# ПРАКТИЧЕСКАЯ РАБОТА № 7. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ

# НА ЯЗЫКЕ C#

**Цель работы**: овладение практическими навыками разработки и программирования алгоритмов циклической структуры; приобретение дальнейших навыков по отладке и тестированию программ.

|  |  |  |
| --- | --- | --- |
| №  п/п | Ф.И.О. | НОМЕР ВАРИАНТА |
|  | Балицкая Анастасия |  |
|  | Вартик Максим |  |
|  | Вдовиченко Ярослав |  |
|  | Веряскин Иван |  |
|  | Гоцалюк Арзу |  |
|  | Градинар Илья |  |
|  | Гребенщиков Данил |  |
|  | Завтонев Роман |  |
|  | Захаров Станислав |  |
|  | Иванов Богдан |  |
|  | Ильинский Никита |  |
|  | Каплин Егор |  |
|  | Коваленко Михаил |  |
|  | Колак Михаил |  |
|  | Левицкая Мария |  |
|  | Милюкова Екатерина |  |
|  | Никифорова Елизавета |  |
|  | Подолян Евгений |  |
|  | Попов Максим |  |
|  | Родичев Максим |  |
|  | Рознован Роман |  |
|  | Тизул Игорь |  |
|  | Тихончук София |  |
|  | Толоченко Павел |  |
|  | Фокша Кирилл |  |
|  | Чебан Евгений | 5 |
|  | Чебан Илья | 7 |
|  | Юраш Ренат | 10 |

**Теоретическая часть**

*Командой повторения* или *циклом* называется такая форма организации действий, при которой одна и та же последовательность действий повторяется до тех пор, пока сохраняется значение некоторого логического выражения. При изменении значения логического выражения на противоположное повторения прекращаются (цикл завершается).

Для организации цикла необходимо выполнить следующие действия:

1. перед началом цикла задать начальное значение параметра;
2. внутри цикла изменять параметр цикла с помощью оператора присваивания;
3. проверять условие повторения или окончания цикла;
4. управлять циклом, т.е. переходить к его началу, если он не закончен, или выходить из цикла в противном случае.

Различают циклы с известным числом повторений (цикл с параметром) и итерационные (с пред- и постусловием).

В цикле с известным числом повторений параметр изменяется в заданном диапазоне.

Если в цикле изменяется простая переменная, то она является параметром цикла; если в цикле изменяется переменная с индексом, то индекс этой переменной является параметром цикла.

К операторам цикла относятся: цикл с предусловием *while*, цикл с постусловием *do while*, цикл с параметром *for* и цикл перебора *foreach*.

##### Цикл с предусловием while

Оператор цикла *while* организует выполнение одного оператора (простого или составного) неизвестное заранее число раз. Формат цикла *while*:

**while ( выражение B) оператор** **S**

где B - выражение, истинность которого проверяется (условие завершения цикла); S - тело цикла - оператор (простой или составной).

Выражение B определяет условие повторения тела цикла: перед каждым выполнением тела цикла анализируется значение выражения В: если оно истинно, то выполняется тело цикла, и управление передается на повторную проверку условия В; если значение В ложно - цикл завершается и управление передается на оператор, следующий за оператором S.

Если результат выражения B окажется ложным при первой проверке, то тело цикла не выполнится ни разу. Отметим, что если условие B во время работы цикла не будет изменяться, то возможна ситуация зацикливания, то есть невозможность выхода из цикла. Поэтому внутри тела должны находиться операторы, приводящие к изменению значения выражения B так, чтобы цикл мог корректно завершиться.

