УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите теоретические сведения к лабораторной работе, выполните практическое задание.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 13.03.2023.

Требования к отчету:

Отчет предоставляется преподавателю в электронном варианте и должен содержать:

- **название работы, постановку цели, вывод;**
- **ответы на контрольные вопросы, указанные преподавателем.**

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лабораторная работа № 8

Тема: «Работа с циклом FOR на языке С++»

Цель: изучить операторы цикла, научиться создавать программы и блок-схемы циклических структур.

Теоретические сведения

Программа начинает и заканчивает свою работу, но это не значит, что каждая инструкция в программе должна выполняться только однажды. Вы можете решить провести несколько циклов обработки данных или выполнить несколько различных вычислений. Вы можете также попросить пользователя вводить данные до тех пор, пока он не сделает это надлежащим образом.

Язык Си и Си++ имеет три структуры, известные под названием *циклов*, которые используются для управления повторами:

цикл for;

цикл do...while;

цикл while.

Любой из этих циклов может быть применен для повторения инструкции, группы инструкций или даже целой программы.

Использование цикла for

Цикл for используется в том случае, когда известно точное количество повторов, которое нужно выполнить. Структура такого цикла показана на рис.1. Обратите внимание на то, что точка с запятой ставится только после инструкции, а не после параметров for. Однако три параметра внутри круглых скобок отделяются друг от друга точкой с запятой.

В приведенной ниже программе цикл for используется для того, чтобы вывести на экран монитора числа от 1 до 10, расположенные друг под другом.

```
main()
        {
        int repeat;
        for (repeat = 1; repeat <= 10; repeat++)
                printf("%d\n", repeat);
        }
```

Этот цикл управляется переменной repeat, которая называется *индексом*. Индексу можно присвоить любое имя, но значение переменной обязательно должно быть целым числом. Выражение в круглых скобках после for делится на три составляющие:

repeat=1	инициализация переменной repeat путем присваивания ей начального значения
repeat <= 10	задает условие повтора цикла до тех пор, пока значение переменной repeat остается меньше или равно 10
repeat++	приращение значения переменной repeat после каждого повтора цикла

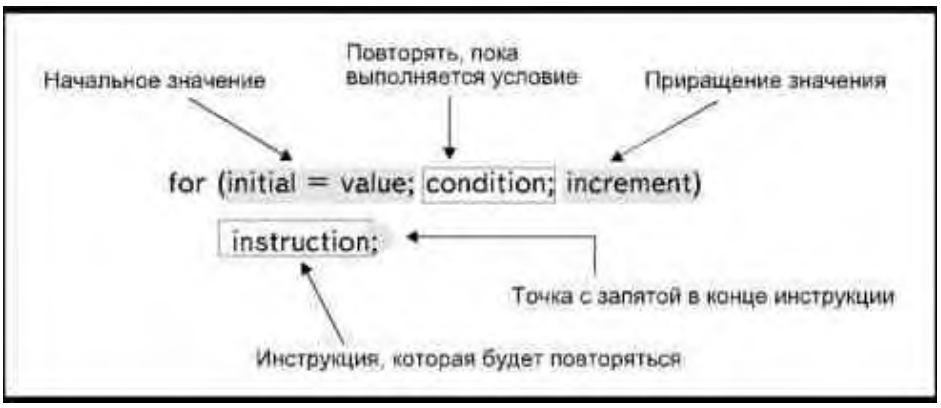

Рис. 1. Структура цикла for

Замечания по Си++

В языке Си++ можно определить также и тип индекса внутри круглых скобок цикла for:

for (int repeat=1; repeat <= 10; repeat++)

При каждом новом повторе цикла программа выводит на экран текущее значение переменной repeat.

Когда программа начнет выполнение цикла, она присвоит переменной repeat начальное значение, равное 1. Затем будет проверено, является ли истинным условие, что значение переменной меньше или равно 10. Если условие истинное, начнется выполнение инструкции, связанной с циклом, то есть вывод на экран значения переменной.

