console.ReadLine());//Ввод X
Console.Write("Введите Y=");//Подсказка о вводе
y = Convert.ToInt32(Console.ReadLine());//Ввод Y
if (x > y)
{
    Max = x;
}
else
{
    Max = y;
}
Console.WriteLine("Максимальное число= " + Max);//Вывод ответа
Console.ReadKey();//Пауза
```

Конструкция if – else if – else if – … – else

Форма записи конструкции **if – else if – else if – … – else.** Эта конструкция является вложенным условным оператором, в котором **Действие** 2 является в свою очередь условным оператором:

if (условное выражение 1) Действие 1**;**
else if (условное выражение2) Действие 2**;**
else if (условное выражение 3) Действие 3**;**
…
else Действие N**;**

Приведенная конструкция используется для выбора возможных ситуаций, когда проверяются **условное выражение 1**, **условное выражение 2**, **условное выражение 3**, … . Соответственно будут выполняться действия **Действие** 1, **Действие** 2, **Действие** 3 и т. д. Если ни одно из условий не является истинным, происходит переход к действиям, прописанным после оператора **else**.

Пример 3.

Формулировка задачи:

Вычислить

$$y = \begin{cases} x + 3, \text{если } x < 0 \\ x^2, \text{если } 0 \le x \le 4 \\ e^{x-6}, \text{если } x > 4 \end{cases}$$

Выбор метода решения:

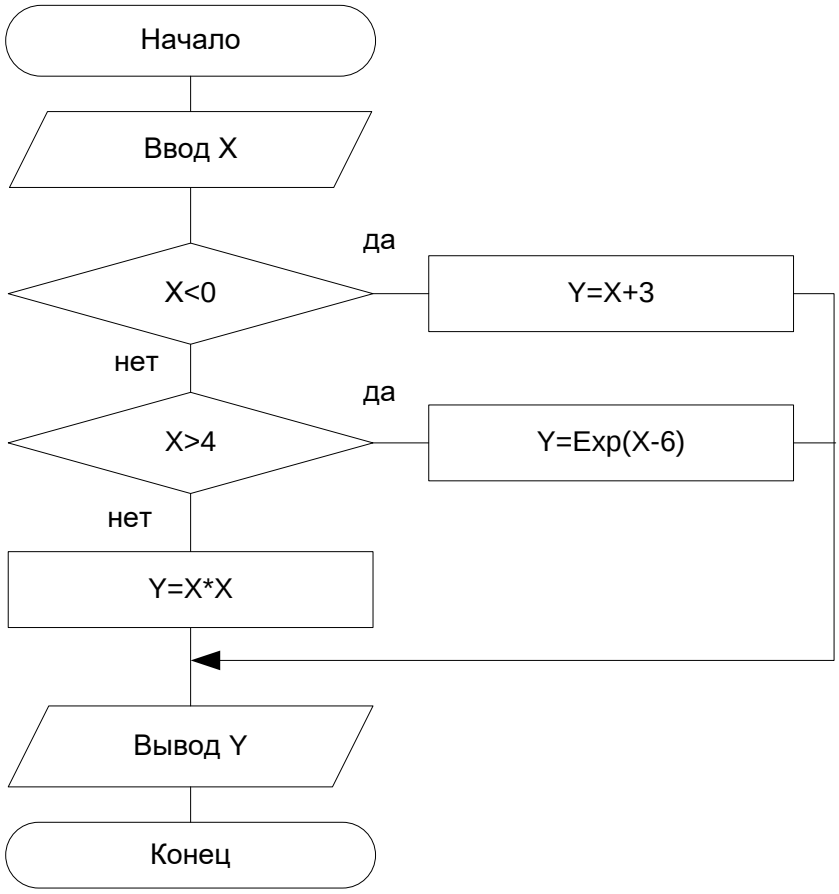
Метод решения прост, если значение ячейки X меньше 0, то рассчитываем значение У по первой формуле, иначе если значение ячейки X больше 4, то рассчитываем значение У по третей формуле, иначе автоматически получается что значение X больше 0 и меньше 4, рассчитываем по второй формуле.

Решение:

Исходные данные: X.

Результат: Y.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример3. Вычислить выражение по формуле

int x; //Исходное число
double y; //Результат
Console.Write("Введите X= ");//Подсказка о вводе
x = Convert.ToInt32(Console.ReadLine());//Ввод X
if (x < 0)
{
    y = x + 3;
}
else if (x > 4)
    {
        y = Math.Exp(x - 6);
    }
    else
    {
        y = x * x;
    }
Console.WriteLine("Ответ Y= " + y);//Вывод ответа
Console.ReadKey();//Пауза
```

Замечание: В случае сложности использования вложенных условий, можно для решения предыдущей задачи использовать простые условия.

```
13      if (X < 0) Y = X + 3;
14      if (X > 4) Y = exp(X - 6);
15      if (X >= 0 && X <= 4) Y = X * X;
```

Таблица 3. Типовые условия

Число положительное	If (x > 0) действие;
Число отрицательное	If (x < 0) действие;
Число четное (на 2 делится без остатка)	If (x % 2 == 0) действие;
Число не четное (на 2 делится с остатком)	If (x % 2 != 0) действие;

Практическая работа №3.

Оператор условия (Оператор IF)

Задание. Составьте алгоритм и программу выполнения варианта задания с использованием оператора **IF** и вывода исходных данных и результатов работы программы. Выполните программу с несколькими наборами исходных данных, так чтобы продемонстрировать работу каждой из ветвей программы.

Варианты заданий (Базовый уровень)

1. Вычислить значение функции y по одной из следующих формул: $y=x^2+3$, если $x<0$; $y=(x^2+3)2$, если $0\le x<1$; $y=x(x^2+3)$, если $x\ge1$.

2. Даны три числа a, b, c. Если хотя бы одно из них равно нулю, то вычислить сумму этих чисел, если все равны нулю, то вывести сообщение об этом, в противном случае вычислить их произведение.

3. Даны три значения a, b, c. Найти значение, наиболее близкое к среднему арифметическому (a+b+c)/3, вывести это значение и его имя.

4. Даны два произвольных числа a, b. Если оба значения принадлежат отрезку [c, d], то вычислить R=sin(a)·sin(b), в противном случае вывести сообщение «Нет решения».

5. В зависимости от параметров a, b, c квадратного уравнения вывести одно из сообщений: «корней нет», «корни действительные и различные», «корни действительные и равные».

6. Даны три числа a, b, c. Вычислить экспоненту (ex)того числа, значение которого ближе всего к значению функции $y=(a^2+b^2+c^2)/(a+b+c)$. Если a+b+c=0, то вывести сообщение об этом.

7. Даны числа a, b, c, d. Определить возможность построения параллелограмма из отрезков с такими значениями.

8. Определить, в каком квадранте координатной плоскости лежит точка с координатами (x, y).

9. Даны три числа a, b, c. Определить количество отрицательных и количество положительных чисел.

10. Даны числа a и b. Если оба значения положительны, то каждое из них удвоить; если оба отрицательны, то отбросить их знаки; в противном случае оставить числа без изменения.

11. Вывести заданные числа a, b, c в порядке убывания их значений.

12. Даны значения a, b, c. Если ни одно из значений не равно нулю, то вычислить D=1/a+1/b+1/c, и если D>0 и a>0, вычислить $y=\sqrt{D}$, в противном случае y=2. Если хотя бы одно из значений a, b, c равно нулю, то вывести сообщение «Нет решения».

13. Даны значения a, b, c. Если a>b и b>c, то вычислить x=0,2; $y=x^2+0,6x+$ +sin(x/2), если b<c и a<c, то x=2; $y=x^3+x^2+x+1$, в остальных случаях x=0, y=0.

14. Определить, лежит ли точка с координатами (x1, y1) внутри прямоугольника, ограниченного линиями с уравнениями x=a, x=b, y=c, y=d.

15. Даны значения x1, x2, x3. Вычислить значения переменных a1, a2, a3 по следующему правилу: если x1>x2>x3, то a1=x1, a2=x2, a3=x3; если $x1\le x2\le x3$, то a1=x3, a2=x1, a3=x2, в остальных случаях вывести только значения x1, x2, x3.

16. Преобразовать заданные значения x и y: если x и y отрицательны, то каждое значение заменить его модулем. Если отрицательно только одно из них, оба значения увеличить на 1, если оба значения неотрицательны, то присвоить им значение, равное наибольшему из них.

Варианты заданий (Повышенный уровень)

1. Определить, принадлежит ли точка с координатами (x, y) области, ограниченной равнобедренным прямоугольным треугольником с катетами длиной a и прямым углом в начале координат (1-й квадрант).

2. Даны три числа a, b, c, которые представляют собой значения сторон треугольника. Определить вид треугольника: прямоугольный, остроугольный, тупоугольный.

3. Даны шесть монет, из которых одна фальшивая. Фальшивая монета отличается от настоящей только весом, причем она либо немного тяжелее, либо немного легче. Имея рычажные весы, определить ложную монету не более чем за три взвешивания.

4. Определить, лежит ли точка с координатами (x, y) внутри кольца с радиусами r1, r2 и центром в точке (xc; yc).

5. Даны: прямоугольное отверстие размером x на y и кирпич с гранями a, b, c. Составить программу для определения возможности прохождения кирпича через заданное отверстие.

Пример выполнения задания

Формулировка задачи:

Вычислить значение функции z по формуле a–b–c, если a максимально, b–a–c, если b максимально, c–a–b, если c максимально.

