**Практическое задание 2.**

**Основы алгоритмизации**

Цель работы ‒ изучение циклических алгоритмов и их описание на языке блок-схем.

**ПЛАН**

[1. Краткое теоретическое введение 1](#_Toc18871095)

[2. Правила организации циклических алгоритмов 2](#_Toc18871096)

[3. Цикл с параметром 3](#_Toc18871097)

[3. Итерационные циклы 7](#_Toc18871098)

# 1. Краткое теоретическое введение

В прикладных задачах достаточно часто есть необходимость в неоднократном повторении одних и тех же действий. Алгоритм, в котором многократно повторяется какой-либо фрагмент, называется *циклическим*.

Примерами циклических являются алгоритмы вычисления таблиц значений функций, накопление суммы, счет количества, учет предыдущего значения и так далее. В наборах данных это поиск, сортировка и им подобные.

*Телом* *цикла* называется блок многократно повторяющихся команд. Собственно, цикл и организуется для выполнения тела.

Для управления процессом повторения вычислений служат другие составляющие циклического алгоритма: начальные установки, модификация параметра цикла внутри тела, и проверка условия завершения цикла. Один проход цикла называется еще *итерацией*.

Результативность циклического алгоритма означает, что за конечное число шагов он закончится. Практически это обеспечивает некоторая переменная, значение которой изменяется в теле цикла таким образом, чтобы на очередной итерации процесс закончился. Такая переменная называется *управляющей* переменной цикла или *параметром* цикла.

Начальные установки нужны, чтобы до входа в цикл задать значения переменных, которые в нем используются, в том числе, значение управляющей переменой.

В теле цикла управляющая переменная изменяется.

На каждой итерации выполняется проверка условия завершения цикла. Это может происходить до тела цикла (*цикл с предусловием*), либо после тела цикла (*цикл с постусловием*). Тело цикла с постусловием всегда выполняется хотя бы один раз. Проверка необходимости выполнения цикла с предусловием делается до начала цикла, поэтому возможно, что он не выполнится ни разу.

Схема организации циклических алгоритмов приведена на рисунке ниже.

|  |  |
| --- | --- |
|  |  |
| Цикл с предусловием | Цикл с постусловием |

# 2. Правила организации циклических алгоритмов

Циклические алгоритмы разделяются на две группы.

*Арифметический цикл* (управляемый счетчиком). Как правило, повторяется заранее известное число раз. Например, спортсмен должен пробежать 10 кругов или 40 км. Например, нужно вычислить прирост продаж за 5 лет.

Итерационный (управляемый событием). Как правило, число повторений заранее неизвестно. Например, спортсмен должен бежать, пока не устанет, или пока суммарный путь пробега не составит 42 км. Например, пока объем продаж не станет больше 5% от суммы вложений.

Управление процессом выполнения циклического алгоритма организует правильно выбранная управляющая переменная (параметр цикла). Это одна из переменных, которая, как правило, изменяется в теле цикла, определяет число повторений цикла и позволяет завершить его работу.

При выполнении цикла всегда должны быть пройдены следующие этапы.

Подготовка цикла: включает действия, которые не относятся непосредственно к логической схеме цикла, но позволяют правильно его выполнить. Как правило, на этом этапе выполняется присваивание начальных значений переменным, в том числе параметру цикла.

Итерация: очередное выполнение тела цикла, то есть фрагмента, который должен быть повторен многократно. Как правило, включает в себя изменение параметра цикла.

Точка проверки условия: момент проверки условия, при котором решается, делать ли новую итерацию, или перейти к оператору, стоящему за циклом. Как правило, в проверке условия явно или нет, присутствует параметр цикла.

Выход из цикла: передача управления блоку, стоящему за циклом.

Не всегда эти составляющие присутствуют явным образом.

# 3. Цикл с параметром

Третий вид циклов – цикл с *параметром* или цикл *со счетчиком*.

Как правило, это *арифметический цикл*. Тело цикла выполняется, пока параметр цикла *i* пробегает множество значений от начального (*In*) до конечного (*Ik*).

Управляющая переменная *i* определяет количество повторений тела цикла *S*. Если шаг изменения значения параметра цикла обозначить через *ΔI*, то количество повторений тела цикла *n* можно вычислить по формуле:



Если параметр цикла *i* изменяется с шагом 1, то шаг может не указываться.

|  |  |
| --- | --- |
|  | Цикл выполняется так: начальное значение параметра цикла *i* равно *In*. Если *i ≤ Ik*, выполняется тело цикла *S*, после чего параметр цикла увеличивается на 1 с помощью оператора присваивания i = i + 1 и снова проверяется условие *i ≤ Ik*. Тело цикла выполняется до тех пор, пока значение *i* не превысит *Ik*. |

**Пример 1**

Дано целое положительное число *n*. Вычислить факториал этого числа. Известно, что факториал любого целого положительного числа *n* определяется как произведение чисел от 1 до заданного числа *n*:

*n*! = 1∙2∙3∙…∙*n*.

По определению *0! = 1* и *1! = 1*.

