Уральский федеральный университет

Виды алгоритмов

Лекция 6

Иллюстративный материал к лекциям по алгоритмизации и программированию

Автор Саблина Н.Г. 2016 г.

Уральский федеральный университет

Содержание

Виды алгоритмов

- Линейные
- Разветвляющиеся
- Циклические

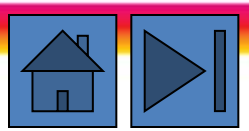
Линейный алгоритм

Линейным или ***простейшим*** называют такой алгоритм, в котором операторы программы выполняются в линейной последовательности, т.е. друг за другом все - от первого до последнего

Структура программы вычислительной задачи

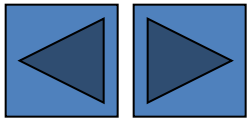
Исполняемый блок любой вычислительной программы можно условно разделить на следующие части

- заставка;
- ввод исходных данных;
- собственно вычислительная часть;
- вывод результатов вычислений



Заставка

- Она является визитной карточкой программы.
- Содержит краткие сведения о
  - назначении,
  - авторе,
  - времени создания программы.
- Все эти сведения выводятся на экран.
- Для формирования заставки используются операторы вывода на экран
  - Функция стандартной библиотеки языка Си printf
  - Средства библиотеки потокового ввода-вывода языка C++ cout



Ввод исходных данных

- Организован в виде диалога
  - «Запрос» - «ответ»
- Формируется чередованием операторов
  - вывода на экран
  - ввода с клавиатуры

Примеры диалога ввода исходных данных на языке Си (1)

- **Пример 1**

```
//Ввод исходных данных
    printf("\nВведите переменную x=");
    scanf("%d", &x);
    printf("Введите погрешность eps=");
    scanf("%f", &eps);
//Конец ввода исходных данных
```

Примеры диалога ввода исходных данных на языке Си (2)

- **Пример 2**

```
//Ввод исходных данных
    printf("\nВведите длины сторон треугольника \n a=");
    scanf("%f", &a);
    printf("\n b="); scanf("%f", &b);
    printf("\n c="); scanf("%f", &c);
//Конец ввода исходных данных
```

Запись формул в программе (1)

Математические операции:

+ сложение; - вычитание;

* умножение; / деление;

% – целочисленное деление (остаток).

++ инкремент -- декремент

При записи действительных чисел десятичная часть отделяется точкой.

Результат вычисления присваивается некоторой переменной с помощью знака присваивания (=)

Запись формул в программе (2)

Математические действия выполняются

- в порядке убывания их приоритета,
- порядок действий регулируется круглыми скобками.

Пример:

Формула:

$$x = \frac{102{,}5(a - 5{,}74) + b(23{,}6 + 1{,}4)}{5{,}23 - 3{,}6 + 2}$$

Оператор Си

```
x=(102.5*(a - 5.74) + b*(23.6 + 1.4)) / (5.23 - 3.6 + 2);
```

Вывод результатов вычислений

Вывод результата осуществляется на экран дисплея (в большинстве учебных примеров)

Используются операторы вывода на экран.

Пример 1

```
//Вывод результатов
printf ( “Количество студентов-отличников в группе %d человек “, N);
//Конец вывода результатов
```

Вывод результата вычислений в Си

Пример 2

```
//Вывод результатов
printf (“ Длина биссектрисы угла С равна Lc= %5.2f \n
      Длина биссектрисы угла А равна La= %5.2f  \n
      Длина биссектрисы угла В равна Lb= %5.2f “, Lc, La, Lb);
//Конец вывода результатов
```

Пример 3

```
//Вывод результатов
printf (“Медианы треугольника A=%6.3f   B= %6.3f C= %6.3f”, x, z, k);
//Конец вывода результатов
```

Пример 1. Площадь и периметр прямоугольника

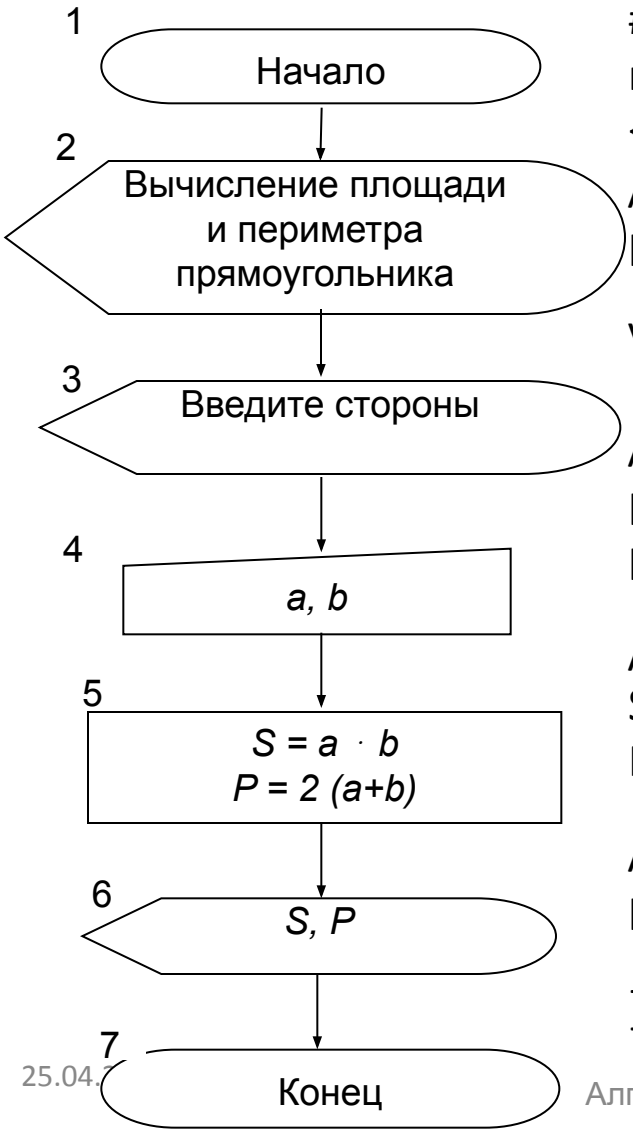
- 1. Постановка задачи
  - Исходные данные:

    a,b – стороны прямоугольника, действительные числа, вводятся с клавиатуры
  - Выходные данные: …