Методика решения задачи:

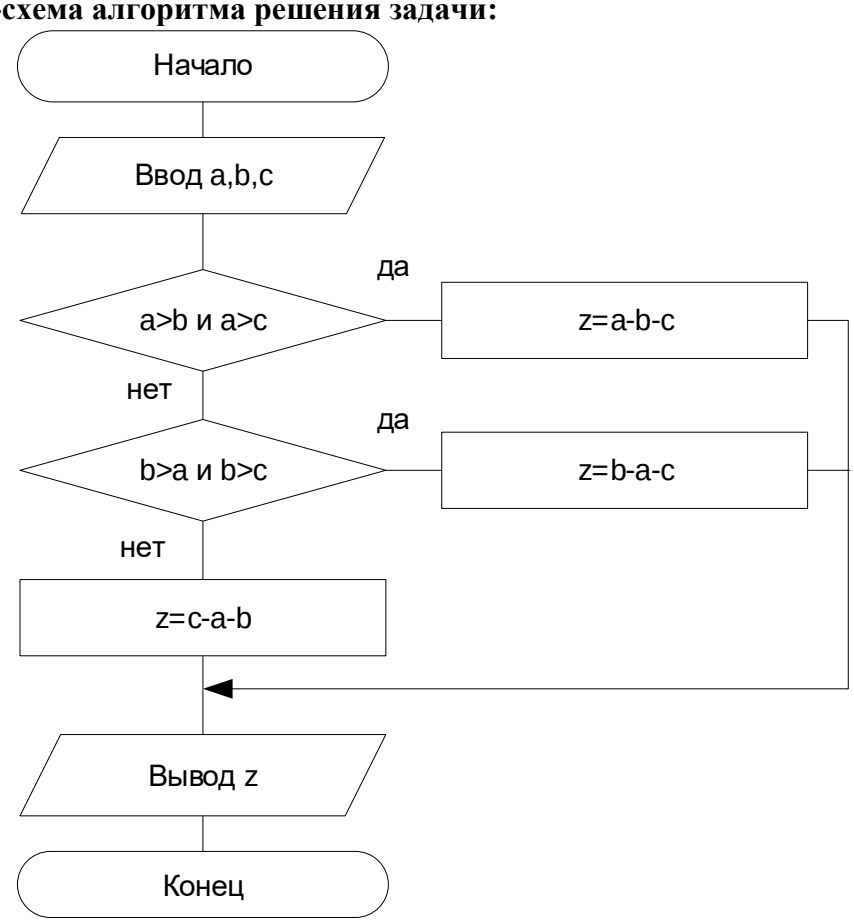
Сравниваем значения a, b, c, находим из них наибольшее и производим расчет по соответствующей формуле. Сравнивать можно сл. образом: если a>b и a>c следовательно значение a наибольшее.

Решение:

Исходные данные: a, b, c – исходные числа.

Результат: z – значение функции.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №3 – оператор условия IF
//Вычислить значение функции z по формуле a–b–c, если a максимально,
//b–a–c, если b максимально, c–a–b, если c максимально
//Выполнил Иванов Сергей гр. ИСП–21 31.02.2022

int a, b, c, //Исходные значения
    z; //Результат
Console.WriteLine("Введите знвчения a, b, c");//Подсказка о вводе
a = Convert.ToInt32(Console.ReadLine());//Ввод
b = Convert.ToInt32(Console.ReadLine());
c = Convert.ToInt32(Console.ReadLine());
if (a > b && a > c)
{
    z = a - b - c;
}
else if (b > a && b > c)
    {
        z = b - a - c;
    }
    else
    {
        z = c - a - b;
    }
Console.WriteLine("Ответ Z=" + z);//Вывод ответа
Console.ReadKey();//Пауза
```

Оператор выбора (оператор варианта)

Условный оператор IF при выполнении программы позволяет выбрать как правило, одно из двух возможных действий, при этом проверяется какое-либо условие и сравниваются некоторые значения. Если же необходимо выбрать из множества действий и выбор осуществляется по простому критерию, например по значению какого-либо выражения (номеру выбора), то удобнее воспользоваться оператором выбора (варианта).

Оператор switch

Общая форма записи оператора **switch**:

```
switch (выражение) {
case значение1:
Действие;
…
break;
case значение2:
Действие;
…
break;
case значениеN:
Действие;
…
break;

default:
Действие;
…
break;
}
```

Выражение, заключенное в круглые скобки оператора, последовательно сравнивается со значениями значение**1**, **значение2**, …, **значениеN**, которые должны быть простыми константами или константными выражениями. В том случае, когда одно из этих значений (например, **значение1**) равно **выражению**, выполняются действия, которые непосредственно следуют за ним.

Оператор **break** сигнализирует об окончании выполнения действий и приводит к выходу из тела оператора **switch**, ставится в конце каждого варианта выбора. Если этого не сделать, то выполнение последовательности действий перейдет в следующий вариант выбора и будет выполняться до тех пор, пока не встретится **break**.

Специальный дополнительный вариант **default** будет выполнен в том случае, когда не будет найдено ни одного совпадения. Ветвь **default** может и отсутствовать.

Пример 4.

Формулировка задачи:

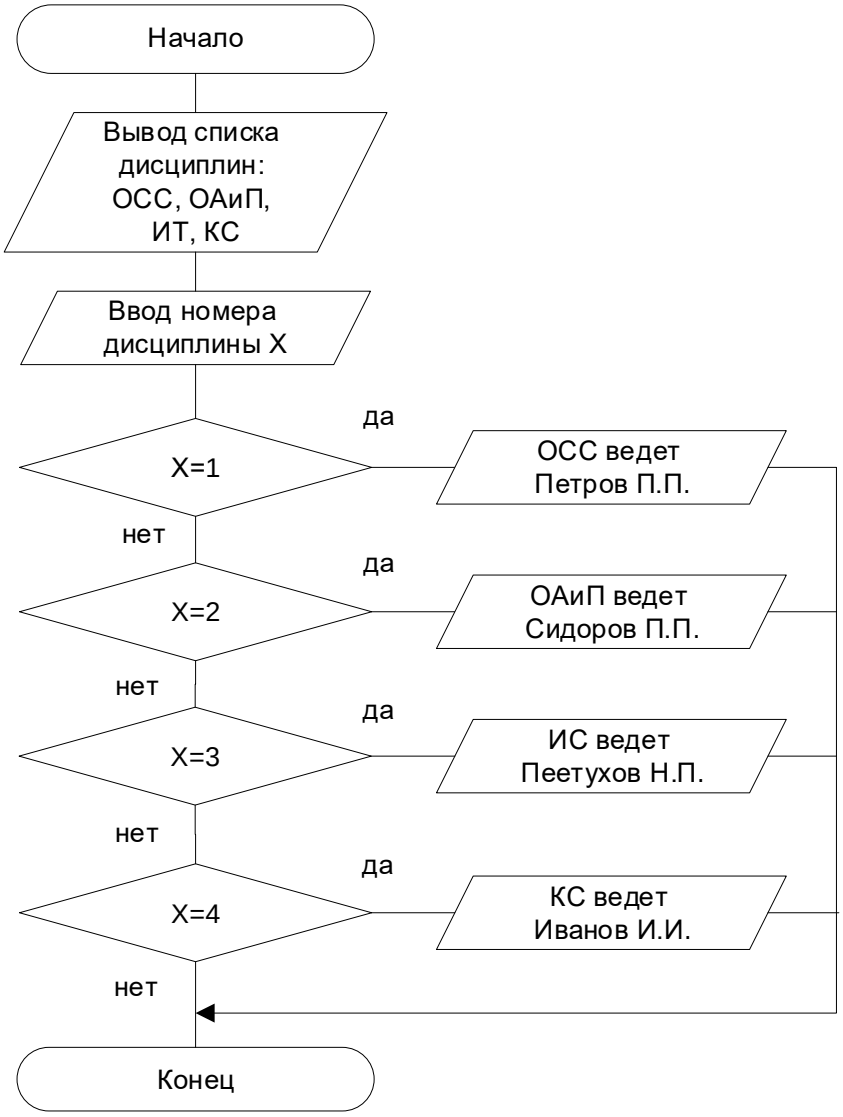
Вывести на экран названия дисциплин 1.ОСС, 2.ОАиП, 3.Информационные технологии, 4.Компьютерные сети. Предоставьте пользователю возможность выбора предмета (ввод номера) и выведите ФИО преподавателя, которые ведут выбранную дисциплину.

Выбор метода решения:

В данной задаче критерий выбора простой – это номер предмета и по номеру нужно вывести фамилию соответствующего преподавателя, поэтому используем оператор **switch**. В качестве селектора будет номер дисциплины.

Решение:
Исходные данные: номер дисциплины X.
Результат: ФИО преподавателя ведущего эту дисциплину.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример4. Указать преподаателя выбранной дисциплины

int value;//Номер выбранной дисциплины
Console.WriteLine("Дисциплины\n" + "1.ОСС\n" + "2.ОАиП\n"
+ "3.Информационные технологии\n" + "4.Компьютерные сети");
Console.Write("Введите номер дисциплины-");
value = Convert.ToInt32(Console.ReadLine());//Ввод номера дисциплины
switch (value)
{
    case 1:
        Console.WriteLine("ОСС ведет Петров П.П.");
        break;
    case 2:
        Console.WriteLine("ОАиП ведет Сидоров П.П.");
        break;
    case 3:
        Console.WriteLine("Информационные технологии ведет Петухов Н.П.");
        break;
    case 4:
        Console.WriteLine("Компьютерные сети ведет Иванов И.И.");
        break;
}
Console.ReadKey();//Пауза
```

Пример 5.

Формулировка задачи:

В качестве выражения в операторе Switch можно использовать любые выражения с перечисляемыми типами данных. В следующем примере используем строковый тип данных.

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример5. Демонстрация оператора Switch

string name = "Alex";

