**Лабораторная работа**

**"Операторы цикла в языке программирования Си++"**

Цель работы: ознакомиться с циклическими алгоритмами и операторами, реализующими эти алгоритмы. Освоить особенности применения каждого оператора. Составить программы с использованием всех операторов цикла.

**1. Теоретические сведения**

**Оператор цикла while**

Описание: while (выражение) оператор;

Действие:

Выполняется оператор до тех пор, пока значение выражения в скобках истинно. Проверка значения выражения происходит перед каждым выполнением оператора. Когда значение выражения ложно, цикл while заканчивается. Если выражение ложно с самого начала, оператор не выполняется ни разу.

Комментарий:

Следует заметить, что после ключевого слова while и выражения, заключенного в круглые скобки, точка с запятой не ставится.

Оператор иногда называется телом цикла. В теле цикла должны выполняться действия, в результате которых меняется значение управляющего выражения. В противном случае можем получить бесконечный цикл.

Пример:

/\*Демонстрация цикла while\*/

#include <stdio.h>

main( )

{

int i=1

while (getchar()!=’R’) i++;

/\*оператор getchar() вводит любой символ с клавиатуры\*/

printf("Символ R %d-й",i);

}

Приведенная программа позволяет определить порядковый номер первой введенной буквы R в последовательности символов. Она показывает использование цикла while, в теле которого всего одна инструкция (i++ - увеличение значения целого числа i на единицу). Если запустить эту программу на выполнение и ввести последовательность символов, например: abFk!Rgm, то на экране появится строка: Символ R 6-й.

**Оператор цикла do-while**

Описание: do оператор while (выражение);

Действие: В операторе do-while тело цикла выполняется по крайней мере один раз. Тело цикла будет выполняться до тех пор, пока выражение в скобках не примет ложное значение. Если оно ложно при входе в цикл, то его тело выполняется ровно один раз.

Комментарий: после слова while и выражения, заключенного в скобки, ставится точка с запятой. Если в теле цикла содержится более одной инструкции, то операторы цикла заключаются в фигурные скобки.

Пример:

/\*Демонстрация цикла do-while \*/

#include <stdio.h>

main()

{

int i=0; /\*i=0, а не единице\*/

do i++; while (getchar()!=’R’);

printf("Символ R %d-й",i);

}

Программа, представленная выше, теперь написана с циклом do-while. Результат программы будет таким же.

**Оператор цикла for**

Описание: for (выражение 1; выражение 2; выражение 3) оператор;

Действие:

В круглых скобках содержится три выражения. Первое из них служит для инициализации счетчика. Она осуществляется только один раз – когда цикл for начинает выполняться. Второе выражение необходимо для проверки условия, которая осуществляется перед каждым возможным выполнением тела цикла. Когда выражение становится ложным, цикл завершается. Третье выражение вычисляется в конце каждого выполнения тела цикла, происходит приращение числа на шаг.

Комментарий: в операторе цикла for точка с запятой после закрывающейся круглой скобки не ставится. Любое из трех или все три выражения в операторе могут отсутствовать, однако разделяющие их точки с запятыми опускать нельзя. Если отсутствует выражение 2, имеем бесконечный цикл. Например:for (scanf("%d",&p);;p++) оператор;

В языке СИ предусмотрены две нетрадиционные операции: (++) – для увеличения на единицу и (--) – для уменьшения на единицу значения операнда. Операции ++ и -- можно записывать как перед операндом, так и после него. В первом случае (++n или --n) значение операнда (n) изменяется перед его использованием в соответствующем выражении, а во втором (n++ или n--) – после его использования.

Если отсутствуют выражения 1 и 3, цикл становится эквивалентным while. Например:for (;a<20;) оператор;

Каждое из выражений может состоять из нескольких выражений, объединенных операцией "запятая". Например: for(i=0, j=1; i<100; i++, j++) a[i]=b[j];

Тело цикла заключается в фигурные скобки, если в нем более одного оператора.

Пример:

/\*демонстрация цикла for\*/

#include <stdio.h>

main()

{int i,j=1,k;

for (i=1;i<=3;i++)

printf("Минск\t");

/\*В цикле for три раза выполняется функция вывода\*/

/\*Здесь i-управляющая переменная цикла\*/

printf("\nУкажите число повторений цикла\n");

scanf("%d",&k);

for (i=1;i<=k;i++)

{j\*=i;

printf("%d",j);}

/\*Здесь две инструкции (более одной), поэтому они заключаются в фигурные скобки\*/

j=i;

printf("\n");

/\*Переменной j присваивается значение 1 и осуществляется перевод курсора\*/

/\*В следующем цикле for выполняются те же действия, что и в предыдущем\*/

for (i=1;i<=k;i++) printf("%d ", j\*=i);

}

Результаты выполнения программы следующие:

Минск Минск Минск

Укажите число повторений цикла; 5

1 2 6 24 120

1 2 6 24 120

**Оператор break**

Описание:

Break используется для прекращения выполнения цикла из-за обнаружения ошибки, для организации дополнения к условию в заголовке цикла, для прекращения бесконечного цикла.

Пример:

while (st>0 && st<25)

{

if st==4||st==8||st==12)

break;

}

Работа цикла полностью прекращается, как только условие в операторе if становится истинным.

**Оператор continue**

Действие: Этот оператор может использоваться во всех трех типах циклов. Как и в случае оператора break, он приводит к изменению характера выполнения программы. Однако вместо завершения работы цикла наличие оператора continue вызывает пропуск "оставшейся" части итерации и переход к началу следующей.

Пример. Заменим в предыдущей программе оператор break на continue.

while (st>0 && st<25)

{

if (st==4||st==8||st==12)

continue;

}

При истинном условии в операторе if оператор continue вызывает пропуск идущих за ним операторов тела цикла и осуществляется переход к началу следующей итерации.

## 2. Задание

1. Задание взять из таблицы 1 и таблицы 2 согласно варианту.
2. Разработать блок-схемы алгоритма.
3. Написать и отладить программы.

Таблица 1

|  |  |
| --- | --- |
| № варианта | Задание |
| 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. |  |

### Таблица 2

|  |  |
| --- | --- |
| № варианта | Задание |
| 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. |  |

Литература

1. Подбельский В.В. Язык Cu ++: Учебное пособие. - М.: Финансы и статистика,1995, - 560 с.
2. Страуструп Б. Язык программирования Сг ++. - М.: Радио и связь, 1991. - 352 стр.
3. Собоцинский В.В. Практический курс Turbo Cu ++. Основы объктно- ориентированного программирования. - М.: Свет, 1993. - 236 с.
4. Романов В.Ю. Программирование на языке Cu ++. Практический подход. - М.: Компьтер, 1993. - 160 с.
5. Уинер Р. Язык турбо Cu . - М.: Мир, 1991. - 384 с.
6. Юлин В.А., Булатова И.Р. Приглашение к Cu. - Мн.: Высш. Шк., 1990,- 224 с.
7. Котлинская Г.П., Галиновский О.И. Программирование на языке Cu. -Мн.: Высш. Шк., 1991. - 156 с.