*Пример*: Вывести на экран целые числа из интервала от 1 до *n*.

static void Main()

{

Console.Write("N= ");

int n=int.Parse(Console.ReadLine());

int i = 1;

while (i <= n) //пока i меньше или равно n

Console.Write(" "+ i++ ); //выводим i на экран, затем увеличиваем его на 1

}

Результаты работы программы:

n ответ

10 1 2 3 4 5 6 7 8 9 10

##### Цикл с постусловием do while

Оператор цикла *do while* также организует выполнение одного оператора (простого или составного) неизвестное заранее число раз. Однако в отличие от цикла *while* условие завершения цикла проверяется после выполнения тела цикла. Формат цикла do while:

**do оператор S while выражение** **B;**

где В - выражение, истинность которого проверяется (условие завершения цикла); оператор S - тело цикла - оператор (простой или блок).

Сначала выполняется оператор S, а затем анализируется значение выражения В: если оно истинно, то управление передается оператору S, если ложно - цикл завершается, и управление передается на оператор, следующий за условием B. Так как условие В проверяется после выполнения тела цикла, то в любом случае тело цикла выполнится хотя бы один раз.

В операторе *do while*, так же как и в операторе *while*, возможна ситуация зацикливания в случае, если условие В всегда будет оставаться истинным.

*Пример*: Вывести на экран целые числа из интервала от 1 до *n*.

static void Main()

{

Console.Write("N= ");

int n=int.Parse(Console.ReadLine());

int i = 1;

do

Console.Write(" " + i++);

//выводим i на экран, затем увеличиваем его на 1

while (i <= n); //пока i меньше или равно n

}

##### Цикл с параметром for

Цикл с параметром имеет следующую структуру:

**for (<инициализация>; <выражение>; <модификация>) <оператор>;**

Инициализация используется для объявления и/или присвоения начальных значений величинам, используемым в цикле в качестве параметров (счетчиков). В этой части можно записать несколько операторов, разделенных запятой. Областью действия переменных, объявленных в части инициализации цикла, является цикл и вложенные блоки. Инициализация выполняется один раз в начале исполнения цикла.

Выражение определяет условие выполнения цикла: если его результат истинен, цикл выполняется. Истинность выражения проверяется перед каждым выполнением тела цикла, таким образом, цикл с параметром реализован как цикл с предусловием. В блоке выражение через запятую можно записать несколько логических выражений, тогда запятая равносильна операции логическое И (&&).

Модификация выполняется после каждой итерации цикла и служит обычно для изменения параметров цикла. В части модификация можно записать несколько операторов через запятую.

Оператор (простой или составной) представляет собой тело цикла.

Любая из частей оператора for (инициализация, выражение, модификация, оператор) может отсутствовать, но точку с запятой, определяющую позицию пропускаемой части, надо оставить.

static void Main()

{

Console.Write("N= ");

int n=int.Parse(Console.ReadLine());

for (int i=1; i<=n;) //блок модификации пустой

Console.Write(" " + i++);

}

Оператор do while обычно используют, когда цикл требуется выполнить хотя бы один раз. Оператором while удобнее пользоваться в случаях, когда число итераций заранее не известно. Оператор for предпочтительнее в большинстве остальных случаев.

*Цикл перебора foreach*

Оператор *foreach* применяется для перебора элементов в специальным образом организованной группе данных, например в массиве. Удобство этого вида цикла заключается в том, что нам не требуется определять количество элементов в группе и выполнять перебор по индексу - мы просто указываем на необходимость перебрать все элементы группы.

Синтаксис оператора:

**foreach (<тип> <имя> in <группа>) <тело цикла>**

где имя определяет локальную по отношению к циклу переменную, которая будет по очереди принимать все значения из указанной группы, а тип соответствует базовому типу элементов группы.

Ограничением оператора *foreach* является то, что с его помощью можно только просматривать значения элементов в группе данных, но нельзя их изменять.

Пример использования оператора *foreach* для работы с одномерными массивами:

static void PrintArray(string a, int [] mas)

{

Console.WriteLine(a);

foreach (int x in mas)Console.Write("{0} ", x);

Console.WriteLine();

}

*Вложенные циклы*

Циклы могут быть простые или вложенные (кратные, циклы в цикле). Вложенными могут быть циклы любых типов: while, do while, for. Каждый внутренний цикл должен быть полностью вложен во все внешние циклы. "Пересечения" циклов не допускаются.