switch (name)
{
    case "Bob":
        Console.WriteLine("Ваше имя - Bob");
        break;
    case "Tom":
        Console.WriteLine("Ваше имя - Tom");
        break;
    case "Sam":
        Console.WriteLine("Ваше имя - Sam");
        break;
    default:
        Console.WriteLine("Неизвестное имя");
        break;
}
Console.ReadKey();//Пауза
```

Практическая работа №4.

Оператор выбора (оператор SWITCH)

Задание. Составьте алгоритм и программу выполнения варианта задания с использованием оператора **SWITCH** и вывода исходных данных и результатов работы программы. Выполните программу с несколькими наборами исходных данных, так чтобы продемонстрировать работу каждой из ветвей программы.

Варианты заданий (Базовый уровень)

1. Выведите на экран информацию:

ФИГУРА:

1. Треугольник 2. Квадрат 3. Прямоугольник
4. Трапеция 5. Круг 6. Кольцо

Предоставьте пользователю возможность выбора варианта (ввода номера фигуры), организуйте ввод нужных данных для вычисления площади фигуры и вычислите площадь выбранной фигуры [$S_1=ah/2$; $S_2=a^2$; $S_3=a\cdot b$; $S_4=(a+b)h/2$; $S_5=\pi\cdot r^2$; $S_6=\pi(r_2^2-r_1^2)$].

2. Введите три целых числа day, mount, year, соответствующие определенной дате. Выведите название месяца и количество дней месяца, учитывая особенности високосного года.

3. Для натурального числа k<30 напишите фразу «Мы нашли k грибов в лесу», согласовав окончание слова «гриб» с числом k.

4. Составьте программу, которая в зависимости от введенного числа n≤7, выводит на экран день недели и сообщение, рабочий день или выходной.

5. Введите произвольные числа – n (целое), a и b. В зависимости от остатка k, получаемого в результате деления n на 6, вычислите значение переменной y по одной из следующих формул: y=1, если k=0; y=a+b, если k=1; $y=a^2+b^2$, если k=2 или 3; y=a·sin(b), если k=4; y=a·ln(|b|), если k=5.

6. Выведите на экран следующую информацию:

ГОРОД

1. Москва 2. Каунас 3. Киев 4. Харьков
5. Брест 6. Вильнюс 7. Рязань 8. Минск

Предоставьте пользователю возможность выбора города и определите страну, в которой находится город.

7. В японском календаре годы носят названия животных и составляют 12-летний цикл (крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, курицы, собаки и свиньи). 1984 год был началом очередного цикла. Составьте программу, которая по номеру введенного года выводит его название по японскому календарю.

8. Введите три целых числа day, mount, year. Если тройка чисел образует правильную дату, выведите ее на экран, где название месяца выражено текстом. В противном случае выдайте сообщение об ошибке.

9. Введите целое число n≤100 и два произвольных числа x и y. В зависимости от значения k младшей цифры числа n вычислите значение переменной z по одной из следующих формул: z=1 при k=0; z=x+y при k=1, 7; $z=x^2\cdot\exp(y)$ при k=2, 5; z=x·y при k=8, 9; z=x·sin(y) при k=3, 4; z=x·ln(|y|) при k= 6.

10. Выведите на экран следующую информацию:

СТРАНЫ:

1. Австрия 2. Болгария 3. Греция 4. Италия
5. Норвегия 6. Польша 7. Франция 8. Испания

Предоставьте пользователю возможность выбора страны (ввод номера) и выведите ее столицу.

11. Для произвольного целого числа k<=30 напишите фразу «У меня в кармане k рублей», согласовав окончание слова «рубль» с числом k.

12. Введите два произвольных числа a и b (оба <10). В зависимости от результата выполнения операции k=(a·b)/(a+b), вычислите значение переменной y по одной из следующих формул: y=1/(a+b) при k=0; y=(a+b)/b^2 при k=1,2; y=a^2+b^2 при k=3; y=b·cos(a) при к=4; y=a·sin(b) при k=5; y=0 в остальных случаях.

13. Выведите на экран следующую информацию:

СТРАНА

1. Австрия 2. Гана 3. Дания 4. Египет
5. Италия 6. Перу 7. США 8. Швеция

Предоставьте пользователю возможность выбора страны и определите континент, на котором находится страна.

14. Для целого числа k от 1 до 99 напечатайте фразу «Мне k лет», учитывая, что при некоторых значениях k слово «лет», надо заменить словом «год».

15. Для натурального числа k<30 напишите фразу «Мы отъехали k километров от дома», согласовав окончание слова «километр» с числом k.

16. Выведите на экран следующую информацию:

МАРКА АВТОМОБИЛЯ

1. Пежо 2. Лада 3. Рено 4. Нива
5. Форд 6. Вольво 7. БМВ 8. Мерседес

Введите номер марки и определите марку и страну-производителя.

17. Дано целое число K. Вывести строку-описание оценки, соответствующей числу K (1 — «плохо», 2 — «неудовлетворительно», 3 — «удовлетворительно», 4 — «хорошо», 5 — «отлично»). Если K не лежит в диапазоне 1–5, то вывести строку «ошибка».

18. Дан номер месяца — целое число в диапазоне 1–12 (1 — январь, 2 — февраль и т. д.). Вывести название соответствующего времени года («зима», «весна», «лето», «осень»).

Пример выполнения задания

Формулировка задачи:

Введите целое число n и два произвольных числа x и y. В зависимости от значения k остатка от деления n на 5 вычислите значение переменной z по формуле: z=x+y при нечетных k, $z=x^2+y^2$ при четных k, в противном случае z=0.

Методика решения задачи:

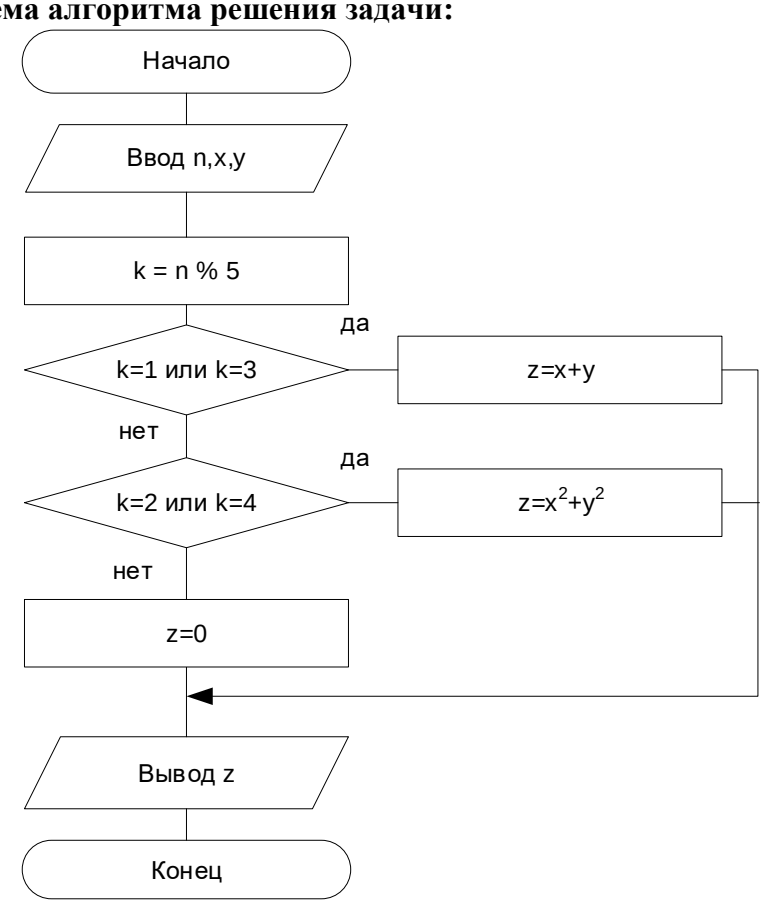
Число n делим на 5 и получаем остаток от деления k=n % 5. Остатком от деления в данном случае будут числа 1, 2, 3, 4 или 0. Для чисел 1, 3 расчет выполняется по формуле z=x+y. Для чисел 2, 4 расчет выполняется по формуле $z= x^2+y^2$. В данной задаче критерий выбора простой – это остаток целое число, по которому выбирается формула расчета, поэтому используем оператор SWITCH. В качестве селектора будет остаток k.

Решение:

Исходные данные: n, x, y – исходные числа.

Результат: z – значение функции.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №4 – оператор выбора SWITCH
//Введите целое число n и два произвольных числа x и y.
//В зависимости от значения k остатка от деления n на 5
//вычислите значение переменной z по формуле: z=x+y при нечетных k,
//z=x2+y2 при четных k, в противном случае z=0.
//Выполнил Иванов Сергей гр. ИСП-21 31.03.2020

int x, y, n, //Исходные значения
    z; //Результат
Console.WriteLine("Введите x, y, n");
x = Convert.ToInt32(Console.ReadLine());//Ввод
y = Convert.ToInt32(Console.ReadLine());
n = Convert.ToInt32(Console.ReadLine());
switch (n % 5)
{
    case 1:
    case 3:
        z = x + y;
        break;
    case 2:
    case 4:
        z = x * x + y * y;
        break;
    default:
        z = 0;
        break;
} // End switch
Console.WriteLine("Результат Z=" + z);
Console.ReadKey();//Пауза
```

Программирование алгоритмов циклической структуры

Вспомним, вначале, что циклический алгоритм реализует повторение некоторых действий. Иными словами, **ЦИКЛИЧЕСКИЕ** алгоритмы включают в себя циклы.

Цикл - это последовательность действий, выполняемых многократно, каждый раз при новых значениях параметров. Группа действий, повторяемая в цикле, называется **тело цикла**. Однократное выполнение тела цикла называется **шагом**.

Виды циклических алгоритмов:

- ***арифметический***, т.е. в таком цикле мы точно знаем сколько раз, цикл повторится;
- ***итерационный***, в таком цикле есть только одна причина его окончания;
- ***поисковый***, в таком цикле есть несколько причин его окончания.

В языке C# имеются четыре различных оператора, с помощью которых можно запрограммировать повторяющиеся фрагменты программы.

Оператор цикла с параметром - for

Это самый популярный оператор цикла для всех с-подобных языков. Популярность объясняется необычайной гибкостью и мощью этого оператора. Обычно этот цикл работает на базе счетчика и выполняет указанное количество повторений.

Общий вид оператора **for** следующий:
for (*начальные действия*; *условие*; *дополнительные действия*)
Оператор тела цикла

или так
for (*начальные действия*; *условие*; *дополнительные действия*)
{
  Операторы тела цикла
}

Начальные действия — это любые операторы присваивания, записанные через запятую. Здесь, как правило, делается инициализация различных переменных, необходимых для работы в цикле **for**.

Условие — логическое выражение, записывается так же, как и для других операторов цикла и оператора **if**.

Дополнительные действия — любые записанные через запятую операторы присваивания и (или) операторы-выражения (автоувеличение, автоуменьшение).

Алгоритм работы оператора for:

1. Вычисляются операторы, составляющие ***начальные действия***.
2. Вычисляется ***условие***. Если оно ложно, то оператор заканчивает свою работу. Если ***условие*** истинно, то выполняются ***операторы тела цикла***, потом — операторы, составляющие группу ***дополнительных действий***, и затем снова делается проверка истинности ***условия***. И так до тех пор, пока оно остаётся истинным.

Обратите внимание, что количество операторов в ***начальных*** и в ***дополнительных действиях*** может быть любым. Любая часть в заголовке (или даже все части) могут отсутствовать:

for(; ;) { *операторы тела цикла* }

Но точки с запятыми обязательно должны быть!

Чаще всего, несмотря на свою гибкость, оператор ***for*** используется в следующем классическом виде:

for (int i = 1; i <= 15; i++) тело цикла;

В данном случае цикл повторится 15 раз. Переменная i - это индекс цикла, которому присваиваем начальное значение 1. Пока **i <= 15** тело цикла будет повторяться. На каждом шаге увеличиваем значение **i** на единичку **i++.**

Пример 6.

Формулировка задачи:

Составить программу, которая определяет, является ли четырёхзначное число "перевёртышем", т.е. одинаково читается слева направо и наоборот.

