Практическое занятие №3

Логические операторы. Операторы сравнения. Тернарный оператор.

Условный оператор if(). Условный оператор switch().

Дополнение:

- Для округления числа до указанной точности можно использовать следующие конструкции:

- 1. Добавляем препроцессорную директиву `#include <iomanip>`

```
double x; // объявляем переменную x
x = 25.3434568678456245 6; // присваиваем ей какое-либо значение
cout << setprecision(6) << fixed << x << endl; // используя данную конструкцию получаем на выходе число:
```

25.343457

Число (6) после манипулятора setprecision означает количество знаков после запятой, манипулятор fixed служит для указания того, что округлять числа нужно именно **после запятой**.

Без использования манипулятора fixed, данная программа выдаст число **25.3434 –** которое в сумме состоит из 6 цифр

Дополнение:

2. Также для этой цели можно использовать стандартный оператор вывода языка С – printf.

В этом случае при выполнении следующих инструкций:

```
double x;
   x = 25.3434568678456246;
   printf("%5.4f \n", x);
```

Программа выдаст число: 25.3434

Результат использования модификатора точности (%5.4f) зависит от типа модифицируемой команды форматирования. Чтобы использовать модификатор точности, надо поместить десятичную точку и точность вслед за ней после количества выводимых десятичных знаков.

Например, %5.4f означает вывод числа **шириной минимум 5 символов** с четырьмя знаками **после** точки.

Логические операторы

- Логические операторы предназначены для работы с операндами логического типа и результатом соответствующих операций являются значения логического типа. В C++ всего три логических оператора, представленных в таблице 1.4

Таблица 1.4. Логические операторы C++

Оператор	Назначение
`&&`	Логическое *И*. Бинарный оператор. Результатом выражения `A&&B` является `true`, если оба операнда `A` и `B` равны `true`. Результатом выражения `A&&B` является `false`, если хотя бы один из операндов `A` или `B` равен `false`
`||`	Логическое *ИЛИ*. Бинарный оператор. Результатом выражения `A||B` является `true`, если хотя бы один из операндов `A` или `B` равен `true`. Результатом выражения `A||B` является `false`, если оба операнда `A` и `B` равны `false`
`!`	Логическое отрицание. Унарный оператор. Результатом выражения `!A` является значение `true`, если операнд `A` равен `false`. Если операнд `A` равен `true`, значение выражения `!A` равно `false`

Операторы сравнения

- Операторы сравнения используются для сравнения значений операндов. Результатом выражения на основе оператора сравнения является логическое значение: true, если соответствующее условие выполнено, и false в противоположном случае. Операторы сравнения перечислены в таблице 1.5

Таблица 1.5. Операторы сравнения C++

Оператор	Назначение
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно
==	Равно
!=	Не равно

Тернарный оператор

- В C++ есть тернарный оператор (у оператора три операнда), который позволяет в зависимости от некоторого условия (первый операнд) выполнять различные действия (второй и третий операнды).
- Синтаксис вызова оператора таков: `условие?выражение1:выражение2`.
- Фактически тернарный оператор представляет собой сокращенную форму условного оператора.

Пример

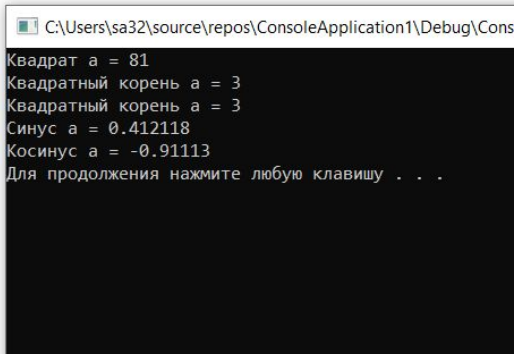
```
int main() // Главная функция программы
{

        int n;
        double x;
        cout << "Enter n = ";
        cin >> n;
        x = n > 0 ? 5.4 : 3.2; /* n > 0 - условие, которое необходимо проверить. Если условие истинно,
                                  то переменной x будет присвоено значение 5.4, если ложно, то 3.2 */
        cout << "x = " << x << "\n";


    system("pause"); // Задержка консольного окна
    return 0; // Значение, возвращаемое главной функцией
}
```

