Тема 2. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

Цель занятия: овладение практическими навыками разработки, программирования вычислительного процесса разветвляющейся структуры; знакомство с задачами, для решения которых используются операторы ветвления.

Теоретические сведения

Составной оператор. К составным операторам относят собственно составные операторы и блоки. В обоих случаях это последовательность операторов, заключенная в фигурные скобки. *Блок отличается от составного оператора наличием определений в теле блока.*

```
{
n++;            // это составной оператор
summa+=n;
}
{
int n=0;
n++;            // это блок
summa+=n;
}
```

Блок представляет собой последовательность инструкций, включая, возможно, объявления, заключенных в фигурные скобки. Общий вид ***блока:***

{
[объявления]
…
инструкция [инструкция]
…
}

В конце блока точка с запятой не ставится.

Объявления в блоке могут располагаться не обязательно перед исполнительными инструкциями. Объявление локальной переменной можно разместить в любом месте программы перед первым обращением к этой переменной.

Блоки инструкций могут быть вложены друг в друга на любую глубину. На переменную, объявленную внутри блока, можно ссылаться только внутри этого или внутри вложенных блоков (если эта переменная не скрыта переменной с таким же именем внутри вложенного блока).

Выполнение блока заключается в последовательном выполнении составляющих его инструкций, включая вложенные в него блоки:

Оператор ветвления if. Оператор ветвления **if** иначе называют ***условным*** оператором. Он имеет две формы: полную и краткую.

Полная *форма* оператора ветвления выглядит так:

if (<логическое выражение>) <оператор 1>;
else <оператор 2>;

Действие оператора. Если <логическое выражение> истинно, то выполняется <оператор 1>, в противном случае – <оператор 2>. Операторы 1 и 2 могут быть простыми или составными.

Алгоритмическая структура, соответствующая этому оператору, показана на рис.2.1.

Пример. Пусть x = 9. Тогда в результате выполнения операторов:

if (x >7) y= x*x; else y= sin(x);

if (x<5) z = exp(x); else z = sqrt(x);

получим y = 81, z = 3.

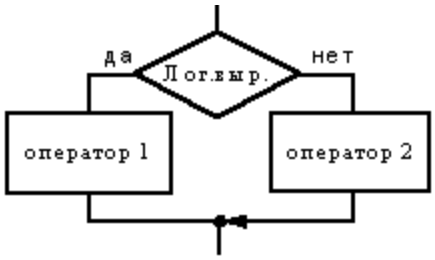

Рис. 2.1. Алгоритмическая структура выполнения полной формы оператора ветвления

Пример. Пусть a = 5. Тогда в результате выполнения операторов:

if *(a < 7)*

*{ b= a – 2;   c= 1 + 2 *a; }*

else

{ b = 2 + 5 a; c = 12 – 4 *(a – 3); }*

получим *b* = 3, *c* = 11.

Задание 2.1. Вычислите значение *b* и *c*, если *a* = 8.

Задача 2.1. Вычислить и вывести значения функции *y* в некоторой заданной точке *x*, если

$$y=\begin{cases}\ln|x|, & x<-1,\\ \sin(x), & -1\le x<1,\\ \cos(x), & x\ge 1.\end{cases}$$

Схема алгоритма решения представлена на рис. 2.2.