Задача решается с помощью циклического алгоритма. Введем следующие обозначения: *N* – заданное число, *F* – факториал числа, *R* – параметр цикла. Составим два варианта алгоритма: с использованием цикла с предусловием и цикла с параметром.

|  |  |
| --- | --- |
| Цикл с предусловием | Цикл с параметром |
|  |  |

Правильность алгоритма проверяется вручную. Так, при *n* = 4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Цикл с предусловием | | |  | Цикл с параметром | | |
| R | R <= N | F |  | R | R <= N | F |
| 1 | Да | 1⋅1 = 1 |  | 1 | Да | 1⋅1 = 1 |
| 2 | Да | 1⋅2 = 2 |  | 2 | Да | 1⋅2 = 2 |
| 3 | Да | 2⋅3 = 6 |  | 3 | Да | 2⋅3 = 6 |
| 4 | Да | 6⋅4 = 24 |  | 4 | Да | 6⋅4 = 24 |
| 5 | Нет | – |  | 5 | Нет | – |

При решении данной задачи выполнение цикла с предусловием ничем не отличается от выполнения цикла с параметром. При R = 5 произойдет выход из цикла и окончательное значение 4! = 24.

В чем отличие? В структуре циклов. В цикле *пока*начальное значение параметра цикла *R* задается перед входом в цикл, в теле цикла организовано изменение параметра цикла командой *R = R + 1*, условие *R ≤ N* определяет условие продолжение цикла. В цикле с заданным числом повторений эти же команды неявно заданы в операторе заголовка цикла.

**Пример 2**

Приведем пример проектирования циклического алгоритма, в котором использованы различные схемы цикла для решения одной и той же задачи.

**Условие задачи**: снаряд выпущен под углом *λ* к горизонту со скоростью *V* (рисунок). Требуется определить высоту и дальность полета в течение промежутка времени от *t* = 1 сек. до *t* = 10 сек. с интервалом, равным 1 сек.

*λ*

y

x

*V*

Содержанием тела цикла является вычисление и вывод на печать очередных значений горизонтальной и вертикальной составляющих расстояния.



Управляющая переменная определяется логикой задачи. В этом случае число повторений заранее известно, и равно 10-ти, значит, цикл арифметического типа. Определяем изменение параметра цикла и условие выхода: t = [0,10] Δt=1.

|  |  |
| --- | --- |
| Начало  λ, V  t=0; 10; 1  Sx = V t cos λ  Sy = V t sin λ – 0.5 g t2  t,Sx,Sy  Конец | Результат решения в этой задаче сильно зависит от входных данных, так как наибольшая дальность полета может быть достигнута за время, меньшее, чем 10 секунд.  При этом координата, определяющая высоту, становится отрицательной, как будто снаряд пробил землю и продолжает лететь под землей.  Этот факт требует улучшения постановки задачи, так как нужно отслеживать траекторию полета, пока снаряд находится в движении, то есть пока его вертикальная координата больше нуля. |

|  |  |
| --- | --- |
| Да  Нет  Начало  λ, V  Sx = V t cos λ  Sy = V t sin λ – 0.5 g t2  t,Sx,Sy  Конец  t=0  t=t+1  Sy>0 | Следовательно, от арифметического цикла следует отказаться, и выбрать итерационный, в котором условием завершения вычислений будет условие «пока снаряд находится в полете», то есть его вертикальная координата больше 0.  Итерационный цикл можно реализовать с использованием любого из видов цикла. |

|  |  |
| --- | --- |
| Да  Нет  Начало  λ, V  Sx = V t cos λ  t,Sx  Конец  t=0  t=t+1  V t sin λ  – 0.5 g t2>0 | Изменим постановку задачи, чтобы получить чисто итерационный процесс.  Например, определить время полета и его дальность при заданных начальных значениях скорости и угла.  В этой постановке задачи итоговыми значениями являются t и Sx, но для проверки условия завершения нужно вычислять значение вертикальной координаты.  Переменная Sy не нужна, так как формула проверки условия записана в виде логического выражения.  Вывод заголовка не входит в подготовку цикла, а вывод на печать вынесен за тело цикла. |

**Пример 3. Вычисление последовательностей**

Пусть задана последовательность *а*0 = 1; *а*k = *k* ⋅ *a*k–1 + 1/*k*, где *k* = 1,2,3… Для некоторого натурального числа *n* получить *а*n.

*Дано*: *а*0 – первый член последовательности; *n* – номер члена последовательности, значение которого требуется найти.

*Найти*: *аn* – *n*-й член последовательности.

*Математическая модель*. Посмотрим, как изменяется значение члена последовательности при изменении значения *k*.

При *k* = 1 *а*1 = 1 ⋅ *a*0 + 1.

При *k* = 2 *а*2 = 2 ⋅ *a*1 + 1/2.

При *k* = 3 *а*3 = 3 ⋅ *a*2 + 1/3.

Выполнив указанные вычисления *n* раз, получим искомое значение *аn*. В задачах такого типа не требуется хранить результаты вычислений на каждом шаге. Поэтому можно использовать простые переменные.