Примеры решения задач

- Средняя скорость движения мотоциклиста
- Решим задачу о вычислении средней скорости движения мотоциклиста на
- участке от пункта А до В через пункт Б, если расстояние между пунктами
- А и Б составляет S_1 , а расстояние между пунктами Б и В равно S_2 . Время
- движения мотоциклиста между пунктами А и Б равно t_1, а время движе-
- ния между пунктами Б и В равно t_2 . Средняя скорость определяется как
- $V = (S_1 + S_2)\ (t_1 + t_2)$. Параметры S_1 , S_2 , t_1 и t_2 вводятся пользовате-

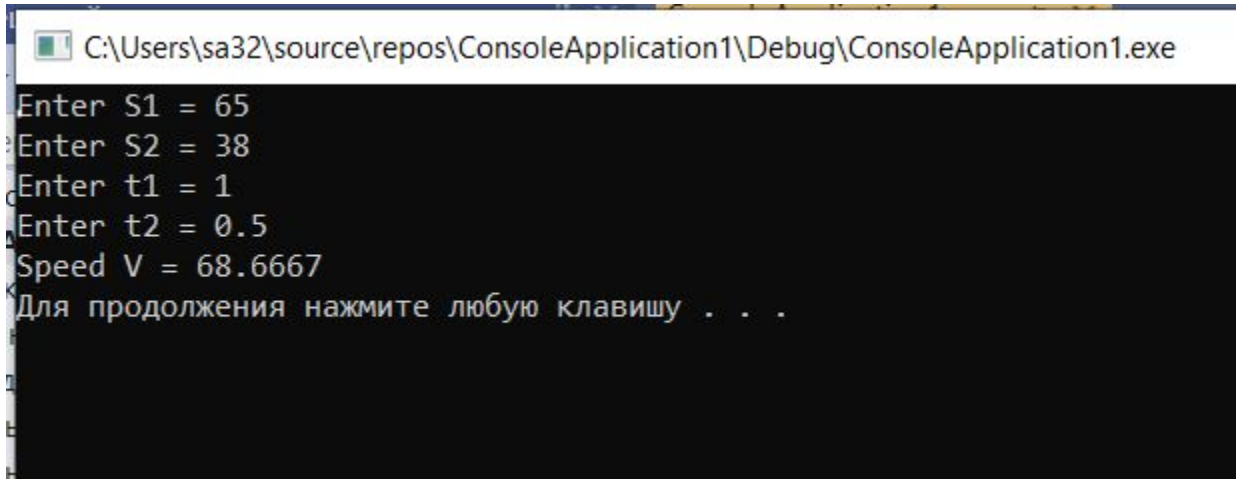
Примеры использования математических конструкций

```
int main() // Главная функция программы
{
    setlocale(LC_ALL, "Russian"); // Изменение кодировки консоли
        int n;
        double a,x,y,z,sinus,cosinus;

        a = 9;

        x = pow(a,2); // возведение переменной a в степень 2
        y = sqrt(a); // квадратный корень переменной a
        z = pow(a, 1.0/2); // квадратный корень переменной a

        sinus = sin(a);
        cosinus = cos(a);

        cout << "Квадрат a = " << x << "\n";
        cout << "Квадратный корень a = " << y << "\n";
        cout << "Квадратный корень a = " << z << "\n";
        cout << "Синус a = " << sinus << "\n";
        cout << "Косинус a = " << cosinus << "\n";

    system("pause"); // Задержка консольного окна
    return 0; // Значение, возвращаемое главной функцией
```



Примеры решения задач

```
int main() // Главная функция программы
{

    //Параметры задачи:
    double S1, S2, t1, t2, V;
    //Ввод параметров:
    cout << "Enter S1 = ";
    cin >> S1;
    cout << "Enter S2 = ";
    cin >> S2;
    cout << "Enter t1 = ";
    cin >> t1;
    cout << "Enter t2 = ";
    cin >> t2;
    V = (S1 + S2) / (t1 + t2);
    cout << "Speed V = " << V << "\n";
```



Условный оператор if()