Методика решения задачи

Перевернем число и сравним его с исходным числом. Для этого от исходного значения будет отрывать числа с конца, и вставлять их в конец перевернутого числа.

Получить число с конца исходного значения можно путем получения остатка от деления на 10. Например: от 456 отделим последнее число - 456 % 10=6, т.е. получили число с конца текущего значения.

Удалить число с конца исходного значения можно путем получения целой части от деления на 10. Например: от 456 удалим последнее число - 456 / 10=45, т.е. удалили число с конца текущего значения.

Добавить число в конец текущего значения можно переместив все числа на разряд влево и добавив это число в конец. Например: к 45 в конец добавить 3 – 45*10+3=453.

Рассмотрим решение в развернутом виде. Для примера возьмем значение Isx=4566. Обозначим Isx - вводимое число, Dub - дубликат числа Isx, Per - "перевёртыш" числа Isx,

Шаг1.

n=Isx % 10; (4566 % 10=6) //Получаем последнюю цифру от исходного числа

Per=n; (=6) //Записываем полученную цифру в конец перевертыша

Isx=Isx / 10; (4566 / 10=456) //Отбрасываем от исходного числа последнюю цифру

Шаг2.

n=Isx % 10; (456 % 10=6) //Получаем последнюю цифру от исходного числа

Per=Per*10+n; (6*10+6=66) //Записываем полученную цифру в конец перевертыша

Isx=Isx / 10; (456 / 10=45) //Отбрасываем от исходного числа последнюю цифру

Шаг3.

n=Isx % 10; (45 % 10=5) //Получаем последнюю цифру от исходного числа

Per=Per*10+n;(66*10+5=665)//Записываем полученную цифру в конец перевертыша

Isx=Isx / 10; (45 / 10=4) //Отбрасываем от исходного числа последнюю цифру

Шаг4.

n=Isx % 10; (4 % 10 =4) //Получаем последнюю цифру от исходного числа

Per=Per*10+n;(665*10+4=6654)//Записываем полученную цифру в конец перевертыша

Isx=Isx / 10; (4 / 10=0) //Отбрасываем от исходного числа последнюю цифру

Как видим все шаги одинаковые, кроме первого шага. Первый шаг можно изменить на эквивалентную запись Per=n -> Per=Per*10+n, предварительно присвоив Per:=0. Т.к. теперь все шаги одинаковые, то для их выполнения можно использовать цикл с параметром. Реализацию программы смотри далее.

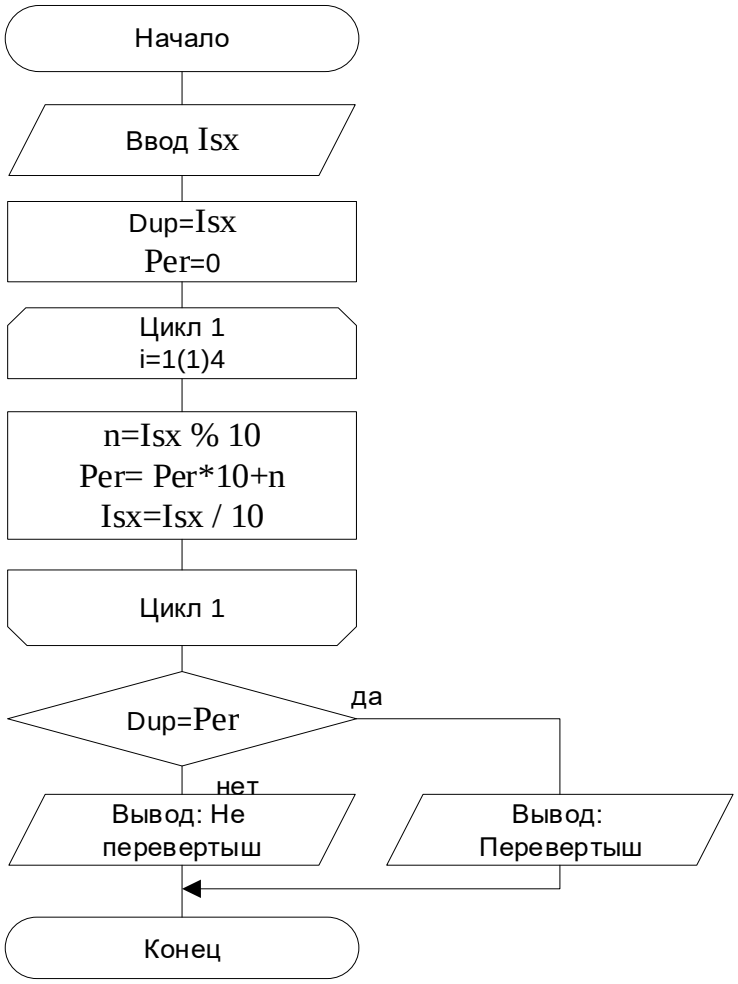
Решение:

Исходные данные: Isx – исходное число.

Промежуточный результат: Per – перевернутое число.

Результат: Число является или не является перевертышем.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример5. Определить, является ли четырёхзначное число "перевёртышем"

int isx, //Исходное число
per, //Перевернутое число
dup, //Резерв исходного числа
n, //Текущая цифра
i; //Счетчик
Console.WriteLine("Введите четырехзначное целое число");
isx = Convert.ToInt32(Console.ReadLine()); //Вводим исх. число
per = 0; dup = isx;
//В цикле переворачиваем число
for (i = 1; i <= 4; i++)
{
    n = isx % 10;   //Получаем последнюю цифру от исходного числа
    per = per * 10 + n; //Записываем полученную цифру в конец перевертыша
    isx = isx / 10; //Отбрасываем от исходного числа использованную цифру
}
//Проверяем на совпадение исх. и перевернутого числа
if (dup == per) Console.WriteLine("Число " + dup + " перевертыш");
else Console.WriteLine("Число {0} не перевертыш", dup);
Console.ReadKey();//Пауза
```

Пример 7

Формулировка задачи:

Составьте алгоритм и программу вычисления значения функции двумя способами по ее аналитической формуле и с помощью эквивалентной суммы или произведения с использованием оператора **FOR**. Вывести для сравнения оба рассчитанных значения.

	$\frac{1}{2}$	$\prod_{n=2}^{100}\left(1-\frac{1}{n^2}\right)$

Методика решения задачи

Вычисляем значение функции по аналитической формуле ½.

Вычисляем значение функции по формуле $S=\prod_{n=2}^{100}\left(1-\frac{1}{n^2}\right)$ и сравниваем, при правильном расчете эти значения близки по значению.

Подставляем значение n и вычисляем произведение числового ряда. $S=(1-1/2^2)*(1-1/3^2)*(1-1/4^2)*\ldots*(1-1/100^2)$. 100 раз мы используем одну и ту же формулу, в которой меняется только значение n, причем увеличиваясь, каждый раз на 1. Соответственно для реализации расчета произведения можно использовать цикл с параметром, где в качестве параметра взято значение n. Для расчета произведения используем формулу с накоплением ***S=S*(1-1/n²).***

Проверим по шагам алгоритм и правильность расчета формулы ***S=S*(1-1/n²).***

ш0 Первоначально определим S=1

ш1 $S=1*(1-1/2^2)$

ш2 $S=1*(1-1/2^2)*(1-1/3^2)$

ш3 $S=1*(1-1/2^2)*(1-1/3^2)*(1-1/4^2)$

…

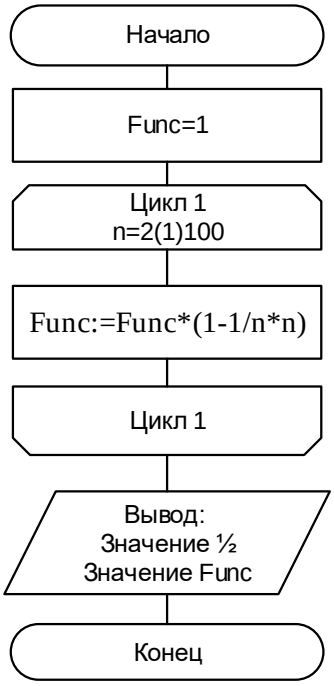
Ш99 $S=1*(1-1/2^2)*(1-1/3^2)*(1-1/4^2)*\ldots*(1-1/100^2)$

Решение:

Исходные данные:

Результат: calcFunc – расчетное значение функции, anltFunc = ½ - аналитическое значение функции.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример6. Подсчитать значение функции 2 способами

double calcFunc, //Расчетная функции
anltFunc;   //Аналитическая функция
int n;      //Счетчик
calcFunc = 1; anltFunc = 1.0 / 2;
//В цикле ищем произведение
for (n = 2; n <= 100; n++)
{
    calcFunc = calcFunc * (1 - 1.0 / (n * n));
}
Console.WriteLine("Результат функции= " + calcFunc);
Console.WriteLine("Аналитическаят функция= " + anltFunc);
//Проверяем на совпадение
if (Math.Abs(calcFunc - anltFunc) < 0.1)
{
    Console.WriteLine("Расчет произведен правильно");
}
else
{
    Console.WriteLine("Расчет произведен неправильно");
}
Console.ReadKey();//Пауза
```

Практическая работа №5.

Цикл с параметром (оператор FOR)

Задание. Составьте алгоритм и программу вычисления значения функции двумя способами по ее аналитической формуле и с помощью эквивалентной суммы или произведения с использованием оператора **FOR**. Значение аргумента (если он есть) выберите с учетом ограничения, приведенного в столбце «Ограничения аргумента», при этом обеспечьте расчет значения только при корректном вводе исходных данных. Вывести для сравнения оба рассчитанных значения. Для обозначения π используйте константы.