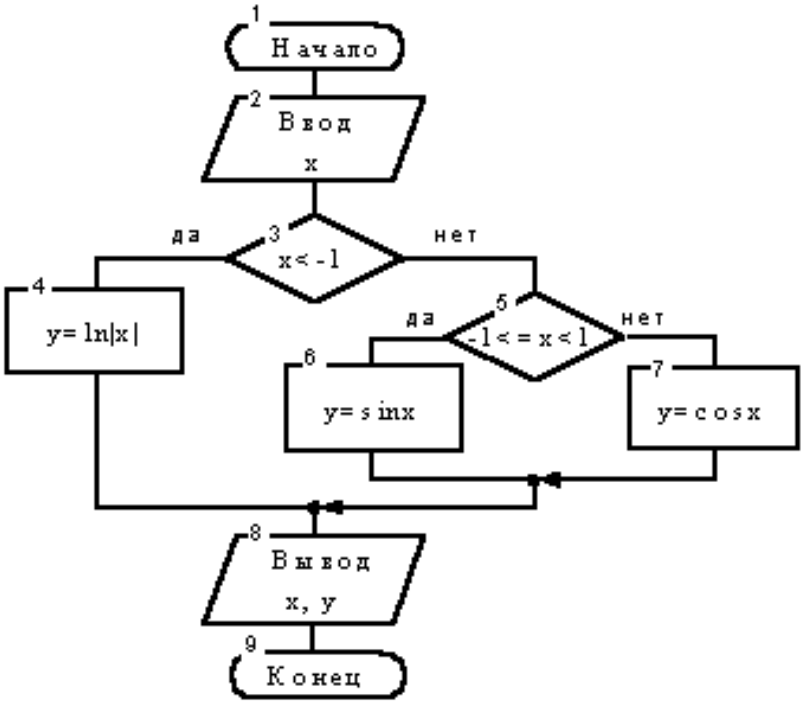

Рис. 2.2. Схема алгоритма решения задачи 2.1

Программа имеет вид:

```
#include <iostream>
#include <math.h>
#include <conio.h>    // файл, где определена функция getch()
using namespace std;
void main()
{
        double x,y;
        cout<<"\nEnter x";
        cin>>x;
         if (x<-1)  y = log(fabs(x));
          else
                  if (x>= -1 && x< 1)  y = sin(x);
                          else y = cos(x);
cout<<"\n"<<"x ="<< x<< "   y ="<< y;
getch();          // ждать нажатия любой клавиши
}
```

Краткая форма оператора ветвления **if** выглядит так:

if (< логическое выражение >) <оператор 1>;

Действие оператора. Если <логическое выражение> истинно, то выполняется <оператор 1>, иначе выполняется оператор, который находится после данной конструкции.

Алгоритмическая структура, соответствующая этому оператору, показана

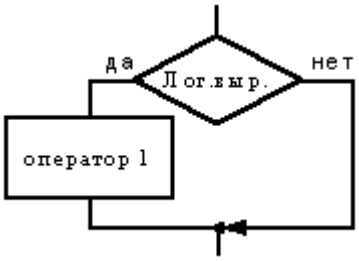

Рис. 2.3. Алгоритмическая структура выполнения краткой формы оператора ветвления

Пример. Пусть x = 25. Тогда в результате выполнения операторов
if (x > 12) y = 2 * x; z = 10;
if (x < 5) z = x / 2;
получим y = 50**,** z = 10.

Задание 2.2. Решить задачу 2.1, используя краткую форму условного оператора.

Задача 2.2. Значения переменных X, Y, Z поменять местами так, чтобы они оказались упорядоченными по возрастанию.

Программа имеет вид:

```
#include <iostream>
#include <windows.h>
#include <conio.h>      // файл, где определена функция getch()
```

```
using namespace std;
int main()
{
  SetConsoleOutputCP(1251);
  cout<<"Введите X, Y, Z: ";
  int X, Y, Z;
  cin>>X>>Y>>Z;
  if(Y < X)
  {       int tmp = X;
          X = Y;
          Y = tmp;
  }
  if(Z < X)
  {       int tmp = Z;
          Z = Y;
          Y = X;
          X = tmp;
  }
  if(Z < Y)
  {       int tmp = Y;
          Y = Z;
          Z = tmp;
  }
cout<<"Упорядоченные по возрастанию числа: "<<X<<" "<<Y<<" "<<Z<<'\n';
getch();  // ждать нажатия любой клавиши
  return 0;
}
```

Задание 2.3. Составьте схему алгоритма задачи 2.2.

Задача 2.3. Написать программу, которая определяет, является ли введенное число – счастливым (Счастливым считается шестизначное число, у которого сумма первых 3 цифр равна сумме вторых трех цифр). Если пользователь ввел не шестизначное число, сообщить об ошибке.