- В C++ два условных оператора: `if()` и `switch()`. **Оператор** `if()` **позволяет выполнять разные блоки операторов в зависимости от того, выполняется ли некое условие.**
- Условие указывается в круглых скобках после
- ключевого слова `if`. Общий синтаксис вызова оператора следующий:
- `if(условие) {операторы 1}`
- `else {операторы 2}`

Условный оператор if()

- Если условие, указанное после ключевого слова if, верно, выполняется
- блок операторов операторы 1. В противном случае выполняется блок операторов операторы 2, указанных после ключевого слова else. После выполнения условного оператора управление передается оператору, следующему после него.
- На рис. 2.1 показана структурная схема, иллюстрирующая работу условного оператора.

Условный оператор if()

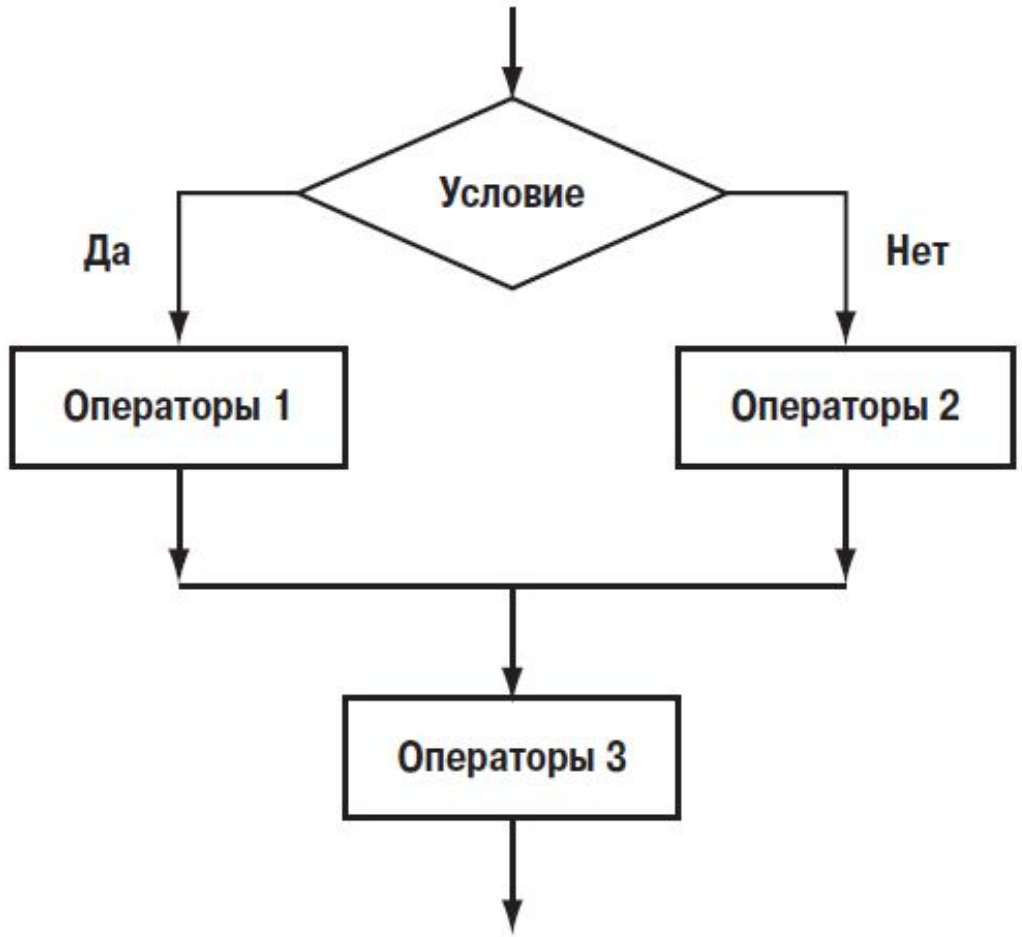


Рис. 2.1. Работа условного оператора

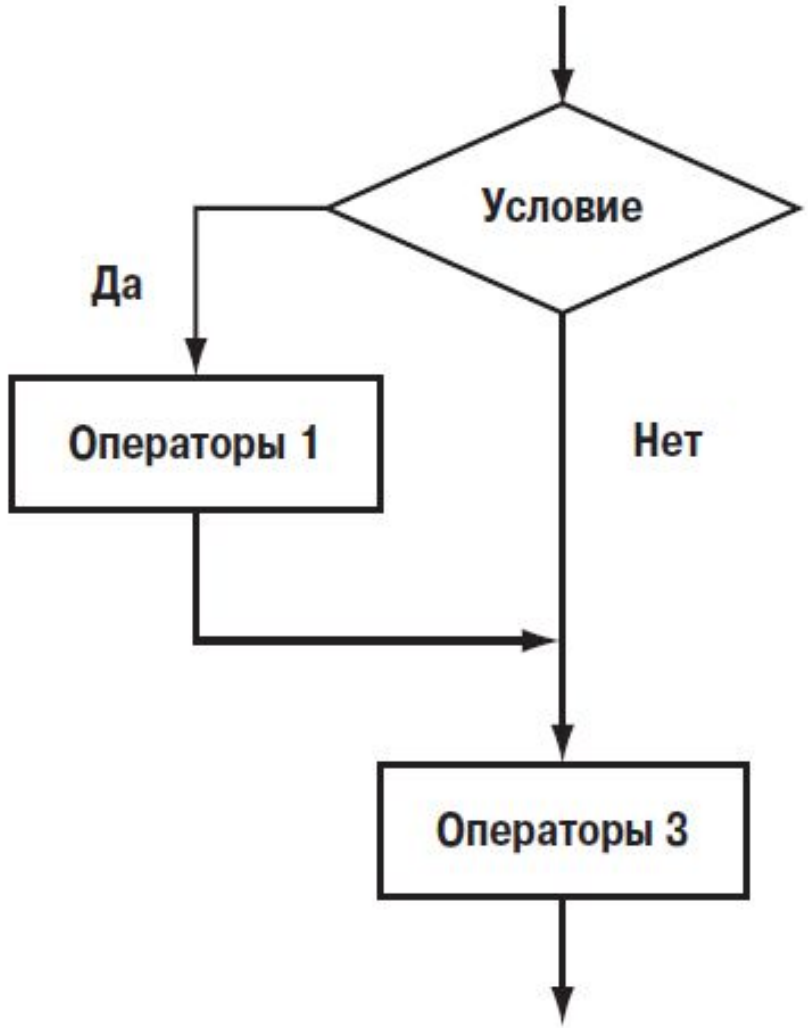
Условный оператор if()

- Допускается использование упрощенного варианта условного оператора, в котором отсутствует ветка else для выполнения операторов при невыполнении условия. Синтаксис вызова условного оператора в такой форме имеет вид:

if(условие) {операторы 1}

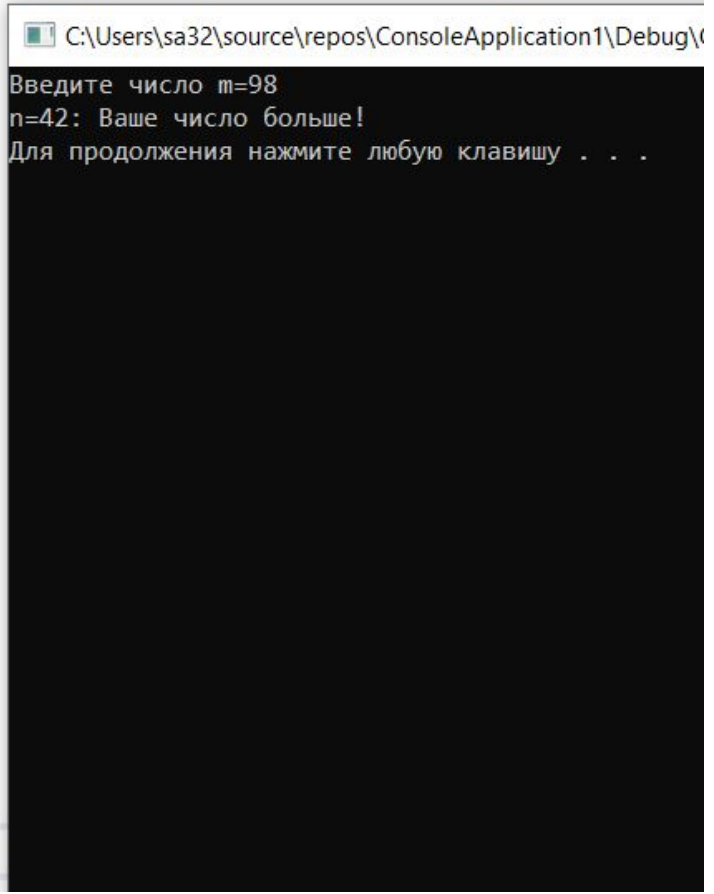
- В этом случае при выполнении условия управление передается блоку операторов, указанному после ключевого слова if. Если условие не выполнено, выполняются операторы, размещенные после условного. На рис. 2.2 показана структурная схема работы упрощенного варианта условного оператора.