После выполнения инструкции произойдет увеличение значения переменной на единицу и снова будет проведена проверка истинности условия (рис.2). Так как условие все еще является истинным, цикл будет выполнен во второй раз, отображая на дисплее текущее значение переменной. Этот процесс будет повторяться до тех пор, пока значение переменной не вырастет до 11. Как только это произойдет, условие repeat <= 10 уже не будет истинным, так что выполнение инструкции прекратится и цикл завершится.

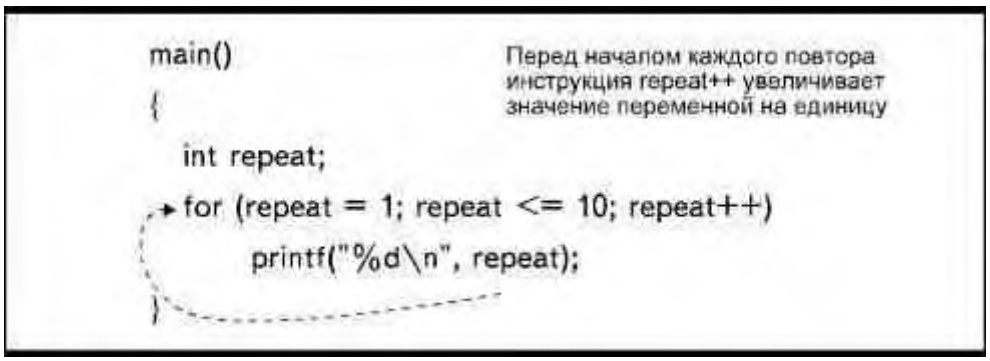

Рис. 2. Условие проверяется перед каждым повтором цикла

В предыдущем примере значение индекса использовалось непосредственно в инструкции вывода. В то же время можно написать инструкции следующим образом:

```
main()
  {
  int repeat;
  char letter;
  puts("Введите 10 символов");
  for(repeat = 1; repeat <= 10; repeat++)
          letter = getchar();
  }
```

В этой программе функция getchar() выполняется 10 раз, по количеству повторов цикла, пока значение переменной repeat не увеличится с 1 до 11. Индекс в данном случае используется только для определения количества повторов. С тем же результатом можно было записать инструкции следующим образом:

```
  for (repeat = 101; repeat <= 110; repeat++)
          letter = getchar();
  }
```

Здесь также вводится 10 символов, но теперь значение индекса изменяется от 101 до 110. Точное значение индекса приобретает значение только в том случае, когда оно само по себе используется в цикле.

Создание паузы в программе

Цикл for можно использовать и без инструкций, с целью создания задержки в программе:

for (delay = 1; delay <= 1000; delay++);

Хотя инструкции, связанные с циклом, отсутствуют, тем не менее, цикл будет повторен 1000 раз, пока выполняется указанное условие, то есть пока значение переменной delay не возрастет с 1 до 1001. Выполнение повторов цикла приостановит переход программы к выполнению следующих инструкций.

Одним из возможных применений такого цикла является временная остановка вывода на экран сообщений, с тем, чтобы дать пользователю время прочитать инструкции или подсказки. Этот способ можно использовать наряду с тем, где пользователю предлагается нажать Enter для продолжения вывода сообщений.

Составные инструкции

В одном цикле можно выполнить несколько инструкций. Для этого всю последовательность инструкций, относящихся к циклу, следует заключить в фигурные скобки. В качестве примера посмотрите программу перевода 101 последовательного значения температур (от 32 до 132) в значения по шкале Цельсия:

```
main()
  {
  int temp;
  float celsius;
          puts("Шкала Фаренгейта\tШкала Цельсия\n");
          for (temp = 32; temp <= 132; temp++)
  {
                  celsius = (5.0 / 9.0) * (temp - 32);
                  printf("%d\t\t%6.2f\n", temp, celsius);
```

```
                    }
  }
```

Функция printf() форматирует вывод и отображает значения на экране в колонках. Два символа табуляции (\t\t) позволяют согласовать вывод значения температуры с заголовком колонки. Указатель формата (%6.2) отображает значение с двумя знаками после точки, определяя значение ширины поля, равное шести символам.

