Лабораторная работа №3

Циклические вычислительные процессы

Цель работы: изучить приемы составления циклических алгоритмов, правила использования операторов `for`, `while`, `do/while`, а также операторов перехода `continue` и `break`.

Краткие теоретические сведения

Операторы цикла используются для многократного повторения вычисления. Любой цикл состоит из тела цикла (операторы, которые выполняются несколько раз), начального значения управляющей переменной, проверки условия выполнения цикла и шага (изменения управляющей переменной). В языке C/C++ есть три оператора цикла — это `for`, `while` и `do/while`.

Оператор `while`, как и оператор `for`, называется оператором цикла с *предусловием*, так как истинность условия проверяется перед входом в цикл.

Оператор `do/while` называется оператором цикла с *постусловием* и используется в тех случаях, когда необходимо обеспечить выполнение цикла хотя бы один раз.

Структура повторения `for`

```
for (выражение_1; выражение_2; выражение_3) оператор;
```

где `выражение_1` — начальное значение управляющей переменной цикла;

`выражение_2` — проверка условия на продолжение цикла;

`выражение_3` — изменение параметра цикла (шаг);

`оператор` — это тело цикла, простой или составной оператор.

Схема работы оператора следующая: только один раз вначале вычисляется `выражение_1`, затем проверяется `выражение_2`, и если оно — «истина», то выполняется тело цикла, затем производится шаг управляющей переменной цикла, и так до тех пор, пока `выражение_2` не примет значение «ложь».

Например:

1. `for (k=l; k<10; k++) printf("%-4d", k);`
   В строке печатаются с выравниванием влево в поля шириной 4 цифры от 1 до 9.

2. `for(ch='a'; ch<='z'; ch++) printf("%4c", ch);`
   На экран выводятся буквы латинского алфавита.

3. Уменьшение параметра: `for (n=10; n>0; n--) оператор;`

4. Шаг с помощью арифметического выражения:
   `for (x=1; y<=75; y=5*(x++)+10) оператор;`

5. Использование несколько инициализирующих или корректирующих выражений:
   `for(x=l, y=0; x<10; x++, y += x) оператор;`

6. Бесконечный цикл `for ( ; ; ) оператор;`

Структуры повторения while и do/while

Основная форма циклического оператора `while`:

```
while (условие) оператор;
```

где `оператор` — это простой, составной или пустой оператор.

Цикл выполняется до тех пор, пока условие принимает значение «истина», т. е. выражение в скобках возвращает ненулевой результат.

Основная форма оператора `do/while`:

```
do
      оператор;
while (условие);
```

где `оператор` — это простой, составной или пустой оператор.

Оператор `do/while` — оператор цикла с постусловием, т. е. сначала выполняется оператор, а затем проверяется условие на истинность. Так как в цикле `do/while` условие проверяется в конце цикла, то цикл будет выполнен хотя бы один раз.

Вложенные циклы.

В случае вложенных циклов один цикл находится внутри другого. Внутренний цикл будет выполняться для каждого значения параметра i, удовлетворяющего условию внешнего цикла.

Пример:

```
int i, j;
   // Печать таблицы умножения
    for(i=2; i<10; i++)
    {     for(j=2; j<10; j++)
          printf("\n %d*%d=%2d", i, j, i*j);
          printf("\n");
    }
```

Операторы перехода

Операторы перехода выполняют безусловную передачу управления.

◇ *Оператор* `break` используется для принудительного выхода из циклов `for`, `while`, `do` и оператора `switch`.

◇ *Оператор* `continue` передает управление на следующую итерацию ближайшего цикла `while`, `do` или `for`. В циклах `while` и `do` это означает немедленную проверку условия, тогда как в структуре `for` сначала выполняется выражение приращения, а затем проверка условия продолжения цикла.

Примеры использования оператора `continue`:

1. В операторе `for`

```
for( int x = 1; x <= 10; x++)
   {  if(x = 5) continue;
      printf("%d",x);
   }
```

   Так как в структуре `for` сначала выполняется выражение приращения, а затем проверка условия продолжения цикла, то в результате в строку печатаются через пробел все цифры от 1 до 10, кроме 5.

2. В операторе `while`

```
//неправильный код
   int x = 1;
   while(x <= 10)
      { if( x = 5) continue;
        printf("%d", x);
        x++;
      }
```

Так как в структурах `while` и `do/while` после выполнения оператора `continue` производится проверка условия продолжения цикла, то в результате печатаются в строку через пробел все цифры от 1 до 4, а далее будет бесконечный цикл.