Программа имеет вид:

```
#include <iostream>
#include <clocale>
#include <conio.h>      // файл, где определена функция getch()
using namespace std;
void main()
{
  setlocale (LC_CTYPE,"rus");
  int num, d1,d2,d3,d4,d5,d6;
  cout<<"Введите шестизначное число = ";
  cin>>num;
  if (num/1000000!=0||num/100000==0)
          cout<<"Вы ввели не шестизначное число\n";
  else
  {
          d1=num/100000;
          d2=num/10000%10;
```

```
        d3=num/1000%10;
        d4=num/100%10;
        d5=num/10%10;
        d6=num%10;
        if (d1+d2+d3==d4+d5+d6)
                cout<<"Число счастливое\n";
        else
                cout<<"Число несчастливое\n";
  }
getch();  // ждать нажатия любой клавиши
}
```

Оператор goto. Оператор goto – это оператор безусловного перехода. Он предназначен для изменения последовательности выполнения операторов в программе путём передачи управления некоторому оператору с меткой. Оператор имеет вид:

goto < метка >;

Метка может стоять перед любым оператором в программе. Она отделяется от оператора двоеточием (:)

< метка > : <оператор>;

Оператор goto передает управления оператору, стоящему после метки. Использование оператора goto оправдано, если необходимо выполнить переход из нескольких вложенных циклов или переключателей вниз по тексту программы или перейти в одно место функции после выполнения различных действий.

Нельзя передавать управление внутрь операторов *if, switch* и *циклов*. Нельзя переходить внутрь блоков, содержащих инициализацию, на операторы, которые стоят после инициализации.

Задача 2.4. Ввести несколько вариантов значений коэффициентов квадратного уравнения $ax^2+bx+c=0$, $a\neq0$. Вывести сообщение о наличие действительных корней для каждого варианта [3].

Программа имеет вид:

```
#include "stdafx.h"
#include <iostream>
#include <clocale>
#include <conio.h>      // файл, где определена функция getch()
using namespace std;
void main()
{
setlocale (LC_CTYPE,"rus");
double a, b, c, d;
mes:cout<<"Введите коэффициенты квадратного уравнения = ";
cin>>a>>b>>c;
if (a == 0)    { cout<<"Данное уравнение не является квадратным\n"; goto finish; }
d=b*b-4*a*c;
if (d >=0)  cout<<"Данное уравнение имеет действительные корни\n";
   else
        {
          cout<<"Данные введены некорректно'\n";
```

```
            cout<<"Уравнение действительных решений не имеет\n";
            goto mes;
        }
 finish: getch();          // ждать нажатия любой клавиши
}
```

Оператор выбора switch.

Инструкция **switch** предназначена для организации выбора из множества различных вариантов.

Формат инструкции следующий:

switch (*выбирающее выражение*)
{
объявления
case *константа 1:*
последовательность инструкций 1
case *константа 2:*
последовательность инструкций 2
…
case *константа N:*
последовательность инструкций N
default:
последовательность инструкций
}

Схема выполнения инструкции ***switch*** следующая:

- вычисляется выбирающее выражение в круглых скобках;
- вычисленное значение последовательно сравнивается с константами, следующими за ключевыми словами ***case***;
- если одна из констант совпадает со значением выражения, то начинает выполняться последовательность инструкций, следующая за соответствующим ключевым словом ***case***;
- если ни одна из констант не равна выражению, то начинает выполняться последовательность инструкций, следующая за ключевым словом ***default,*** а в случае его отсутствия управление передается на следующую после ***switch*** инструкцию.

Обычно в качестве константного выражения используются целые или символьные константы.

Все значения константных выражений в инструкции switch должны быть уникальны. Кроме последовательностей инструкций, помеченных ключевым словом case, может быть, но обязательно одна, последовательность с начальным ключевым словом default с двоеточием.

Каждая из последовательностей инструкций может быть пустой, либо содержать одну или более инструкций; не требуется заключать эти последовательности инструкций в фигурные скобки.