При каждом повторе цикла выполняются две инструкции. Значение индекса определяет как количество повторов, так и значения, которые следует перевести в шкалу Цельсия. Сравните только что просмотренную вами программу со следующей:

```
main()
  {
  int temp, repeat;
  float celsius;
  puts("Шкала Фаренгейта\tШкала Цельсия\n");
  temp = 10;
  for (repeat = 1; repeat <= 10; repeat++)
          {
          celsius = (5.0 / 9.0) * (temp - 32);
          printf("%d\t\t%6.2f\n", temp, celsius);
          temp += 10;
          }
  }
```

Теперь индекс определяет только количество повторов. Значения температуры, которые следует преобразовать, определяет переменная temp. Эта переменная увеличивает свое значение на 10 при каждом повторе: с 10 до 20, 30 и так далее, вплоть до 100.

Использование переменных

Если во время составления программы вы не знаете, сколько повторов потребуется, вы все же можете использовать цикл for, при условии, что

количество повторов будет указано в момент запуска программы на выполнение. Можно ввести значение в переменную и использовать ее в условии. Например, следующая программа просит пользователя указать пределы значений температуры, которые требуется преобразовать, то есть, по сути, пользователь сам должен определить количество повторов цикла:

```
main()
  {
  int temp, start, end;
  float celsius;
        printf("Введите начальное значение температуры: ");
        scanf("%d", &start);
        printf("Введите конечное значение температуры: ");
        scanf("%d", &end);
        puts("Шкала Фаренгейта\tШкала Цельсия\n");
        for (temp = start; temp <= end; temp++)
              {
              celsius = (5.0 / 9.0) * (temp - 32);
              printf("%d\t\t%6.2f\n", temp, celsius);
              }
  }
```

Здесь требуется ввести начальное и конечное значения температур, которые мы хотим привести к шкале Цельсия. Переменные start и end используются в цикле for для установки начального значения индекса и для проверки условия. Цикл завершится, когда значение переменной temp превысит величину переменной end. Таким образом, если вы введете числа 20 и 43, программа преобразует значения температур от 20 до 43 градусов по Фаренгейту в соответствующие значения по шкале Цельсия. Цикл будет повторен 24 раза, затем завершится.

Вложенные циклы

Если один цикл for выполняется внутри другого, принято говорить, что мы имеем дело с вложенным циклом. Внутренний цикл целиком выполняется

во время каждого повторения внешнего цикла. Вложенные циклы for можно представить себе как двухмерные, а единичный — как одномерный.

В качестве иллюстрации рассмотрим следующую программу:

```
main()
    {
    int row, column;
    for (row = 1; row <= 10; row++)
            {
            for (column = 1; column <= 10; column++)
                    printf("*");
            putchar('\n');   /*вне внутреннего цикла, но внутри внешнего*/
            }
    }
```

Программа выводит на экран монитора 10 рядов, состоящих из 10 звездочек. Здесь используются две целочисленные переменные row и column. Во внешнем цикле переменная row увеличивает свое значение на единицу при каждом повторе. Кроме того, при каждом повторе внешнего цикла, внутренний цикл повторяется 10 раз, увеличивая значение переменной column и выводя на экран ряд из десяти звездочек (обратите внимание, что имена переменным даны с таким расчетом, чтобы пояснить логику программы: row по-английски значит «строка», а column — «столбец» или «колонка»).