Чтобы избежать данной проблемы, в указанных циклах до оператора `continue` необходимо обеспечить шаг цикла.

```
//правильный код
   int x = 1;
   while(x <= 10)
      { if(x = 5){ x++; continue;}
           printf("%d", x);
           x++;
      }
```

◇ *Оператор* `return` — оператор возврата значений из вызываемой функции в вызывающую. Он всегда завершает выполнение функции и передает управление в точку ее вызова. Вид оператора:

`return выражение;`

`выражение` может отсутствовать, в этом случае в вызывающую функцию ничего не передается.

ПРИМЕРЫ РЕШЕНИЙ

1. Ввести два целых числа и найти сумму чисел между ними.

```
#include <stdio.h>
#include <math.h>
#include <conio.h>

void main(void)
{  int i, a, b, sum=0;
   printf("input two numbers: \n");
   scanf("%d%d", &a, &b);
      if(a>b){ i=a; a=b; b=i; } //упорядочиваем числа по возрастанию
   for(i=a+1; i<b; i++)
      { if(i>=0) { if(i!=a+1) printf("+ %d ", i);
                   else printf("%d", i);
                 }
        else { if(i!=a+1) printf("+ (%d)", i);
               else printf("(%d)", i);
```

```
            }
        sum+=i;
      }
   if(sum>=0) printf("= %d\n", sum);
   else printf("=(%d)\n", sum);
   getch()
}
```

2. Ввести число n и вычислить 2^n.

```
#include <stdio.h>
#include <conio.h>

void main(void)
{ int n, st=l, i;
     printf("Input degree: \n");
     scanf("%d", &n);
  for(i=1; i<=n; i++) st *= 2;
  printf("2^%d=%d\n", n, st);
  getch();
}
```

3. Ввести целое число и определить является ли данное число простым.

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
   int n, i, f=l;
   printf("Input number: ");
   scanf ("%d", &n);
   for(i=2; i<=n/2; i++)
      { if(!(n%i)){ f = 0;
                    break;
                  }
      }
   if(f==1) printf("This is a prime number\n");
      else printf("NO!!!!!\n");
   getch();
}
```

4. Посчитать среднее арифметическое всех вводимых чисел; ввод чисел завершается, когда вводим 99.

Используя оператор `while`:

```
#include <stdio.h>
#include <conio.h>
   void main(void)
{
   int x, kol=0;
```

```
   double sr=0;
   printf("Input number: ");
   scanf("%d", &x);
   while(x!=99)
        { kol++;
          sr+=x;
          printf("Input number: ");
          scanf("%d",&x) ;
        }
   if(kol) sr/=kol;
      printf("arithmetic mean = %.21f\n", sr) ;
      getch();
}
```

Используя оператор do/while:

```
#include <stdio.h>
#include <conio.h>
   void main(void)
{
   int x, kol=0;
   double sr = 0;
   do {
         printf("Input number: ");
         scanf("%d", &x);
         kol++;
         sr+=x;
      }
   while(x!=99);
   if(kol) sr/=kol;
   printf("arithmetic mean = %.21f\n", sr);
   getch();
}
```

Задачи для выполнения

1. Ввести целое неотрицательное число. Посчитать факториал данного числа.
2. Введите два числа. Первое число x — основание, второе число n — степень. Посчитать x^n.
3. Ввести натуральное N. Используя один цикл, найти сумму $1! + 2! + \cdots + N!$. Чтобы избежать целочисленного переполнения, проводить вычисления с помощью вещественных переменных.
4. Введите целые положительные числа A и B ($A < B$). Вывести все целые числа от A до B включительно, при этом каждое число должно выводиться столько раз, каково его значение (например, число 3 выводится три раза).
5. Ввести целое число и определить, является ли данное число совершенным или нет. Целое число является совершенным, если его сомножители, включая 1 (но не само число), в сумме дают это число. Например, 6 — это совершенное число, так как $6 = 1 + 2 + 3$.

6. Вывести все простые числа в интервале от 2 до 10000.

7. Посчитать: количество нечетных, отрицательных чисел и произведение всех положительных чисел, ввод чисел завершается, когда вводим 0.

8. Вводим числа, ввод чисел заканчивается 0. Посчитать, сколько раз вводилось каждое из следующих чисел: -10, 5, 25 и 100.

9. Программа, которая находит наименьшее из нескольких вещественных чисел. Предполагается, что первое введенное число задает количество последующих вводимых чисел.

10. Ввести целое число и посчитать сумму и произведение цифр данного числа.

11. Вводим попеременно числа и вводим символы (`+`, `-`, `*`, `/`, `=`). Ввод завершается при вводе символа `=`. Программа считает и выводит результат введенного выражения.

12. Дано целое число N (> 1). Вывести наименьшее из целых чисел K, для которых сумма $1 + 2 + ... + K$ будет больше или равна N, и саму эту сумму.

13. Разработать программу, которая должна определять заработную плату для каждого работника. Компания выплачивает каждому служащему почасовую зарплату за первые 40 часов работы и выплачивает в полуторном размере за все рабочие часы сверх 40. Дан список сотрудников, количество часов, отработанных ими, и почасовая ставка каждого сотрудника. Программа должна ввести эти данные для каждого сотрудника, распечатать и вывести на экран его суммарную зарплату.

ВАРИАНТЫ ЗАДАНИЙ

Значение аргумента x изменяется от a до b с шагом h. Для каждого x найти значения функции $Y(x)$, суммы $S(x)$ и $|Y(x) - S(x)|$ и вывести в виде таблицы. Значения a, b, h и n вводятся с клавиатуры. Так как значение $S(x)$ является рядом разложения функции $Y(x)$, то значения S и Y для данного аргумента x должны совпадать в целой части и в первых двух-четырех позициях после десятичной точки.

Работу программы проверить для $a = 0.1$; $b = 1.0$; $h = 0.1$; n выбрать максимально возможным!

1. $S(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k+1}}{(2k+1)!}, \quad Y(x) = \sin x.$

2. $S(x) = \sum_{k=1}^{n} (-1)^{k+1} \frac{\sin(kx)}{k}, \quad Y(x) = x/2.$

3. $S(x) = \sum_{k=0}^{n} \frac{\cos(k\pi/4)\, x^k}{k!}, \quad Y(x) = e^{x\cos(\pi/4)} \cos(x \sin(\pi/4)).$

4. $S(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k}}{(2k)!}, \quad Y(x) = \cos x.$

5. $S(x) = \sum_{k=0}^{n} \frac{\cos(kx)}{k!}, \quad Y(x) = e^{\cos x} \cos(\sin x).$

6. $S(x) = \sum_{k=0}^{n} \frac{2k+1}{k!}\, x^{2k}, \quad Y(x) = (1 + 2x^2)e^{x^2}.$

7. $S(x) = \sum_{k=1}^{n} \frac{\cos(k\pi/3)\, x^k}{k}, \quad Y(x) = -\frac{1}{2} \ln\left(1 - 2x\cos\frac{\pi}{3} + x^2\right).$

8. $S(x) = \sum_{k=1}^{n} (-1)^k \frac{\cos(kx)}{k^2}, \quad Y(x) = \frac{1}{4}(x^2 - \pi^2/3).$

9. $S(x) = \sum_{k=1}^{n} (-1)^{k+1} \frac{x^{2k+1}}{4k^2 - 1}, \quad Y(x) = \frac{1 + x^2}{2} \operatorname{arctg} x - \frac{x}{2}.$

10. $S(x) = \sum_{k=0}^{n} \frac{x^{2k}}{(2k)!}, \quad Y(x) = \operatorname{ch} x = \frac{e^x + e^{-x}}{2}.$

11. $S(x) = \sum_{k=0}^{n} \frac{k^2 + 1}{2^k \cdot k!} x^k, \quad Y(x) = \left(1 + \frac{x}{2} + \frac{x^2}{4}\right) e^{x/2}.$

12. $S(x) = \sum_{k=0}^{n} (-1)^k \frac{2k^2 + 1}{(2k)!} x^{2k}, \quad Y(x) = \left(1 - \frac{x^2}{2}\right) \cos x - \frac{x}{2} \sin x.$

13. $S(x) = \sum_{k=1}^{n} (-1)^k \frac{(2x)^{2k}}{(2k)!}, \quad Y(x) = 2(\cos^2 x - 1).$

14. $S(x) = \sum_{k=0}^{n} \frac{x^{2k+1}}{(2k+1)!}, \quad Y(x) = (e^x - e^{-x})/2.$

15. $S(x) = \sum_{k=1}^{n} (-1)^{k+1} \frac{x^{2k}}{2k(2k-1)}, \quad Y(x) = x \operatorname{arctg} x - \ln \sqrt{1 + x^2}.$