Лабораторная работа №3. Циклические алгоритмы и рекурсия

Циклы

При написании любого цикла надо иметь в виду, что в нем всегда явно или неявно присутствуют четыре элемента: начальные установки, тело цикла, модификация параметра цикла и проверка условия продолжения цикла. Начинающие часто забывают про первое и/или третье.

Вообще (а не только при использовании циклов) рекомендуется определять переменные непосредственно перед использованием. Это является хорошим стилем, поскольку снижает вероятность ошибок.

Допустимо определять переменные внутри тела цикла, а в случае цикла **for** в заголовке цикла. Имейте в виду, что переменная, объявленная внутри цикла, при каждой итерации будет заново инициализироваться. Если переменная объявлена в заголовке цикла, то областью ее видимости является только цикл (в некоторых версиях компиляторов есть «глюк» – переменная видна и после цикла). Если переменная вне цикла не требуется (например, счетчик цикла), то рекомендуется объявлять ее в заголовке.

Если количество итераций цикла заранее известно, рекомендуется использовать цикл **for**. Это хорошо еще и тем, что все управление циклом **for** сосредоточенно в его заголовке, что делает программу более читаемой, а так же помогает избегать ошибок.

Оператор **do while** обычно используют, когда цикл требуется выполнить хотя бы один раз, – например, если в цикле производится ввод данных. Оператором **while** удобнее пользоваться в тех случаях, когда либо число итераций заранее неизвестно, либо очевидных параметров цикла нет, либо модификацию параметров удобнее записывать не в конце тела цикла. Оператор **for** предпочтительнее в большинстве остальных случаев. Так же следует заметить, что в языке С для цикла **for** в качестве проверки условия продолжения цикла может быть записано любое выражение, даже никоим образом не зависящее от параметра цикла. Поэтому с некоторой осторожностью цикл **for** можно использовать, даже если число итераций заранее неизвестно.

Если количество повторений цикла заранее неизвестно, необходимо предусматривать аварийный выход из цикла по достижении некоторого количества итераций.

Особенности работы с функциями

Функция — это именованная последовательность операторов, выполняющая законченное действие. Функции нужны для упрощения структуры программы. Для решения задач данной лабораторной работы рекомендуется использовать функции.

Интерфейс грамотно написанной функции определяется ее заголовком. Для вызова функции надо указать ее имя и набор аргументов.

Печать диагностических сообщений внутри функции крайне нежелательна. Рекомендуется отделять логику программы от пользовательского интерфейса. Иными словами, никакие функции, кроме интерфейсных и main, не должны «общаться» с устройствами ввода и вывода.

Рекурсивные функции

Рекурсивной является функция, вызывающая сама себя. Такая рекурсия называется простой. Существует еще и сложная рекурсия, когда несколько функций вызывают друг друга таким образом, что вызовы оказываются «закольцованными». Далее будем рассматривать только простую рекурсию.

Подготовлено Латухиной Е.А. и Бурковым В.А.

Рекурсивные алгоритмы, как правило, короче своих нерекурсивных аналогов и выглядят более «красивыми» с математической точки зрения. Однако довольно часто рекурсивная реализация оказывается менее эффективной с точки зрения времени работы программы или занятой памяти. Поэтому рекурсию стоит использовать очень осторожно.

Генерация псевдослучайных чисел

Для генерации случайных чисел используется команда rand(), для работы которой необходимо подключить заголовочный файл `<stdlib.h>`. Для того, чтобы генератор каждый раз начинал генерацию с нового числа, необходимо подключить заголовочный файл `<time.h>`:

```
#include <stdlib.h>
#include <time.h>
```

Для того, чтобы генератор каждый раз возвращал различные числа, необходимо до использования rand (желательно в начале функции main) написать следующую строку в программе:

```
srand((unsigned)time(0));
```

Функция rand() генерирует псевдослучайные числа в диапазоне от 0 до 2^{15}-1. Чтобы сгенерировать целое число в диапазоне от a до b, необходимо добавить в программу следующую функцию:

```
int irand(int a, int b)
{
  return rand()%(b-a+1)+a;
}
```

Если функция добавлена после функции main, не забудьте продублировать ее заголовок в начале программы.