*Пример*: Вывести на экран числа следующим образом:

2 2 2 2 2

2 2 2 2 2

2 2 2 2 2

2 2 2 2 2

static void Main()

{

for (int i = 1; i <= 4; ++i, Console.WriteLine()) //1

for (int j=1; j<=5; ++j)

Console.Write(" " + 2);

}

Замечание. В строке 1 в блоке модификации содержится два оператора ++i и Console.WriteLine(). В данном случае после каждого увеличения параметра *i* на 1 курсор будет переводиться на новую строку.

##### Оператор перехода к следующей итерации цикла continue

Оператор перехода к следующей итерации цикла continue пропускает все операторы, оставшиеся до конца тела цикла, и передает управление на начало следующей итерации (повторение тела цикла).

*Пример*:

static void Main()

{

Console.WriteLine("n=");

int n = int.Parse(Console.ReadLine());

for (int i = 1; i <= n; i++)

{

if (i % 2 == 0) continue;

Console.Write(" " + i);

}

}

**Вопросы для самоконтроля**

1. Из каких элементов состоят циклические операторы?
2. Что необходимо для выполнения в цикле более одного оператора?
3. В чем основное отличие операторов цикла с предусловием и постусловием?
4. Для чего используется оператор continue?
5. Для чего используется оператор break? goto?
6. Когда целесообразно использовать оператор цикла for?
7. Как привести цикл while к циклу for и наоборот?
8. Какие операторы цикла существуют в языке С#?
9. Что представляет собой «тело цикла»?
10. Что такое «итерация»?
11. Что такое «счетчик цикла»?
12. Верно ли, что истинность выражения в цикле с предусловием является условием продолжения цикла?
13. Сколько раз выполнится оператор в теле цикла while, если с самого начала значение выражения равно false?

**Практическая часть**

**Задание №1.** Вывести на экран (задачу решите тремя способами - используя операторы цикла while, do while и for):

* 1. целые числа 1, 3, 5, …, 101 в строчку через пробел;
  2. целые числа 10, 12, 14, …, 80 в обратном порядке в столбик;
  3. таблицу соответствия между весом в фунтах и весом в килограммах для значений 1, 2, 3, …, 100 фунтов (1 фунтов = 453г);
  4. таблицу перевода 5, 10, 15, …, 500 долларов США в рубли по текущему курсу (значение курса вводится с клавиатуры);
  5. таблицу стоимости для 10, 20, 30,…, 200 штук товара, при условии, что одна штука товара стоит *х* руб (значение *х* водится с клавиатуры);
  6. таблицу перевода расстояний в дюймах в сантиметры для значений 2, 4, 6, …, 24 дюймов (1 дюйм = 25.4 мм);
  7. кубы всех целых чисел из диапазона от А до В () в обратном порядке;
  8. все целые числа из диапазона от А до В (), оканчивающиеся на цифру Х;
  9. все целые числа из диапазона от А до В (), оканчивающиеся на цифру Х или У;
  10. все целые числа из диапазона от А до В (), оканчивающиеся на любую четную цифру;
  11. только положительные целые числа из диапазона от А до В ();
  12. все целые четные числа из диапазона от А до В (), оканчивающиеся на цифру Х или У в обратном порядке;
  13. все целые числа из диапазона от А до В, кратные трем ();
  14. все трехзначные числа, в записи которых две цифры одинаковые;
  15. все четные числа из диапазона от А до В, кратные пяти ();
  16. только отрицательные четные числа из диапазона от А до В () в обратном порядке;
  17. все двухзначные числа, в записи которых все цифры разные;
  18. все двухзначные числа, в которых старшая цифра отличается от младшей не больше чем на 1;
  19. все трехзначные числа, которые начинаются и заканчиваются на одну и ту же цифру;
  20. все трехзначные числа, в которых хотя бы две цифры повторяются;
  21. числа следующим образом:

10 10.4

11 11.4

…

25 25.4

* 1. числа следующим образом:

25 25.5 24.8

26 26.5 25.8

…

35 35.5 34.8

* 1. квадраты всех целых нечетных чисел из диапазона от А до В () в прямом и обратном порядке;
  2. все целые отрицательные числа из диапазона от А до В, кратные семи ();
  3. все трехзначные числа, в которых старшая цифра отличается от младшей не больше чем на 3.