Обозначим через *a* – произвольный член последовательности. Тогда формула для вычисления члена последовательности будет выглядеть так:   
*а* = *k ⋅ a* + 1/*k*.

В этой формуле значение *а*, стоящее справа от знака «=», определяется на предыдущем шаге вычисления, а значение *а*, стоящее в левой части выражения, определяется на данном шаге и заменяет в памяти предыдущее значение.

Переменная *а* – вещественного типа; переменные *k* и *n* – целого типа.

|  |  |
| --- | --- |
|  | 1. Ввод количества членов последовательности *n*  2*.*Начальное значение *a* = 1  3. Арифметический цикл. Начальное значение параметра цикла *k* равно *1*, конечное значение равно *n*, шаг изменения параметра – 1.  4. Для каждого значения *k* вычисляется выражение *а = k \* a + 1/ k*.  5. Вывод *а* |

# 4. Рекуррентные соотношения и итерационные циклы

Циклы с заранее неизвестным количеством повторений называются *итерационными*. Итерационный процесс – это последовательное приближение к результату за некоторое количество шагов.

Рассмотрим блок-схемы итерационных циклов на примерах.

**Пример 4**. Даны действительные числа *x, ε* (x ≠ 0, ε > 0). Вычислить сумму с точностью *ε*:

.

Вычисление бесконечной суммы с заданной точностью *ε* означает, что требуемая точность достигнута, когда вычислена сумма нескольких первых слагаемых и очередное слагаемое оказалось по модулю меньше, чем *ε*, – это и все последующие слагаемые можно не учитывать.

Пусть *с* – слагаемое, *S* – сумма, *F* – факториал числа *k.*

*Математическая модель*. Выведем рекуррентное соотношение, используя способ, показанный в примере 1.5.

Для вычисления факториала числа можно воспользоваться формулой *F* = *F* ⋅ *k*, которую мы вывели ранее. Начальное значение *F* = 1. Проверьте правильность этой формулы, последовательно вычисляя значение *k*!

Выведем общую формулу для вычисления одного слагаемого (без учета факториала числа *k*).

*k* = 0, *с* = *x*0 / 20 = 1;

*k* = 1, *с* = *x*1/ 21 = *x*/ 2;

*k* = 2, *с* = *x*2/ 22 = (*x*/ 2) (*x*/ 2) = *с ⋅ x*/ 2;

*k* = 3, *с* = *x*3/ 23 = (*x*2/ 4) (*x*/ 2) = *с ⋅ x*/ 2;

*k* = 4, *с* = *x*4/ 24 = (*x*3/ 8) (*x*/ 2) = *с ⋅ x*/ 2.

Обобщая, для произвольного значения *k* можно записать c = с ⋅*x*/2, где значение переменной *с* в правой части формулы вычисляется на предыдущем шаге. Начальное значение *с* =1 при *k* = 0.

Сумму вычисляем по формуле *S = S + c / F* (см. пример 1.5). Начальное значение *S = 0*.

В данной задаче следует использовать цикл с предусловием, потому что количество слагаемых будет зависеть от введенного значения *x* и требуемой точности вычислений *ε*.

Переменная *k* – целого типа, переменные *x, ε, c, S* – вещественного типа. Переменную *F* следует взять вещественного типа, т.к. диапазон целых чисел ограничен, а факториал быстро возрастает с ростом значения *k*.

|  |  |
| --- | --- |
|  | 1. Ввод исходных данных.  2. Присваивание переменным начальных значений. Параметр цикла *k = 1*.  3. Прежде чем вычислять сумму, определим выполнение условия продолжения цикла. Так как может быть введено отрицательное значение *х*, то использована функция *Abs(),* определяющая модуль числа. |

4. По выходу «Да» блока 3 определяем новые значения переменных и продолжаем цикл.

5. При выходе из цикла (выход «Нет») блока 3 печатаем значение переменной *S*.

Цикл, который мы использовали при решении задачи, называется *итерационным*. Особенностью такого цикла является то, что число его повторений зависит от выполнения условия, которое записывается в заголовке цикла. В итерационных алгоритмах необходимо обеспечить обязательное достижение условия выхода из цикла (сходимость итерационного процесса). В противном случае произойдет «зацикливание» алгоритма. Управление итерационным циклом организует программист, который должен позаботиться и об инициализации управляющей переменной, и об ее приращении, и об условии завершения цикла.

# Упражнения

**Упражнение 1.**

Пусть для x ∈ [–π/2;+π/2] требуется вычислить таблицу значений функции, имеющей разрыв в точках | x | = π/4, по формуле:



**Упражнение 2**

Мяч брошен вертикально вверх со скоростью V. Требуется построить таблицу зависимости высоты Y от времени t, начиная с момента броска до момента падения мяча на землю, если

 , где g – константа тяготения, равная 9,8.

**Упражнение 3**

Составить программу для определения наибольшего общего делителя (НОД) двух простых чисел m и n по алгоритму Евклида:

НОД = m, если m = n, иначе, если m > n, то m = m – n, иначе, если m < n, то n = n – m. Значения n, m вводить в диалоге, иметь возможность повторного обращения.