Условный оператор if()



Пример

```
int main() // Главная функция программы
{
    setlocale(LC_ALL, "Russian"); // Изменение кодировки консоли

    int n, m;
    n = rand() % 100 + 1;
    cout << "Введите число m=";
    cin >> m;
    cout << "n=" << n << ": ";
    if (m > n) cout << "Ваше число больше!\n";
    else if (n > m) cout << "Ваше число меньше!\n";
    else cout << "Вы угадали!\n";

    system("pause"); // Задержка консольного окна
    return 0; // Значение, возвращаемое главной функцией
}
```



Задание 1

- Написать программу которая определяет подъезд и этаж по номеру квартиры.

- Пример задачи, которую должна выполнять программа:
- В пятиэтажном доме 3 подъезда, на одном этаже располагаются 4 квартиры. В каком подъезде и на каком этаже находится квартира №48

- Дополнение: модифицировать программу так, чтобы пользователь сам вводил количество подъездов и этажность здания, а затем номер квартиры.

Задание 2: Написать программу для решения прямой геодезической задачи

Решение **прямой геодезической задачи** выполняется по формулам:

$$\left.\begin{aligned} X_B &= X_A + \Delta X \\ Y_B &= Y_A + \Delta Y \end{aligned}\right\} (1)$$

где $\Delta X, \Delta Y$ называются приращениями координат и определяются из решения прямоугольного треугольника $AA'B$:

$$\left.\begin{aligned} \Delta X &= d\cos\alpha \\ \Delta Y &= d\sin\alpha \end{aligned}\right\} (2)$$

Знаки приращений координат ($\Delta X, \Delta Y$) зависят от четверти, в которой находится заданное направление и определяются по формулам 2, с помощью рисунка приведенного выше, или с помощью таблицы

Четверть	Значение дирекционного угла	Название румба	Связь между румбами и дирекционными углами	Знаки приращения координат	
				ΔX	ΔY
1	$0^0 - 90^0$	СВ	$r = \alpha$	+	+
2	$90^0 - 180^0$	ЮВ	$r = 180^0 - \alpha$	–	+
3	$180^0 - 270^0$	ЮЗ	$r = \alpha - 180^0$	–	–
4	$270^0 - 360^0$	СЗ	$r = 360^0 - \alpha$	+	–

Пример решения:

Пример решения прямой геодезической задачи

Дано: координаты точки A равны $X_A = 25м$, $Y_A = 140м$, горизонтальное проложение линии $d_{AB} = 124м$, дирекционный угол линии AB равен $\alpha_{AB} = 217^0 14' 23''$

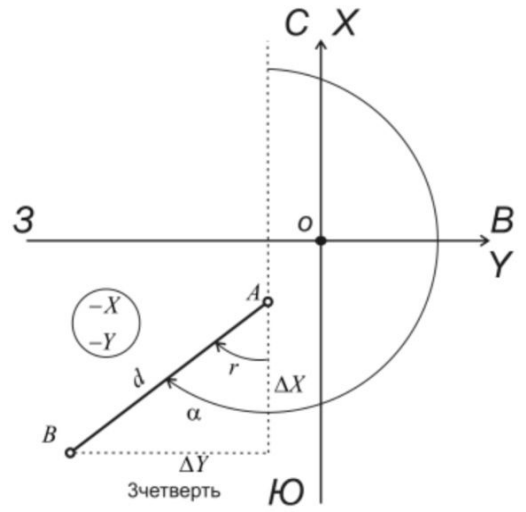
Найти: координаты точки B ($X_B = ?$, $Y_B = ?$)