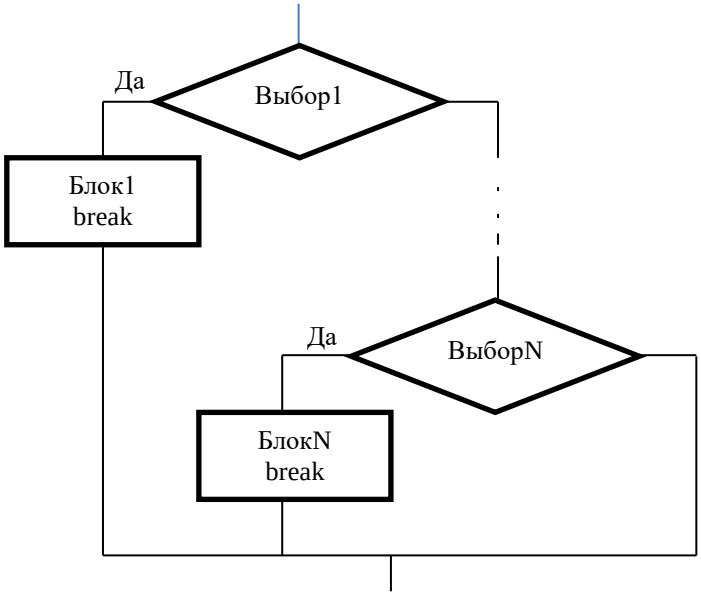

Рис. 2.4. Алгоритмическая структура выполнения оператора ***switch***

В инструкции ***switch*** можно использовать свои локальные переменные, объявления которых должны находиться перед первым ключевым словом ***case;*** однако в объявлениях не должна использоваться инициализация.

Все последовательности инструкций, следующие за ***case***, должны заканчиваться одной из инструкцией: ***break, return*** *или* ***goto***; как правило, используется ***break***.

Инструкция возврата return ***завершает выполнение функции, в которой она находится***, и возвращает управление в вызывающую функцию, в точку, непосредственно следующую за вызовом. Функция main() при выполнении инструкции return возвращает управление операционной системе.

Формат инструкции:

return ***выражение;***

Инструкция прерывания break ***обеспечивает прекращение выполнения самой внутренней из объемлющих её конструкций switch, while, do, for***. После выполнения инструкции **break** управление передается инструкции, следующей за объемлющей её конструкций.

Формат инструкции:

break;

Задача 2.5. Клиент покупает билеты в количестве N штук в определенную зону (A B C D), по 20, 30, 50 и 100 руб. соответственно, если клиент купит, больше чем на 2000 руб, то скидка 3%, больше 5000 – 5%.

Программа имеет вид:

```
#include <iostream>
#include <clocale>
#include <conio.h>      // файл, где определена функция getch()
using namespace std;
void main()
{
setlocale (LC_CTYPE,"rus");
char a;   int pb, numb;  double oplata, disc=0;
cout<<"В какую зону хотите купить билеты? A,B,C,D?";
cin>>a;
```

```
        if (a=='a'||a=='A'||a=='b'||a=='B'||a=='c'||a=='C'||a=='d'||a=='D')
        {switch (a)
          {case 'a':
           case 'A':
                    cout<<"Вы выбрали зону A\n";
                    pb=20;
                    break;
          case 'b':
          case 'B':
                    cout<<"Вы выбрали зону B\n";
                    pb=30;
                    break;
          case 'c':
          case 'C':
                    cout<<"Вы выбрали зону C\n";
                    pb=50;
                    break;
          case 'd':
          case 'D':
                    cout<<"Вы выбрали зону D\n";
                    pb=100;
                    break;
        }
        cout<<"Сколько вы хотите купить билетов? ";
        cin>>numb;
        if (numb*pb>=5000)
        {disc=numb*pb*0.05; cout<<"Ваша скидка = "<<disc;}
        else
          if (numb*pb>=2000)
          {disc=numb*pb*0.03; cout<<"Ваша скидка = "<<disc;}
            else {disc=0; cout<<"Ваша скидка = "<<disc;}
        cout<<"\nВнесите оплату в размере = "<<numb*pb-disc<<"руб.\n";
        cin>>oplata;
        if (oplata>=numb*pb-disc)
        cout<<"Ваша сдача = "<<oplata-(numb*pb-disc)<<" руб."<<"\nВы выбрали зону =
"<<a<<"\nКоличество билетов = "<<numb<<"\nСтоимость билетов = "<<numb*pb-disc<< "
руб.\n";
          else
                    cout<<"Не хватает "<<numb*pb-disc-oplata<<" руб. для оплаты\n";
      }
        else
        cout<<"Нет такой зоны\n";
        getch();  // ждать нажатия любой клавиши
      }
```

Методические указания

1. При подготовке к занятию необходимо изучить операторы: составной, условный, варианта;
2. В задачах, проверяющих правильность ветвей программы, или при отсутствии решения должны быть напечатаны соответствующие тексты.