При вызове функции в качестве параметров a и b используйте границы диапазона генерации. Например, получить псевдослучайное целое число из диапазона от -7 до 4 можно так:

```
irand(-7,4);
```

Задания

За каждое задание, если не сказано иначе, можно получить не более одного балла. Для решения задач **использовать функции**. Номер варианта можно узнать у преподавателя. Работы, выполненные по чужому варианту, не принимаются. Номер варианта остается постоянным для всех лабораторных работ.

Циклические алгоритмы:

1. Для введенного с клавиатуры n вычислить значение выражения:
$$P = \frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \ldots + \frac{1}{n\cdot(n+1)}$$
2. Написать программу для ввода числа из заданного диапазона. Программа запрашивает ввод числа до тех пор, пока не будет введено число, удовлетворяющее условиям (использовать цикл do).
3. Написать игру «Угадай число». Суть игры: пользователю предлагается вводить число из некоторого диапазона до тех пор, пока он его не угадает. Если пользователь вводит число, большее чем загаданное, то программа выводит сообщение «Перелет». Если пользователь вводит число, меньшее чем загаданное, то программа выводит сообщение «Недолет». Программа завершает свою работу, когда пользователь вводит загаданное число.
4. Найти НОД двух чисел при помощи алгоритма Евклида. Суть алгоритма:
    1. Если числа равны, алгоритм останавливается.
    2. Если первое число больше второго, то из первого вычитаем второе, возврат к п.1.
    3. Если второе число больше первого, то из второго вычитаем первое, возврат к п.1.

5. Вычислить с точностью eps значение y по следующей рекуррентной формуле (x и eps вводятся с клавиатуры) $y_n = \frac{1}{2}\left(y_{n-1} + \frac{x}{y_{n-1}}\right), y_1 = 1$
6. (2 балла) Дана монотонная последовательность, в которой каждое натуральное число k встречается ровно k раз: 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, ...
   По данному натуральному n выведите первые n членов этой последовательности.
   Схема выставления баллов:
   - Реализована функция, использующая два цикла for — 1 балл.
   - Реализована функция, использующая один цикл for — 2 балла.
7. (3 балла) См. стр. 44 – 52 Практикума. Задание необходимо выполнить полностью.
8. (3 балла) См. стр. 52 – 54 Практикума. Для понимания принципа решения задачи нужно использовать задачу 2.5 из Семинара 2 Практикума.

Рекурсия:

9. (2 балла) Написать программу, организующую вычисление числа Фибоначчи с номером n с использованием рекурсии. Считать, что Fib(0) = 0, Fib(1) = 1.
   Схема выставления баллов:
   - Реализована функция, возвращающая сумму результатов двух рекурсивных вызовов — 1 балл.
   - Реализованы две функции, рекурсивная с одним вызовом и нерекурсивная, вызывающая первую — 2 балла.
10. (3 балла) Решите следующую задачу двумя способами:
    a) с помощью циклического алгоритма.
    b) с использованием рекурсии.

    Схема выставления баллов:
    - Реализована функция, вычисляющая значение с помощью цикла **или** функция, вычисляющая значение с помощью рекурсии — 1 балл.
    - Реализована функция, вычисляющая значение с помощью цикла **и** функция, вычисляющая значение с помощью рекурсии — 3 балла.

    Вариант 1: Вычислить n-е значение $x = \sqrt{a}$ для заданного *a*, используя рекуррентное соотношение

    $$y_n = \frac{1}{2}\left(y_{n-1} + \frac{x}{y_{n-1}}\right), y_1 = 1$$

    Вариант 2: Вычислить n-е значение $x = \sqrt[3]{a}$ для заданного *a*, используя рекуррентное соотношение

    $$x_{n+1} = \frac{1}{3}\left(x_n + 2\sqrt{\frac{a}{x_n}}\right), x_0 = a$$

    .