№	Аналитическая функция	Эквивалентная сумма или произведение	Ограничения аргумента
1	$\ln\lvert\sin(x)\rvert$	$-\ln 2-\sum_{n=1}^{100}\frac{\cos 2nx}{n}$	$\lvert x\rvert<\pi,\ x\neq\pi/2$
2	$\frac{\pi-x}{2}$	$\sum_{n=1}^{100}\frac{\sin nx}{n}$	$0<x<2\pi$
3	$\sin(x)$	$x\prod_{n=1}^{50}\left(1-\frac{x^2}{n^2\pi^2}\right)$	$\lvert x\rvert<1$
4	$\frac{\pi}{4}\sin x$	$\frac{1}{2}-\sum_{n=1}^{50}\frac{\cos 2nx}{(2n-1)(2n+1)}$	$\lvert x\rvert\le\pi/2$
5	$\frac{\pi}{2}$	$\prod_{n=1}^{50}\frac{4n^2}{4n^2-1}$	–
6	$\frac{1}{1-x}$	$\prod_{n=0}^{10}\left(1+x^{2^n}\right)$	$0\le x<1$
7	$\frac{1}{2}-\frac{\pi}{4}\lvert\sin x\rvert$	$\sum_{n=1}^{50}\frac{\cos 2nx}{4n^2-1}$	$\lvert x\rvert<1$
8	1	$\sum_{n=1}^{100}\frac{1}{n(n+1)}$	–
9	$\frac{\pi\cos 1}{8}$	$\sum_{n=1}^{50}\frac{n\sin 2n}{(2n-1)(2n+1)}$	–
10	$\frac{\pi^4}{90}$	$\sum_{n=1}^{12}\frac{1}{n^4}$	–
11	$\frac{3}{4}$	$\sum_{n=2}^{100}\frac{1}{(n-1)(n+1)}$	–
12	$\frac{\pi}{4}$	$\sum_{n=0}^{100}\frac{\sin((2n+1)x)}{(2n+1)}$	$0<x<\pi$
13	$\frac{\pi}{4}\left(\frac{\pi}{2}-x\right)$	$\sum_{n=0}^{100}\frac{\cos((2n+1)x)}{(2n+1)^2}$	$0\le x\le\pi$
14	$\cos(x)$	$\prod_{n=1}^{50}\left(1-\frac{4x^2}{(2n-1)^2\pi^2}\right)$	$\lvert x\rvert<1$
15	$\frac{\pi}{8}\cos x$	$\sum_{n=1}^{50}\frac{n\sin 2nx}{(2n-1)(2n+1)}$	$0<x<\pi$
16	$\frac{\pi}{4}$	$\prod_{n=1}^{100}\left(1-\frac{1}{(2n+1)^2}\right)$	–
17	$\frac{\pi^2}{6}$	$\sum_{n=1}^{100}\frac{1}{n^2}$	–

18	$\frac{\pi^2}{8}$	$\sum_{n=1}^{50}\frac{1}{(2n-1)^2}$	–
19	$\frac{1}{2}-\frac{\pi}{8}$	$\sum_{n=1}^{25}\frac{1}{(4n-1)(4n+1)}$	–
20	$\frac{2-\pi\sin 1}{4}$	$\sum_{n=1}^{50}\frac{\cos 2n}{(2n-1)(2n+1)}$	–
21	$\frac{1}{4}$	$\sum_{n=1}^{30}\frac{1}{n(n+1)(n+2)}$	–
22	$\frac{\pi^2-6}{12}$	$\sum_{n=1}^{50}\frac{\cos n}{n^2}$	–

Пример выполнения задания

Формулировка задачи:

Составьте алгоритм и программу вычисления значения функции жвумя способами по ее аналитической формуле и с помощью эквивалентной суммы или произведения с использованием оператора **FOR**. Вывести для сравнения оба рассчитанных значения.

	$\frac{1}{4}$	$\sum_{n=1}^{30}\frac{1}{n(n+1)(n+2)}$	–

Методика решения задачи

Вычисляем значение функции по аналитической формуле 1/4.

Вычисляем значение функции по формуле S=$\sum_{n=1}^{30}\frac{1}{n(n+1)(n+2)}$ и сравниваем, при правильном расчете эти значения близки по значению.

Подставляем значение n и вычисляем сумму числового ряда. S=1/(1*(1+1)*(1+2)) + 1/(2*(2+1)*(2+2)) +1/(3*(3+1)*(3+2)) +…+1/(30*(30+1)*(30+2)). 30 раз мы используем одну и ту же формулу, в которой меняется только значение n, причем увеличиваясь, каждый раз на 1. Соответственно для реализации расчета суммы можно использовать цикл с параметром, где в качестве параметра взято значение n. Для расчета суммы используем формулу с накоплением ***S=S+1/(n*(n+1)*(n+2)).***

Проверим по шагам алгоритм и правильность расчета формулы ***S=S+1/(n*(n+1)*(n+2)).***

ш0 Первоначально определим S=0

ш1 S=0+1/(1*(1+1)*(1+2))

ш2 S=0+1/(1*(1+1)*(1+2))+ 1/(2*(2+1)*(2+2))

ш3 S=0+1/(1*(1+1)*(1+2))+ 1/(2*(2+1)*(2+2))+ 1/(3*(3+1)*(3+2))

…

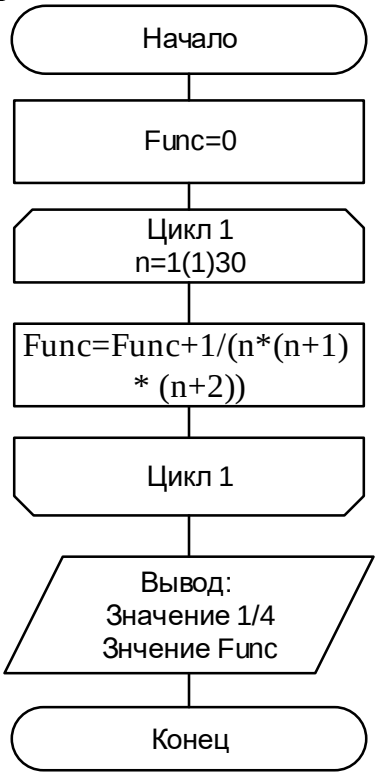
Ш30 S=0+1/(1*(1+1)*(1+2))+ 1/(2*(2+1)*(2+2))+ 1/(3*(3+1)*(3+2)) +…+ 1/(30*(30+1)*(30+2)).

Решение:

Исходные данные:

Результат: calcFunc – расчетное значение функции, anltFunc = 1/4 - аналитическое значение функции.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №5 – оператор цикла For
//Подсчитать значение функции 2 способами
//Выполнил Иванов Сергей гр. ИСП-21 31.03.2022

double calcFunc, //Расчетная функции
anltFunc;   //Аналитическая функция
int n;      //Счетчик
calcFunc = 0; anltFunc = 1.0 / 4;
//В цикле ищем сумму
for (n = 1; n <= 30; n++)
{
    calcFunc = calcFunc + 1.0 / (n * (n + 1) * (n + 2));
}
Console.WriteLine("Результат функции= " + calcFunc);
Console.WriteLine("Аналитическаят функция= " + anltFunc);
//Проверяем на совпадение
if (Math.Abs(calcFunc - anltFunc) < 0.1)
{
    Console.WriteLine("Расчет произведен правильно");
}
else
{
    Console.WriteLine("Расчет произведен неправильно");
}
Console.ReadKey();//Пауза
```

Оператор цикла с предусловием - while

Общий вид оператора **while** следующий:

while (*условие*) // Заголовок

{

операторы тела цикла

}

Слово **while** переводится как «**пока**». То есть, пока истинно некое условие, повторять цикл.

Схематично этот цикл можно изобразить так, как показано на рисунке:

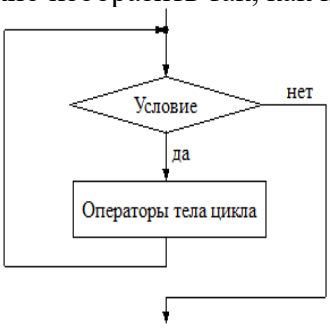


Рисунок 1 - Структура цикла с предусловием

Как видим — это цикл с предусловием. Вначале проверяем истинность некоторого ***условия***, а затем, если оно истинно, выполняем операторы (один или несколько), составляющие тело цикла.

Такой цикл не выполниться ни разу, если условие изначально ложно.

Условие — это любое выражение, имеющее результат логического или арифметического типа, т.е. условие для операторов цикла записывается точно также, как для оператора ветвления **if**.

Тело цикла практически всегда необходимо оформлять как блок, так как оператор **while** редко содержит в своём теле только один оператор.

Алгоритм работы оператора прост:

1. Вычисляется значение условия в заголовке оператора.
2. Если оно истинно, то выполняются операторы тела цикла, а затем управление снова передаётся на заголовок. Если условие ложно, то оператор заканчивает работу. После этого будет выполняться оператор, следующий сразу же за оператором цикла.

Пример 8.

Формулировка задачи:

Дано натуральное число n. Подсчитать количество цифр данного числа.

Методика решения задачи

Отделяем по одной цифре с конца числа, подсчитывая их количество. Так делаем до тех пор, пока цифры не закончатся, т.е. число будет равно нулю. Отделять цифры можно операцией деления без остатка / (Например: 567 / 10=56 и т.д.).

Рассмотрим решение в развернутом виде. Для примера возьмем значение Isx=4566. Обозначим Kol=0 – количество чисел

Шаг1.

Isx=Isx / 10; (4566 / 10=456) //Отбросили последнюю цифру

Kol=Kol+1; (0+1=1) //Счетчик увеличили на 1

Шаг2.

Isx=Isx / 10; (456 / 10=45) //Отбросили последнюю цифру

Kol=Kol+1; (1+1=2) //Счетчик увеличили на 1

Шаг3.

Isx=Isx / 10; (45 / 10=4) //Отбросили последнюю цифру

Kol=Kol+1; (2+1=3) //Счетчик увеличили на 1

Шаг4.

Isx=Isx / 10; (4 / 10=0) //Отбросили последнюю цифру

Kol=Kol+1; (3+1=4) //Счетчик увеличили на 1

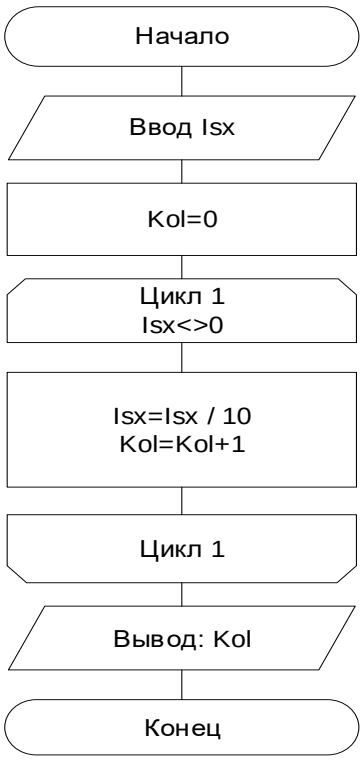
Как видим все шаги одинаковые, то для их выполнения можно использовать цикл. Закончится цикл тогда, когда закончатся цифры в числе, т.е. число будет равно нулю Реализацию программы смотри далее.

Решение.

Исходные данные: Isx – исходное число.

Результат: kol – количество цифр в числе.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример7. Подсчитать количество цифр данного числа

int Kol = 0,    //Счетчик
value, dublicateValue;//Исходное значение
Console.Write("Введите исходное число- ");
value = dublicateValue = Convert.ToInt32(Console.ReadLine());
//В цикле ищем кол-во цифр
while (dublicateValue != 0)
{
    dublicateValue = dublicateValue / 10;
    Kol++;
}
Console.WriteLine("Кол-во цифр в числе - {0} будет {1}", value, Kol);
Console.WriteLine($"Кол-во цифр в числе - {value} будет {Kol}");
Console.ReadKey();//Пауза
```

Практическая работа №6

Цикл с предусловием (оператор WHILE)

Задание. Составьте алгоритм и программу выполнения варианта задания с использованием оператора **WHILE** и вывода исходных данных и результатов работы программы.