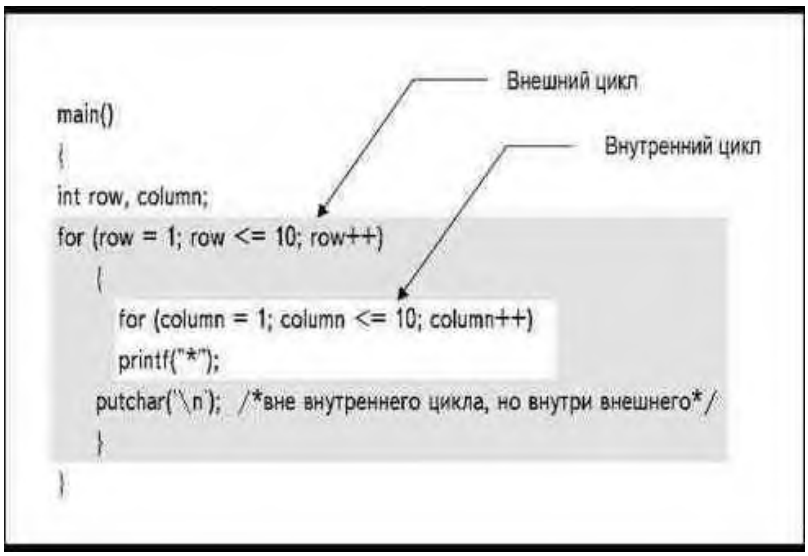

Рис. 3. Внешний и внутренний циклы

На рис. 3 продемонстрирована работа этих вложенных циклов. К инструкциям внутреннего цикла относится

for (column = 1; column <= 10; column++)

printf("");*

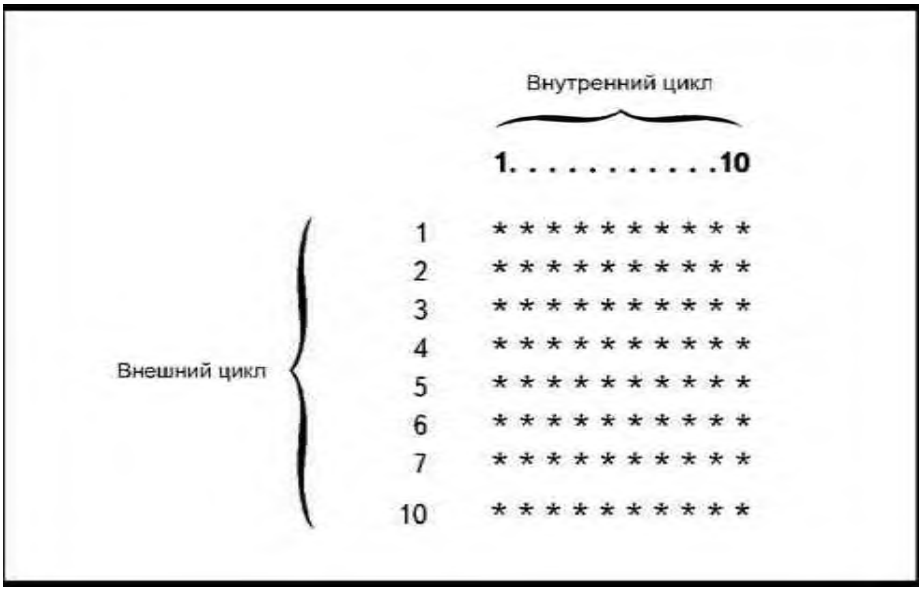

Рис. 4. Значения, которые имеют переменные во время каждого повтора цикла

В результате работы программы на экране появятся 100 звездочек: 10 внутренних циклов сформируют 10 колонок, а 10 внешних циклов — 10 рядов.

Значения, которые переменные приобретают при каждом повторе цикла, показаны на рис. 4.