Решение:

1) определяем приращения координат

$$\Delta X = d\cos\alpha = 124м \times \cos 217^0 14' 23'' = 124м \times (-0,7961) = -98,72м$$
$$\Delta Y = d\sin\alpha = 124м \times \sin 217^0 14' 23'' = 124м \times (-0,6052) = -75,04м$$

знаки приращений координат говорят о том, что заданное направление находится в 3 четверти



2) определяем координаты точки B

$$X_B = X_A + \Delta X = 25м - 98,72м = -73,72м$$
$$Y_B = Y_A + \Delta Y = 140м - 75,04м = 64,96м$$

Задание 3: Написать программу для решения обратной геодезической задачи

Решение **обратной геодезической задачи** выполняется в следующем порядке:

1) вычисляют приращения координат

$$\Delta X = X_B - X_A$$
$$\Delta Y = Y_B - Y_A$$

2) из решения прямоугольного треугольника $AB'B$ определяют румб линии r_{AB} :

$$tgr = \frac{\Delta Y}{\Delta X}$$

откуда

$$r = arctg\left|\frac{\Delta Y}{\Delta X}\right|$$

3) по знакам приращений координат ($\Delta X, \Delta Y$) с помощью таблицы определяют в какой четверти находится заданное направление и по известному румбу линии (r_{AB}) определяют дирекционный угол линии α_{AB}

Четверть	Значение дирекционного угла	Название румба	Связь между румбами и дирекционными углами	Знаки приращения координат	
				ΔX	ΔY
1	$0^0 - 90^0$	СВ	$r = \alpha$	+	+
2	$90^0 - 180^0$	ЮВ	$r = 180^0 - \alpha$	–	+
3	$180^0 - 270^0$	ЮЗ	$r = \alpha - 180^0$	–	–
4	$270^0 - 360^0$	СЗ	$r = 360^0 - \alpha$	+	–

4) определяют горизонтальное проложение (длину линии)

$$d = \frac{\Delta X}{\cos\alpha}$$
$$d = \frac{\Delta Y}{\sin\alpha}$$
$$d = \sqrt{\Delta X^2 + \Delta Y^2}$$

Горизонтальное проложение линии может быть вычислено трижды, что является хорошим контролем вычислений.

Пример решения:

Пример решения обратной геодезической задачи

Дано: координаты точек A и B равны

$$X_A = 247{,}32м$$
$$Y_A = 870{,}54м$$
$$X_B = 705{,}65м$$
$$Y_B = -567{,}83м$$

Найти: горизонтальное проложение и дирекционный угол линии A $(d_{AB} - ?, \alpha_{AB} - ?)$

Решение:

1) определяем приращения координат

$$\Delta X = X_B - X_A = 705{,}65 - 247{,}32 = 458{,}33м$$
$$\Delta Y = Y_B - Y_A = -567{,}83 - 870{,}54 = -1438{,}37м$$

2) определяем румб линии AB

$$r = arctg\left|\frac{\Delta Y}{\Delta X}\right| = arctg\left|\frac{-1438{,}37}{458{,}33}\right| = arctg\left|-3{,}13829\right| = 72{,}325759^0 = 72^0 19' 33''$$

3) по знакам приращений координат (числитель ΔY имеет знак минус, знаменатель ΔX имеет знак плюс), пользуясь таблицей связи румбов и дирекционных углов, определяем, что заданное направление находится в 4 четверти и румб линии AB равен $r_{AB} = СЗ : 72^0 19' 33''$

4) Для 4 четверти, пользуясь таблицей, определяем, что румб находится по формуле

$$r = 360^0 - \alpha$$

Переписав формулу румба (переносим правую и левую части формулы с изменением знака), дирекционный угол найдем по формуле:

$$r = 360^0 - \alpha \Rightarrow \alpha = 360^0 - r = 360^0 - 72^0 19' 33'' = 287^0 40' 27''$$