**Задание №2.** Вывести на экран числа следующим образом:

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

**Задание №3.** Решите задачу:

1. Найти среднее арифметическое всех целых чисел от 1 до 1000.
2. Найти среднее арифметическое всех целых чисел от 100 до *b* (значение *b* вводится с клавиатуры; );
3. Найти среднее арифметическое всех целых чисел от *а* до 200 (значение *а* вводится с клавиатуры; );
4. Найти среднее арифметическое всех целых чисел от *а* до *b* (значения *а* и *b* вводятся с клавиатуры; );
5. Найти сумму квадратов всех целых чисел от *а* до 50 (значение *а* вводится с клавиатуры; );
6. Найти сумму квадратов всех целых чисел от 1 до *n* (значение *n* вводится с клавиатуры; );
7. Найти сумму квадратов всех целых чисел от *а* до *b* (значения *а* и *b* вводятся с клавиатуры; );
8. Дано натуральное число *n*. Напечатать разложение этого числа на простые множители. Реализовать вариант каждый простой множитель должен быть напечатан один раз;
9. Дано натуральное число *n*. Напечатать разложение этого числа на простые множители. Реализовать вариант каждый простой множитель должен быть напечатан столько раз сколько раз он входит в разложение.
10. Дано натуральное число *n*. Получить все простые делители этого числа.
11. Дано натуральное число *n*. Получить все натуральные числа, меньшие *n* и взаимно простые с ним (два натуральных числа называются взаимно простыми, если их наибольший общий делитель равен 1).
12. Даны целые числа *n* и *m*. Получить все натуральные числа, меньшие *n* и взаимно простые с *р*.
13. Даны целые числа *n* и *m*. Получить все делители числа *m*, взаимно простые с *р*.
14. Составить программу для графического отображения 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 *п*=4, то на экране должно быть напечатано:

1+

2++

3++

4+++

1. Найти все целые числа из промежутка от *a* до *b*, у которых количество делителей равно *k*.
2. Найти натуральное число *m* интервала от *a* до *b*, у которого количество делителей максимально. Если таких чисел несколько, то должно быть найдено максимальное из них;
3. Найти натуральное число *m* интервала от *a* до *b*, у которого количество делителей максимально. Если таких чисел несколько, то должно быть найдено минимальное из них
4. Найти все трехзначные простые числа (простым называется натуральное число, большее 1, не имеющее других делителей, кроме единицы и самого себя).
5. Найти все целые числя из промежутка от 300 до 600, у которых сумма делителей кратна 10.
6. Натуральное число называется совершенным, если оно равно сумме своих делителей, включай 1 и, естественно, исключая это самое число. Например, совершенным является число 6 (6=1+2+3). Найти все совершенные числа, меньшие 100 000.
7. Найти натуральное число из интервала от *a* до *b* с максимальной суммой делителей.
8. Найти все целые числа из промежутка от 100 до 300, у который сумма делителей равна 50.
9. Дано натуральное число *п* (). Найти все трехзначные числа, сумма цифр которых равна *п*. Операции деления, целочисленного деления и определения остатка не использовать.
10. Напечатать в возрастающем порядке все трехзначные числа, в десятичной записи которых нет одинаковых цифр. Операции деления, целочисленного деления и определения остатка не использовать.
11. Даны натуральные числа *m* и *n*. Получить все натуральные числа, меньшие *n*, квадрат суммы цифр которых равен *m*.