Обратите внимание на положение инструкции putchar('\n');. Она не относится к внутреннему циклу, но в то же время находится внутри фигурных скобок, ограничивающих внешний цикл. Эта инструкция выполняется десять раз, по одному на каждый повтор внешнего цикла, вставляя код «новая строка» после каждого ряда звездочек.

В Листинге 1 приведен другой пример использования вложенных циклов. Десять внешних и десять внутренних циклов здесь используются для того, чтобы создать таблицу умножения. Вместо того чтобы просто выводить на экран ряды звездочек, программа выводит результаты произведения значения переменной row на значение переменной column.

Листинг 1. Программа создания таблицы умножения.

```
/*timestab.c*/
main()
  {
  int row, column;
  puts("\t\tТаблица Пифагора\n\n");
  for(row = 1; row <= 10; row++)
          {
          for(column = 1; column <= 10; column++)
                  printf("%6d", row * column);
          putchar('\n');
          }
  }
```

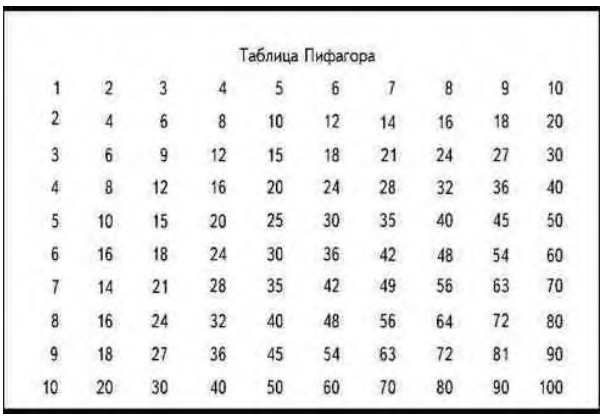

Таблица Пифагора

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	16	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Рис. 5. Результат работы программы, создающей таблицу умножения

Результат работы программы, названной нами TIMESTAB.C, изображен на рис. 5.

Наконец, рассмотрим следующую программу:

```
main()
    {
    int row, column;
    for (row = 1; row <= 10; row++)
            {
            for (column = 1; column <= row; column++)
                    printf("*");
            putchar('\n');
            }
    }
```

Она выводит на экран последовательность звездочек, показанную на рис.6.

Рис. 6. Количество повторов внутреннего цикла определяется номером повтора внешнего цикла

Мы видим, что каждый ряд звездочек имеет разную длину. Это обусловлено тем, что количество повторов внутреннего цикла не одинаково, а возрастает с каждым следующим выполнением внешнего цикла: одна звездочка в первом ряду, две звездочки во втором ряду, три в третьем и так далее. Количество колонок совпадает с номером ряда. Мы добились такого эффекта, используя переменную row в качестве условия внутреннего цикла.

При первом выполнении внешнего цикла внутренний цикл выполняется только один раз, выводя на экран одну звездочку. При втором повторе внешнего цикла внутренний цикл выполняется два раза, выводя две звездочки. В результате продолжения этого процесса получается узор из звездочек.

Будьте внимательны, когда пишете программу, содержащую два и больше вложенных циклов for. Если в программе указано 100 повторов внешнего и 100 повторов внутреннего цикла, это означает, что потребуется выполнить 10 тысяч повторов!

Задание к лабораторной работе:

1. Ознакомиться с теоретическим материалом.

2. Проверить свою теоретическую подготовку по контрольным вопросам.

3. В соответствии с вариантом составить блок-схему алгоритма и программу для вычисления значения выражения для z своего варианта (табл.1), если

$$a = \sum_{x=i}^{i+8} f_i(x), \quad b = \prod_{x=i}^{i+5} f_{i+1}(x)$$

где i - номер варианта, x - целое число. Выражения функций fi (x) и fi + 1 (x) определить в табл. 2.

Задачу решить тремя способами, используя операторы (задание 1 и 2):

1. Оператор while

2. Оператор for

3. Оператор do–while

4. Ввести программу в ЭВМ, вычислить значение, вывести и, a, b, z. Сделать выводы.