Варианты заданий (Базовый уровень)

1. Найти 10 первых натуральных чисел, оканчивающихся на цифру 7, кратных числу 9 и находящихся в интервале, левая граница которого равна 100.

2. Ввести последовательность целых чисел. Вычислять сумму чисел в диапазоне от - 3 до 7, пока эта сумма не превышает некоторого числа K. Вывести на экран сгенерированные числа, значение суммы, и количество сгенерированных чисел.

3. Ввести последовательность целых чисел, Вычислять для чисел > 0 $\sqrt{X}$, а для чисел < 0 функцию x^2. Вычисления прекратить, когда подряд появится два одинаковых числа.

4. Дано число n. Из чисел 1, 4, 9, 16, 25, ... напечатать те, которые не превышают n. Среди чисел 1, 4, 9, 16, 25, ... найти первое число, большее n.

5. Дано число a (1 < a <1,5). Из чисел 1+1/2, 1+1/3…, напечатать те, которые не меньше a. Ограничить числа 2 разрядами после запятой.

6. Дано натуральное число. Определить: а) количество цифр 3 в нем; б) сколько раз в нем встречается последняя цифра; в) количество четных цифр в нем.

7. Дано натуральное число. Определить номер цифры 8 в нем, считая от конца числа. Если такой цифры нет, ответом должно быть число 0, если таких цифр в числе несколько — должен быть определен номер самой левой из них.

8. Дано натуральное число. Определить: а) сколько раз в нем встречается цифра a; б) количество его цифр, кратных z; в) сумму его цифр, больших a.

9. Найти 15 первых натуральных чисел, делящихся нацело на 19 и находящихся в интервале, левая граница которого равна 100.

10. Дано натуральное число. Установить, является ли последовательность его цифр при просмотре их справо налево упорядоченной по убыванию. Например, для чисел 1368 и 1679 ответ положительный, для числа 1782 — отрицательный и т. п.

11. Дано натуральное число. Определить, является ли оно членом последовательности Фибоначчи (первый и второй члены последовательности равны 1, каждый следующий равен сумме двух предыдущих).

12. Выяснить, является ли заданное число n членом арифметической прогрессии, первый член которой равен f, а шаг — s.

13. Дано натуральное число. Определить, какая цифра встречается в нем чаще: 0 или 9.

14. Дано натуральное число. Если в нем есть цифры 2 и 5, то определить, какая из них расположена в числе левее. Если одна или обе эти цифры встречаются в числе несколько раз, то должны быть учтены самые левые из одинаковых цифр.

15. Гражданин 1 марта открыл счет в банке, вложив 1000 руб. Через каждый месяц размер вклада увеличивается на 2% от имеющейся суммы. Определить: а) за какой месяц величина ежемесячного увеличения вклада превысит 30 руб.; б) через сколько месяцев размер вклада превысит 1200 руб.

16. Найти наименьшее общее кратное двух заданных натуральных чисел.

Варианты заданий (Повышенный уровень)

1. Рассмотрим последовательность, образованную дробями: 1/1, 2/1, 3/2, ..., в которой числитель (знаменатель) следующего члена последовательности получается сложением числителей (знаменателей) двух предыдущих членов. Числители двух первых дробей равны 1 и 2, знаменатели — 1 и 1. Найти первый член такой последовательности, который отличается от предыдущего члена не более чем на 0,001.

2. Дано натуральное число. а) Определить его максимальную и минимальную цифры. б) Определить, на сколько его максимальная цифра превышает минимальную.

3. Дано натуральное число, в котором все цифры различны. Определить: а) порядковый номер его максимальной цифры, считая номера: от конца числа; от начала числа.

4. Дано натуральное число. Определить, сколько раз в нем встречается максимальная цифра (например, для числа 132 233 ответ равен 3, для числа 46 336 — 2, для числа 12 345 — 1).

5. Найти минимум из n целых случайных чисел X, распределенных в диапазоне от 10 до 40. Вывести на экран на одной строке сгенерированные числа, на другой строке результат.

Пример выполнения задания

Формулировка задачи:

Определить, является ли заданное число X степенью числа 3.

Методика решения задачи

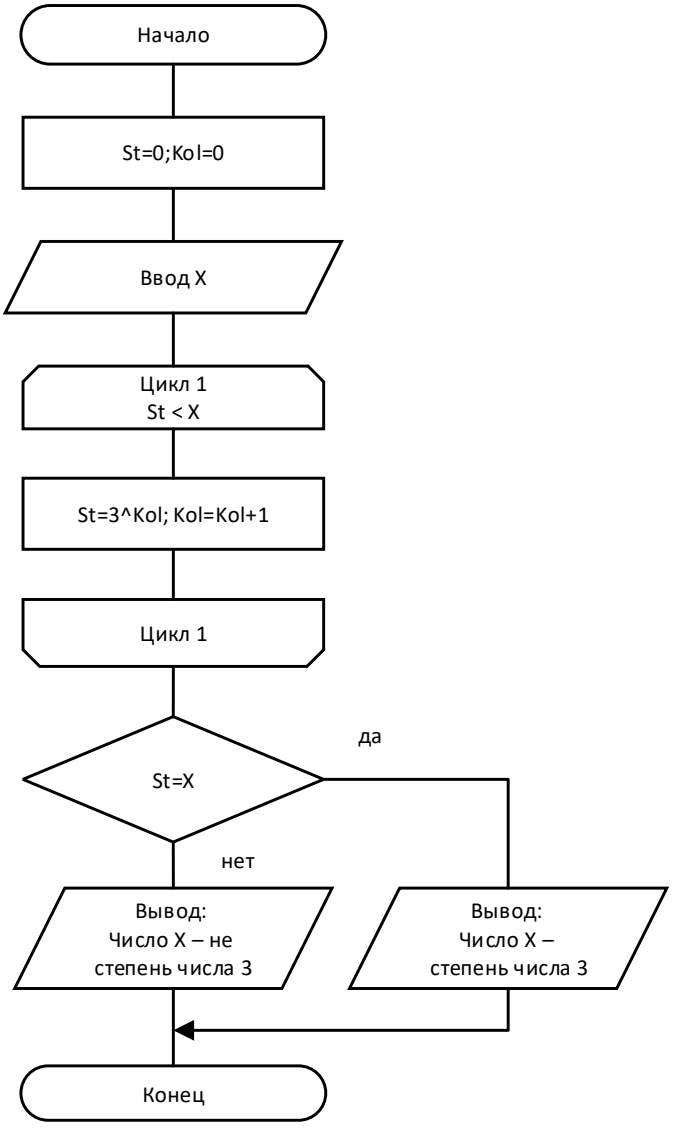
Методика проста, число 3 возводим последовательно в степени 1,2,3 и т.д. и сравниваем с числом X пока степень числа 3 меньше числа X. Если степень числа 3 и число X равны, то ответ утвердительный.

Решение:

Исходные данные: X – число

Результат: является ли заданное число X степенью числа 3.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №6 – оператор цикла While
//Определить, является ли заданное число X степенью числа 3
//Выполнил Иванов Сергей гр. ИСП-21 31.03.2022

int valueX,  //Число
Kol = 0;    //Счетчик Степени
double Step3 = 0;  //Степень числа 3
Console.Write("Введите число X-"); //Приглашение к воду
valueX = Convert.ToInt32(Console.ReadLine()); //Ввох x
while (Step3 < valueX)
{
    Step3 = Math.Pow(3, Kol);
    Kol++;
}
if (Step3 == valueX)
{
    Console.WriteLine("Число X = {0} является степенью числа 3", valueX);
}
else
{
    Console.WriteLine("Число X = {0} не является степенью числа 3", valueX);
}
Console.ReadKey();//Пауза
```

Оператор break и continue

BREAK - реализует немедленный выход из цикла, управление передается оператору, стоящему сразу после тела цикла.

CONTINUE - обеспечивает досрочное прохождение тела цикла, эквивалентна передаче управления в конец циклического оператора.

Оператор цикла с постусловием – do - while

Цикл **do** — это цикл с постусловием.

Общий вид оператора **do - while**:

do{

Операторы тела цикла

} while(*Условие*);

Цикл **do** всегда выполняется хотя бы один раз. И в некоторых задачах это хорошо, а в других нет. Для нашей только что рассмотренной задачи этот цикл может дать побочный эффект: если *число* по ошибке задать равным нулю, то количество будет равно **1**, что явно не так. А вот использовать цикл **do** для проверки входных данных очень даже удобно! Что мы и сделаем, тем самым избавимся от побочного эффекта.

Схематично этот цикл можно изобразить так, как показано на рисунке:

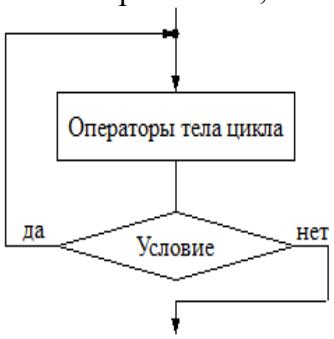


Рисунок 2 - Структура цикла с постусловием

Пример 10.

Формулировка задачи:

Составить программу, в которой пользователь вводит символ с клавиатуры и на экран выводится код этого символа. Для завершения работы программы нужно нажать *.

Методика решения задачи:

Вводим с клавиатуры символы, переводим их в код. Повторяем процесс, пока не встретим код символа *.

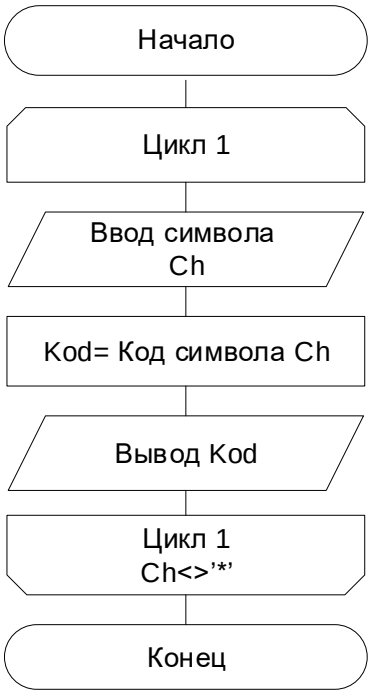
Для решения этой задачи воспользуемся следующими знаниями языка C#. Для хранения символа используем тип Char, для кода тип Int. Переменная типа Char позволяет хранить 1 символ, но по сути реализации в ней вместо символа хранится код этого символа, поэтому если переменную с символом присвоить переменной целого типа, то в нее запишется код этого символа.

Решение:

Исходные данные: Ch – символ.

Результат: Kod – код числа.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример8. Вывод кода введенного символа.* завершений

int kodCh; //Код числа
char ch; //Символ
Console.WriteLine("Введите букву");
do
{
    ch = Convert.ToChar(Console.ReadKey().KeyChar);
    kodCh = ch;//Переводим символ в код
    Console.WriteLine(" - " + kodCh);
} while (ch != '*');
Console.ReadKey();//Пауза
```