5) определяем горизонтальное проложение линии AB

$$d = \frac{\Delta X}{\cos\alpha} = \frac{458{,}33}{\cos 287^0 40' 27''} = \frac{458{,}33}{0{,}3036035} = 1509{,}63м$$
$$d = \frac{\Delta Y}{\sin\alpha} = \frac{-1438{,}37}{\sin 287^0 40' 27''} = \frac{-1438{,}37}{-0{,}9527985} = 1509{,}63м$$
$$d = \sqrt{\Delta X^2 + \Delta Y^2} = \sqrt{458{,}33^2 + (-1438{,}37)^2} = 1509{,}63м$$

Условный оператор switch()

- В тех случаях, когда проверяется больше одного условия, вместо нескольких вложенных условных операторов `if()` нередко используют оператор `switch()`.
- Синтаксис вызова оператора `switch()` следующий:

```
switch(выражение){
  case значение1:
    Операторы
    break;
  case значение2:
    операторы
    break;
    ...
  default:
  операторы
}
```

Условный оператор switch()

- В круглых скобках после ключевого слова switch указывается выражение, значение которого проверяется. Результатом выражения может быть целое число или символ. Значение, возвращаемое выражением, сравнивается со значениями, указанными после ключевых слов case. Если имеет место совпадение, выполняется соответствующий блок операторов. Операторы выполняются до конца оператора switch() или пока не встретится инструкция break (в общем случае инструкция break используется для выхода из оператора цикла и перехода к следующему оператору). Если совпадения нет, выполняются операторы после инструкции default.

Условный оператор switch()

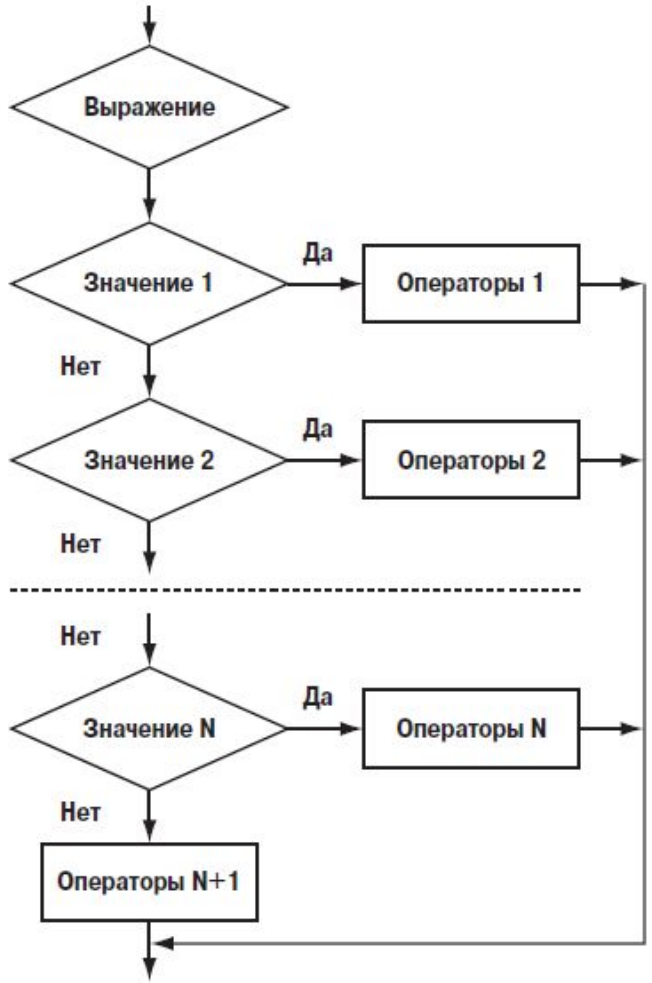


Рис. 2.4. Схема работы оператора выбора `switch()`

Условный оператор switch()

- Представленная схема приведена в предположении, что в каждом case-блоке использована инструкция break, а в конце switch-оператора использована инструкция default. Отметим, что эта инструкция не является обязательной, также как и инструкции break.