5. Оформить отчет по лабораторной работе работы.

Таблица 1. Задание 1

Вариант	**Значение**	**Вариант**	**Значение**	**Вариант**	**Значение**
1	$z = a + b$	**11**	$z = a - 2b$	**21**	$z = tg(a + b)$
2	$z = ab$	**12**	$z = atgb$	**22**	$z = \ln\lvert a + 4b\rvert$
3	$z = tg(b) - a$	**13**	$z = \cos(a + b)$	**23**	$z = 3ab - \cos b$
4	$z = (a + b)^2$	**14**	$z = \lvert a - b\rvert$	**24**	$z = 4a + e^b$
5	$z = 5ab - 4$	**15**	$z = ctg(2a) - b$	**25**	$z = 5a - 2b$
6	$z = \sin(a) + b$	**16**	$z = e^{3ab}$	**26**	$z = a^2 + 3b$
7	$z = btg(a)$	**17**	$z = 4ab - b$	**27**	$z = \sin(a^2) - b$
8	$z = a^2 + 3b$	**18**	$z = 2a - b$	**28**	$z = \cos^2(a + b)$

9	$z = (\lvert a+b\rvert)^{\frac{1}{4}}$	**19**	$z = \lvert 12a - \cos(b)\rvert$	**29**	$z = a^b + b$
10	$z = ab - \pi$	**20**	$z = a - b^2$	**30**	$z = a - b^a$

Таблица 2. Задание 1

i	**Функция** $f_i(x)$	i	**Функция** $f_i(x)$	i	**Функция** $f_i(x)$
1	$\sqrt[3]{\lvert x + \sin x^2\rvert} - 2$	**11**	$tg^4 x - x^2$	**21**	$\pi \cdot \lg\lvert x\rvert - 5$
2	$\ln\lvert x\rvert + e^{\sqrt{\lvert x-1\rvert}}$	**12**	$\sqrt{x} + \sqrt[3]{\lvert x\rvert}$	**22**	$1.8 \cdot \sqrt[3]{\lvert \sin x\rvert} + e^{\sqrt{\lvert x-1\rvert}}$
3	$x^2 - 4\cos x^2$	**13**	$x^3 + 2x \cdot e^x$	**23**	$\sqrt[3]{\lvert \sin x\rvert} + \ln x$
4	$\sqrt[3]{\lvert 2x\rvert + \sqrt{\lvert tgx\rvert}}$	**14**	$\sqrt{x^3 - x} + \sin x^2$	**24**	$\sin x^2 + tgx$
5	$2\pi + tgx^2$	**15**	$3x + \cos^4 x$	**25**	$x^3 + \cos^4 x$
6	$\left(x + 2x^2\right) \cdot \lvert \cos x^2\rvert$	**16**	$tgx^3 \cdot \sin x$	**26**	$2x^3 + \log_x 3$
7	$3x^2 + 2\cos x$	**17**	$x^3 - \sqrt{x} + \log_2 x$	**27**	$\left(\sin x + tg^2 x\right) \cdot e^{\lvert x\rvert}$
8	$\sqrt{x + 2x^2} \cdot \sin x$	**18**	$\left(\lvert \sin x^3\rvert + x\right) \cdot e^x$	**28**	$\left(\lvert \ln x^2\rvert + 3\right) \cdot \sqrt{x}$
9	$\sqrt[3]{\lvert x + \sin x^2\rvert}$	**19**	$tgx^3 - \sqrt[3]{x^2 - 3}$	**29**	$\pi \cdot \lvert \lg x\rvert - x^3$
10	$\sqrt{x^2 + x^3} \cdot tgx$	**20**	$e^{\sqrt{\lvert x-1\rvert}} \cdot \ln x$	**30**	$\sqrt[4]{\lvert \sin^2 x\rvert}$