    Вариант 3: Написать программу, находящую значение n-й степени числа x (x >= 0 – вещественное число, n – целое отрицательное число).

    Вариант 4: Вычислить с точностью eps число π по формуле

    $$\pi = 4(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - ...)$$

    Вариант 5: Написать программу вычисления двойного факториала для введенного n (n!!). Двойной факториал рассчитывается как произведение как всех натуральных чисел в отрезке [1,n], имеющих ту же чётность что и n:
    (2k)!! = 2*4*…*2k для четных n.
    (2k + 1)!! = 1*3*…*(2k+1) для нечетных n.

Подготовлено Латухиной Е.А. и Бурковым В.А.

Вариант 6: Вычислить с точностью eps число π по формуле

$$\pi = \sqrt{6(1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + ...)}$$

Вариант 7: Вычислить n-е приближение $y = \sqrt[3]{x}$ для заданного *x*, используя рекуррентное соотношение

$$y(x,n) = \begin{cases} 1, n = 0 \\ \frac{1}{3}\left(2y(x,n-1) + \frac{x}{y(x,n-1)^2}\right) \end{cases}.$$

Вариант 8: Написать программу, находящую значение n-й степени числа x (x >= 0 – вещественное число, n – целое неотрицательное число).

Вариант 9: Дано натуральное число n>0, десятичная запись которого не содержит нулей. Получите число, записанное теми же цифрами, но в противоположном порядке.

Вариант 10: Дано натуральное число n>1. Проверьте, является ли оно простым. Программа должна вывести слово YES, если число простое и NO, если число составное.

Вариант 11: Написать программу, которая переводит введенное число в двоичную систему счисления.

Вариант 12: С клавиатуры вводится последовательность целых чисел, завершающаяся числом 0. Выведите все нечетные числа из этой последовательности, сохраняя их порядок.

Вариант 13: Написать программу, которая переводит введенное число в заданную систему счисления (основание системы счисления вводится с клавиатуры).

Вариант 14: Для введенного с клавиатуры числа, состоящего из 2*n цифр, выяснить, является ли оно счастливым (сумма первых n цифр числа равна сумме оставшихся).

Вариант 15: Дана последовательность натуральных чисел, завершающаяся числом 0. Определите значение второго по величине элемента в этой последовательности. Гарантируется, что последовательность содержит хотя бы два числа (кроме нуля).

Вариант 16: Для натурального n вывести количество разных цифр, участвовавших в его записи.

Вариант 17: написать программу, которая будет вычислять для заданного целого n функцию Маккарти: $M(n) = \begin{cases} n - 10, & \text{если } n \geq 100 \\ M(n + 11) & \text{иначе} \end{cases}$

Вариант 18: написать программу, которая будет вычислять для заданных натуральных x, y, z функцию Кадью: $K(x,y,z) = \begin{cases} z, \text{если } x = y \\ K(x, y+1, (y+1)\cdot z) \text{иначе} \end{cases}$

Вариант 19: написать программу, находящую приближенное значение корня k-й степени из вещественного неотрицательного числа x по формуле:

$$y(x,k,n) = \begin{cases} 1, n = 0 \\ \frac{1}{k}\left((k-1)y(x,k,n-1) + \frac{x}{y(x,k,n-1)^{k-1}}\right) \end{cases}$$

Вариант 20: написать программу, которая будет вычислять для заданных целых m и n функцию Аккермана: $A(m,\ n) = \begin{cases} n+1, & m = 0; \\ A(m-1,\ 1), & m > 0,\ n = 0; \\ A(m-1,\ A(m,\ n-1)), & m > 0,\ n > 0. \end{cases}$

с использованием рекурсии. n – число приближений, вводится с клавиатуры.