Пример

```
int main() // Главная функция программы
{
    setlocale(LC_ALL, "Russian"); // Изменение кодировки консоли

    int n;
    cout << "Enter n=";
    cin >> n;
    switch (n) {
    case 1:
        cout << "First case-block\n";
        break;
    case 2:
        cout << "Second case-block\n";
        break;
    case 3:
        cout << "Third case-block\n";
        break;
    default:
        cout << "By default\n";
    }
    system("pause"); // Задержка консольного окна
    return 0; // Значение, возвращаемое главной функцией
}
```

Пример

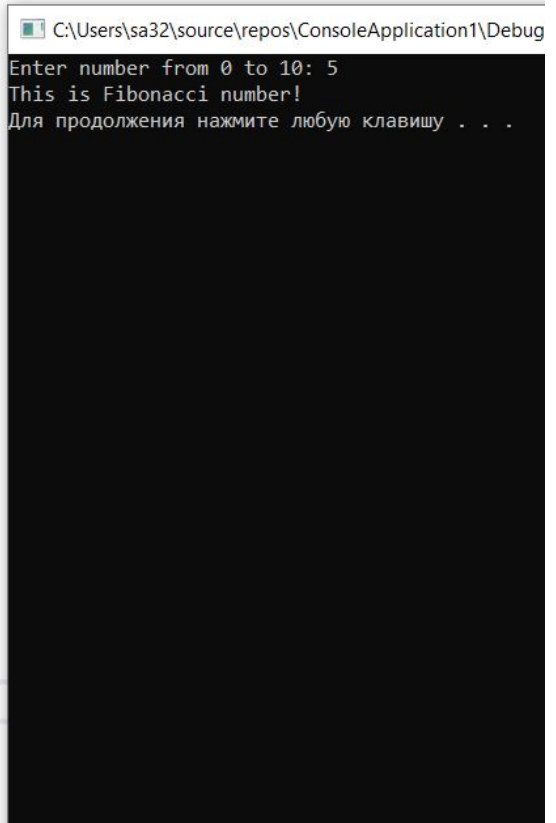
- В программе с клавиатуры вводится значение для целочисленной переменной n. Далее в switch-операторе проверяется значение этой переменной. Выделяются три значения этой переменной (1, 2 и 3 соответственно), а также предусмотрен default-блок для обработки ситуации, не предусмотренной в case-блоках. Результат выполнения программы прост: в зависимости от введенного числа выводится сообщение соответствующего содержания. Если пользователь вводит целое число от 1 до 3 включительно, выводится сообщение First case-block, Second case-block и Third case-block соответственно. В противном случае (т.е. когда введенное пользователем число не равно 1, 2 или 3) выводится сообщение By default.

Пример

- Особенность механизма выхода из оператора switch (имеется в виду выход из оператора с помощью инструкции break) позволяет объединять несколько case-условий.
- В процессе выполнения программы пользователю предлагается ввести целое число в диапазоне от 0 до 10. Число считывается с клавиатуры и выполняется проверка на предмет того, является ли оно нулем и принадлежит ли последовательности чисел Фибоначчи.
- Напомним, что последовательность Фибоначчи получается так: первые два числа последовательности равны 1, а каждое последующее равно сумме двух предыдущих. Начальные числа в последовательности Фибоначчи, таким образом, равны 1, 1, 2, 3, 5 и 8 (это те числа, что попадают в диапазон от 0 до 10).

Пример

```
int main() // Главная функция программы
{
    setlocale(LC_ALL, "Russian"); // Изменение кодировки консоли

    int n;
    cout << "Enter number from 0 to 10: ";
    cin >> n;
    switch (n) {
    case 0:
        cout << "The number is zero!\n";
        break;
    case 1:
    case 2:
    case 3:
    case 5:
    case 8:
        cout << "This is Fibonacci number!\n";
        break;
    default:
        cout << "This is integer number!\n";
    }
    system("pause"); // Задержка консольного окна
    return 0; // Значение, возвращаемое главной функцией
```



Задание 4

- Используя оператор switch объединить 2 предыдущие программы для решения ПГЗ и ОГЗ в одну.

- Подсказка: При запуске программа должна спросить пользователя, что он собирается вычислить, и после его ответа запросить данные, необходимые для решения выбранной им задачи.