Аудиторные и домашние задания

1. Даны целые числа *k, m, n*. Верно ли, что все три числа четные.
2. Известно, что из четырех чисел a_1, a_2, a_3, a_4 одно отлично от трех других, равных между собой; присвоить номер этого числа переменной *n*.
3. Даны *x, y, z*. Найти $\min\left(\sqrt{x+y+z}, 1.5xyz, 0.1\frac{xy}{z}\right)$.
4. На плоскости *X0Y* задана своими координатами точка *A*. Указать, где она расположена (на какой оси или в каком квадранте).
5. Известны два расстояния: одно в километрах, другое в футах. Какое из расстояний меньше? (1 фут = 0.0003048 км)
6. Составить программу, осуществляющую перевод величин из радиан в градусы и наоборот. Программа должна запрашивать, какой перевод нужно осуществить, и выполнять указанное действие.
7. Для заданных чисел *x* и *y* вычислить $z = \begin{cases} \max(x, y) & \text{при } x < 0; \\ \min(x, y) & \text{при } x \geq 0. \end{cases}$
8. Даны целые числа *k, m*, вещественные числа *x, y, z*. При $k<m$, $k=m$ или $k>m$ заменить модулем соответственно значения *x, y* или *z*, а два других значения уменьшить на *0.5*.
9. Дано трехзначное целое число *k*. Проверить, входит ли цифра *5* в запись числа *k*.
10. Определить, равен ли квадрат заданного трехзначного числа кубу суммы цифр этого числа.
11. Определить, является ли заданное четырехзначное число палиндромом (перевёртышем), например, числа 2222, 6116, 0440 и т.д.
12. Даны три ненулевых целых числа. Определить могут ли они представлять стороны прямоугольного треугольника.
13. Даны произвольные числа *a, b* и *c*. Если нельзя построить треугольник с такими длинами сторон, то напечатать 0, иначе напечатать 3, 2 или 1 в зависимости от того, равносторонний это треугольник, равнобедренный или какой-либо иной.
14. Даны *x, y*. Проверить, лежит ли точка с координатами *(x, y)* в кольце с радиусами r_1 и r_2. Определить, какой из радиусов больше. Центры окружностей совпадают с началом координат.
15. Дано натуральное число *n* ($n \leq 100$), определяющее возраст человека (в годах). Дать для этого числа наименования «год», «года» или «лет»: например, 1 год, 23 года, 45 лет и т.д.
16. Составить программу, позволяющую получить словесное описание оценок (2 – «неудовлетворительно», 3 – «удовлетворительно», 4 – «хорошо», 5 – «отлично»).
17. Написать программу, которая анализирует данные о возрасте и относит человека к одной из четырех групп: дошкольник, ученик, работник, пенсионер. Возраст вводится с клавиатуры.
18. К финалу конкурса лучшего по профессии «Специалист электронного офиса» были допущены трое: Иванов, Петров, Сидоров. Соревнования проходили в три тура. Иванов в первом туре набрал m_1 баллов, во втором – n_1, в третьем – p_1. Петров – m_2, n_2, p_2 соответственно; Сидоров – m_3, n_3, p_3. Составить программу, определяющую, сколько баллов набрал победитель.
19. Определить, попадет ли точка с координатами (x, y) в заштрихованную область (рис.2.5).

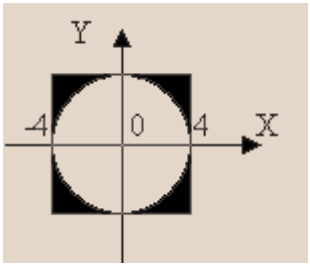

Рис.2.5. Границы области для задачи №19

Контрольные вопросы

1. Что такое логическое выражение? Какие значения оно может принимать?
2. Какие операторы называются простыми?
3. Что такое составной оператор?
4. Поясните назначения и правила использования оператора **goto**.
5. Что такое разветвляющийся вычислительный процесс?
6. Как выполняется условный оператор **if**?
7. Какие особенности существуют при написании вложенных операторов **if**?
8. Какой оператор позволяет выполнить одно из нескольких действий в зависимости от результата вычисления выражения?