Практическая работа №7

Цикл с постусловием (оператор DO - WHILE)

Задание. Составьте алгоритм и программу выполнения варианта задания с использованием оператора **DO - WHILE** и вывода исходных данных и результатов работы программы.

Варианты заданий (Базовый уровень)

1. Дана непустая последовательность целых чисел, оканчивающаяся нулем. Найти: а) сумму всех чисел последовательности; б) количество всех чисел последовательности.

2. Дана непустая последовательность неотрицательных целых чисел, оканчивающаяся отрицательным числом. Найти среднее арифметическое всех чисел последовательности (без учета отрицательного числа).

3. Дана последовательность натуральных чисел, оканчивающаяся нулем. Определить, есть ли в последовательности хотя бы одно число, оканчивающееся цифрой 7? В случае положительного ответа определить порядковый номер последнего из них.

4. Дана непустая последовательность целых чисел, оканчивающаяся -5. Найти: а)количество четных чисел последовательности; б)количество положительных чисел последовательности.

5. Дана непустая последовательность чисел, оканчивающаяся нулем. Последовательность является невозрастающей. Найти количество различных чисел в последовательности.

6. Дана последовательность целых чисел, оканчивающаяся числом -1. Количество чисел в последовательности не меньше двух. Определить, есть ли в ней хотя бы одна пара "соседних" четных чисел. В случае положительного ответа определить порядковые номера чисел последней из таких пар.

7. Дана непустая последовательность чисел, оканчивающаяся нулем. Найти: а) сумму всех чисел последовательности, больших числа x; б) количество всех четных чисел последовательности.

8. Дана последовательность ненулевых целых чисел, оканчивающая нулем. Определить, сколько раз в этой последовательности меняется знак. (Например, в последовательности 10, –4, 12, 56, –4 знак меняется 3 раза.)

9. Дана непустая последовательность чисел, оканчивающаяся числом 1000. Последовательность является неубывающей. Несколько чисел, идущих подряд, равны между собой. Найти количество таких чисел. Сколько различных чисел имеется в последовательности?

10. Дана последовательность чисел, упорядоченная по возрастанию, оканчивающаяся нулем и число k, не равное ни одному из чисел последовательности и такое, что a1< k< an . а) Определить сумму чисел последовательности, меньших k. б) Найти два элемента последовательности (их порядковые номера и значение) в интервале, между которыми находится значение k.

11. Дана последовательность, упорядоченная по возрастанию, которая заканчивается числом 1000 и число k, не равное ни одному из чисел последовательности и такое, что a1<k<an. Найти элемент последовательности (его порядковый номер и значение), ближайший к числу k.

12. Дана непустая последовательность целых чисел, оканчивающаяся числом 100. Определить, есть ли в последовательности число 77? Если имеются несколько таких чисел, то определить порядковый номер первого из них.

13. Дана последовательность натуральных чисел, оканчивающаяся числом 0. Определить, есть ли в последовательности хотя бы одна пара одинаковых "соседних" чисел. В случае положительного ответа определить порядковые номера чисел первой из таких пар.

14. Дана последовательность чисел, оканчивающаяся числом 10 000. Количество чисел в последовательности не меньше двух. Определить, является ли последовательность упорядоченной по возрастанию. В случае отрицательного ответа определить порядковый номер первого числа, нарушающего такую упорядоченность.

15. Дана последовательность натуральных чисел, оканчивающаяся числом 0. Определить, есть ли в последовательности пары одинаковых "соседних" чисел. В случае положительного ответа определить количество таких пар.

16. Дана последовательность натуральных чисел, оканчивающаяся числом -1. Определить среднее арифметическое тех из них, которые больше 10.

Пример выполнения задания

Формулировка задачи:

Дана последовательность чисел, оканчивающаяся нулем. Определить, есть ли в последовательности отрицательные числа. В случае положительного ответа определить порядковый номер первого из них.

Методика решения задачи

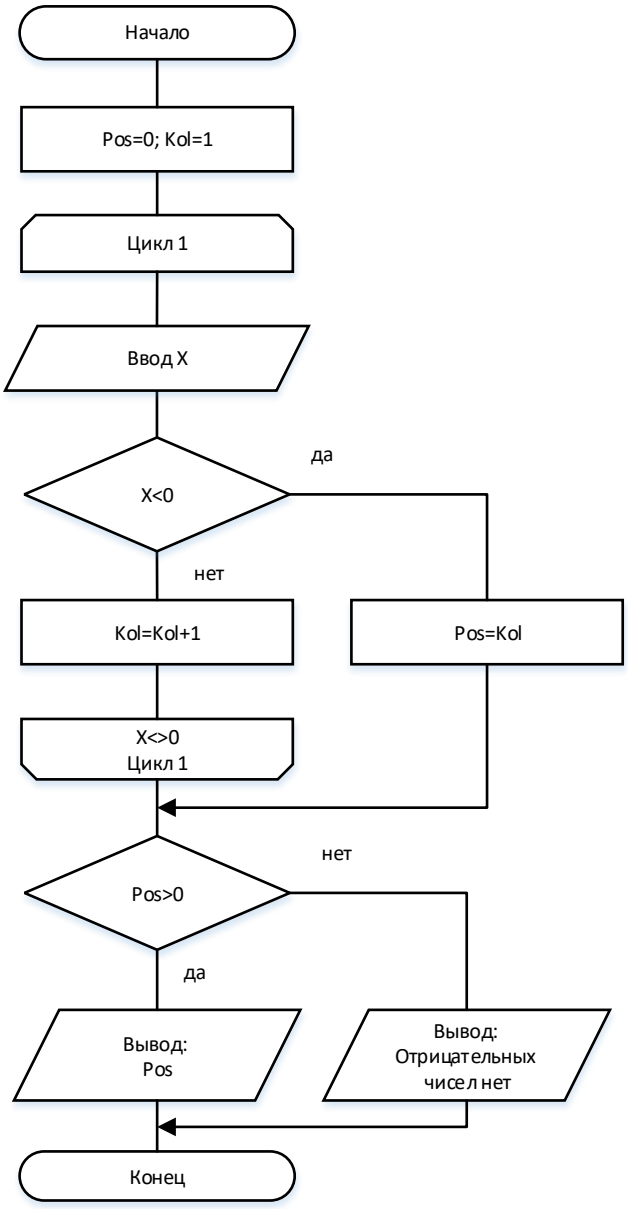
Так как дана последовательность чисел, нужно ее вводить в программе. Так как мы не умеем хранить много чисел в программе, то будем вводить числа поочередно в цикле и использовать для работы только одно текущее число. Так же при анализе задачи обратим внимание, что цикл может завершиться досрочно, когда найдем первое отрицательное число, порядковый номер которого нужно найти по заданию.

Решение:

Исходные данные: X – текущее число

Результат: Pos – порядковый номер отрицательного числа.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №7 – оператор цикла Do – While
//Дана последовательность чисел, оканчивающаяся нулем.
//Определить, есть ли в последовательности отрицательные числа.
//В случае положительного ответа определить порядковый номер первого из них.
//Выполнил Иванов Сергей гр. ИСП-21  31.03.2022

double value,   //Текущее число
kol = 1,    //Счетчик чисел
pos = 0;    //Порядковый номер от. числа
Console.WriteLine("Вводите числа последовательности");
do
{
    value = Convert.ToInt32(Console.ReadLine());// Ввод числа
    if (value < 0)
    {
        pos = kol;
        break;
    }
    kol++;
} while (value != 0);
if (pos > 0) Console.WriteLine("Порядковый номер отрицательного числа = " + pos);
else Console.WriteLine("Отрицательных чисел нет");
Console.ReadKey();//Пауза
```

Вложенные циклы

Существует возможность организовать цикл внутри тела другого цикла. Такой цикл будет называться вложенным циклом. Вложенный цикл по отношению к циклу в тело, которого он вложен, будет именоваться внутренним циклом, и наоборот цикл, в теле которого существует вложенный цикл, будет именоваться внешним по отношению к вложенному циклу. Внутри вложенного цикла в свою очередь может быть вложен еще один цикл, образуя следующий уровень вложенности и так далее. Количество уровней вложенности, как правило, не ограничивается.

Полное число исполнений тела внутреннего цикла не превышает произведения числа итераций внутреннего и всех внешних циклов. Например, взяв, три вложенных друг в друга цикла, каждый по 10 итераций, получим 10 исполнений тела для внешнего цикла, 100 для цикла второго уровня и 1000 в самом внутреннем цикле.

Пример 9.

Формулировка задачи:

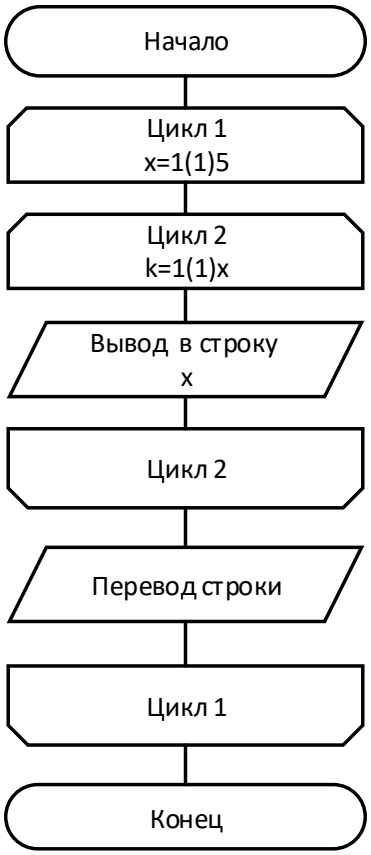
Напечатать числа в виде следующей таблицы:

1
2 2
3 3 3
4 4 4 4
5 5 5 5 5

Методика решения задачи:

Для решения этой задачи воспользуемся двумя вложенными циклами for. Первый цикл генерирует цифры 1, 2, 3, 4, 5. Второй вложенный цикл выводит полученную цифру столько раз, чему равна эта цифра.

Блок-схема алгоритма решения задачи:



Код программы:

```
//31.03.2022 Сидоров С.С. ИСП-21
//Пример9. Напечатать таблицу цифр, с повторением каждой цифры
//по значению этой цифры


for (int x = 1; x <= 5; x++) //Формируем цифру
{
    //Выводи эту цифру столько раз чкму равна эта цифра
    for (int k = 1; k <= x; k++)
        Console.Write(x + " ");
    Console.WriteLine();//Перевод строки
}
Console.ReadKey();//Пауза
```

Практическая работа №8

Вложенные циклы

Задание. Составьте алгоритм и программу выполнения варианта задания с использованием вложенных циклов и вывода исходных данных и результатов работы программы.

Варианты заданий (Базовый уровень)

1. Найти количество делителей каждого из целых чисел от 120 до 140.

Составить программу для графического изображения делимости чисел от 1 до n (значение n вводится с клавиатуры). В каждой строке надо напечатать очередное число и столько символов "+", сколько делителей у этого числа. Например, если n=4 то на экране должно быть напечатано:

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

2. Найти все целые числа из промежутка от 1 до 300, у которых ровно пять делителей.

3. Найти все трехзначные простые числа (простым называется натуральное число, большее 1, не имеющее других делителей, кроме единицы и самого себя).

4. Найти 100 первых простых чисел.

5. Найти все целые числа из промежутка от 100 до 300, у которых сумма делителей равна 50.

6. Натуральное число называется *совершенным*, если оно равно сумме своих делителей, включая 1 и, естественно, исключая это самое число. Например, совершенным является число 6=(1+2+3). Найти все совершенные числа, меньшие 100 000.

7. Дано натуральное число *n*. Вычислить $1^1 + 2^2 + ... + n^n$. Для вычисления степени функцию pow не использовать.

8. Дано натуральное число n (n 27). Найти все трехзначные числа, сумма цифр которых равна n.

9. Напечатать в возрастающем порядке все трехзначные числа, в десятичной записи которых нет одинаковых цифр.

10. Даны натуральные числа m и n. Получить все натуральные числа, меньшие n, квадрат суммы цифр которых равен m.

11. Найти все целые числа из промежутка от 300 до 600, у которых сумма делите- лей кратна 10.

12. Составить программу нахождения цифрового корня натурального числа. Цифровой корень данного числа получается следующим образом. Если сложить все цифры этого числа, затем все цифры найденной суммы и повторять этот процесс, то в результате будет получено однозначное число (цифра), которая и называется цифровым корнем данного числа.

13. Найти размеры всех прямоугольников, площадь которых равна заданному натуральному числу s и стороны которых выражены натуральными числами. При этом решения, которые получаются перестановкой размеров сторон считать разными.

14. Напечатать полную таблицу умножения в виде:

```
1 x 1 = 1    2 x 1 = 2    ...    9 x 1 =9
1 x 2 = 2    2 x 2 = 4    ...    9 x 2 =18
...          ...          ...    ...
1 x 9 = 9    2 x 9 =18    ...    9 x 9 =81
```

15. Доказать гипотезу Сиракуз на диапазоне чисел. Гипотеза Сиракуз утверждает, что любое натуральное число сводится к единице в результате повторения следующих действий над самим числом и результатами этих действий.

Если число четное следует разделить его на 2.

Если нечетное, то умножить его на 3, прибавить 1 и разделить на 2.

Варианты заданий (Повышенный уровень)

1. Два натуральных числа называются *дружественными*, если каждое из них равно сумме всех делителей другого (само другое число в качестве делителя не рассматривается). Найти все пары натуральных дружественных чисел, меньших 50 000.

2. Найти натуральное число из интервала от a до b с максимальной суммой делителей.

3. Имеется 100 рублей. Сколько быков, коров и телят можно купить на все эти деньги, если плата за быка — 10 рублей, за корову — 5 рублей, за теленка — полтинник (0,5 рубля) и надо купить 100 голов скота?

Пример выполнения задания

Формулировка задачи:

Найти все целые числа из промежутка от a до b, у которых количество делителей равно k.

Методика решения задачи

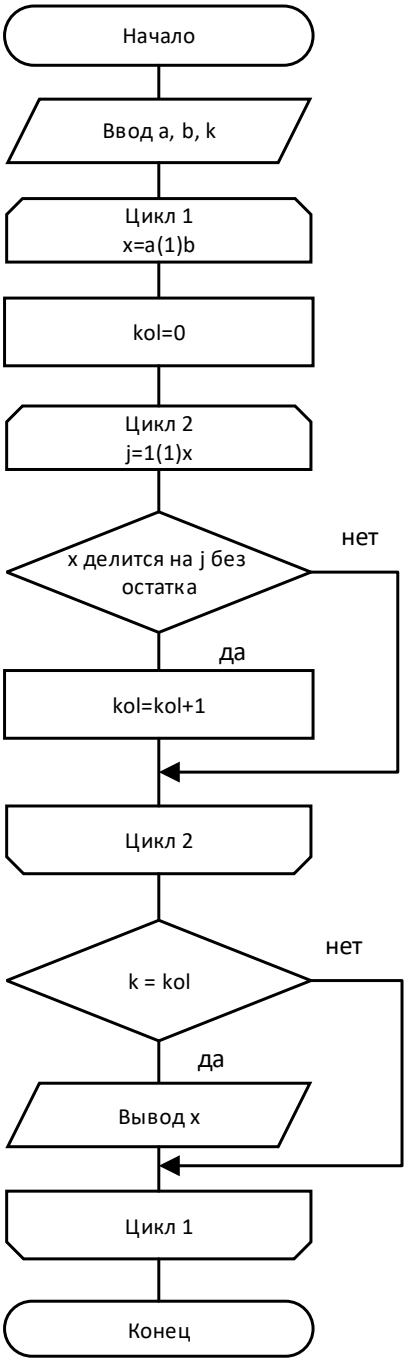
Для решения этой задачи воспользуемся двумя вложенными циклами for. Первый цикл генерирует цифры из промежутка от a до b. Второй вложенный цикл ищет для текущего числа количество делителей. Под делителями в данном случае понимаются числа, на которое делится это число без остатка. Делители ищутся методом простого перебора, т.е. число делится на 1, 2, 3, …, и на само себя.

Решение:

Исходные данные: a, b – промежуток чисел; к – количество делителей.

Результат: числа соответствующие условию задачи.

Блок-схема алгоритма решения задачи:



Код программы:

```
//Практическая работа №8 – Вложенные циклы
//Найти все целые числа из промежутка от a до b,
//у которых количество делителей равно k
//Выполнил Иванов Сергей гр. ИСП-21 31.03.2022

int a, b,   //Диапазон чисел
k,      //Количество искомых делителей
kol;    //Количество делителей числа
Console.WriteLine("Введите диапазон чисел a < b");
a = Convert.ToInt32(Console.ReadLine());
b = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Введите число k – количество искомых делителей ");
k = Convert.ToInt32(Console.ReadLine());
for (int x = a; x <= b; x++) // Формируем проверяемое число x
{
    kol = 0;
    for (int j = 1; j <= x; j++)//Перебираем делители числа x
        //Если x делится без остатка считаем такие делители
        if (x % j == 0) kol++;
    //Если число подходит по условию задачи, то вывод x
    if (k == kol) Console.Write(x + " ");
}
Console.WriteLine();
Console.ReadKey();//Пауза
```

Контрольные вопросы

(Оператор условия)

1. Какие программы называются разветвляющимися?
2. Как работает условный оператор? Нарисуйте фрагмент блок-схемы.
3. Назовите полную и сокращенную формы записи условного оператора.
4. С помощью, каких стандартных процедур осуществляется вывод на дисплей?
5. Какие логические операторы отношения используются в языке C#?
6. Что произойдет, если в операторе switch после метки case не использовать оператор break?
7. Что произойдет, если в операторе switch не поставить метку default и условие переключения не совпадет ни с одной меткой case?
8.
9. Как тестировать ветвящиеся алгоритмы?
10. Какие значения имеют переменные в начале выполнения программы?

11. Даны три положительных числа a,b,c. Определите, можно ли построить треугольник с таким и длинам и сторон.
12. Определите, есть ли среди цифр некоторого трехзначного числа повторяющиеся.
13. Определите, равна ли сумма крайних цифр некоторого четырехзначного числа сумме его средних цифр.

(Операторы цикла)

14. Какие программы называются циклическими?
15. Сколько операторов цикла Вам известно?
16. Каков формат цикла с предусловием?
17. Как исполняется цикл с предусловием?
18. Как в тело цикла включить несколько операторов?
19. Выполнится ли оператор в теле цикла с предусловием, если изначально значение логического выражения ложно?
20. Изобразите на блок-схеме оператор цикла с предусловием.
21. В каких случаях используется оператор цикла с предусловием?
22. Каков формат оператора цикла с постусловием?
23. Как исполняется оператор цикла с постусловием?
24. Почему в цикле с постусловием серия операторов из тела цикла будет выполнена хотя бы один раз?
25. В каком случае цикл с постусловием прекращается?
26. Какой тип должны иметь начальное и конечное значения в цикле с параметром?
27. Как исполняется цикл по возрастанию с параметром?
28. Можно ли в оператор цикла for включать действия по изменению параметра?
29. Как исполняется цикл по убыванию с параметром?
30. Какие циклы называются вложенными?
31. Какой цикл называется внешним? Внутренним?
32. Как организовать принудительный выход из цикла?
33. Как принудительно начать новую итерацию цикла, если предыдущая итерация не завершена?
34. Каждая бактерия делится на две в течение одной минуты. В начальный момент времени имеется одна бактерия. Составьте программу для расчета количества бактерий через заданное количество минут.