    S, P – площадь и периметр прямоугольника, действительные числа
- 2. Метод решения …

  $S = a \cdot b \qquad P = 2\,(a+b)$

Пример 1. Площадь и периметр прямоугольника




```
#include <stdio.h>
main()
{  float a,b,S,P;
// заставка
printf(" \nПрограмма вычисления площади
     прямоугольника
\nразмером axb \nАвтор: студент гр. Р-15061 Иванов
     И.И. 2006 г.");
// ввод исходных данных
printf(" \n,Введите ширину a=");  scanf("%f", &a);
printf(" \n,Введите длину b=");  scanf("%f", &b);

// вычисляемая часть
S=a*b;
P=2*(a+b);

// Вывод результатов
printf("\nплощадь прямоугольника S=%f \nпериметр
     P=%f ", S,P);
}
```

Разветвляющиеся алгоритмы

Позволяют разделить ход выполнения программы на взаимоисключающие ветви в зависимости от некоторого условия.

Реализуются с помощью

- условного оператора
- оператора варианта

Условный оператор (полная форма)

позволяет разделить выполнение программы на две взаимоисключающие ветви

Синтаксис оператора

if (логическое выражение) ОПЕРАТОР 1;

else ОПЕРАТОР 2;

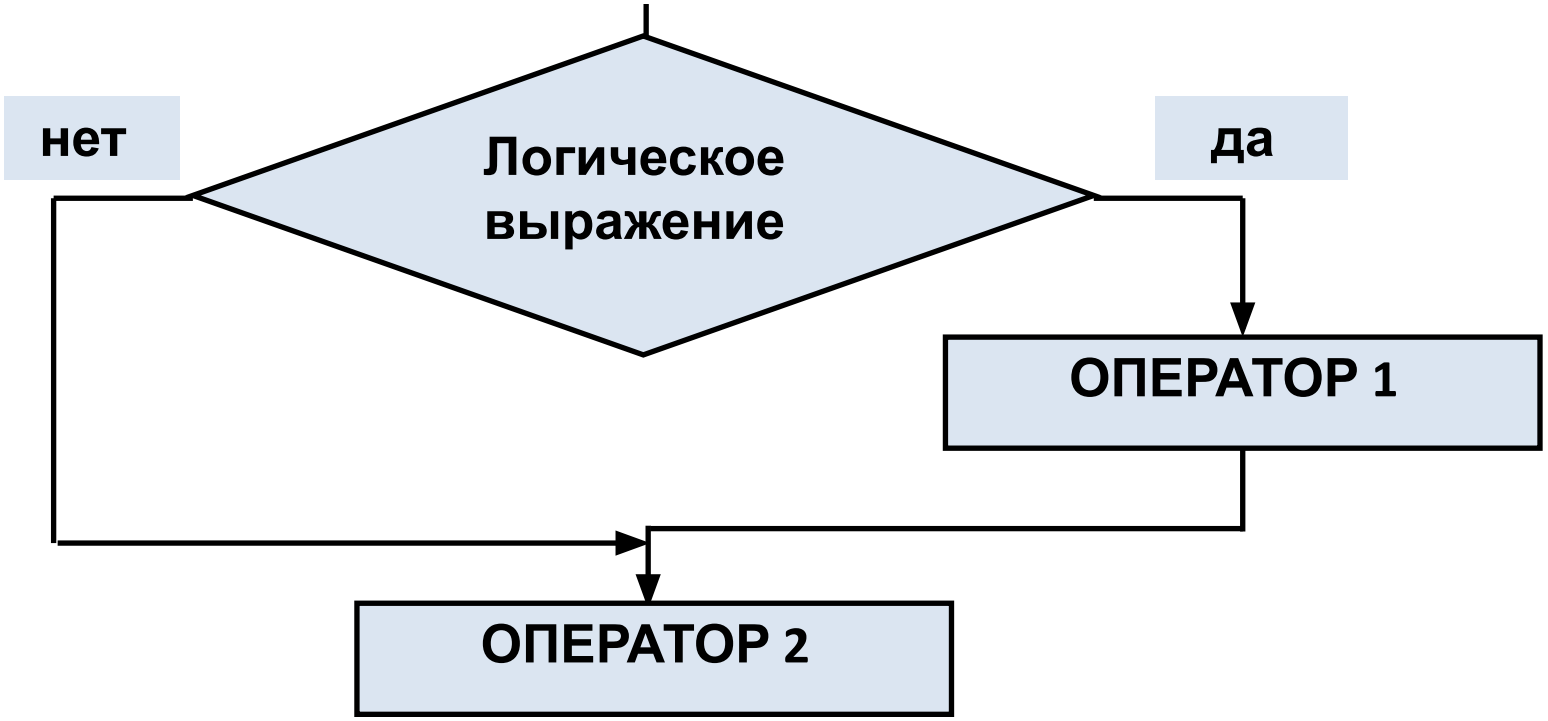
ОПЕРАТОР 3;

- ***if, else*** – служебные слова;
- ***логическое выражение*** – принимает значения **false** или **true**;
- <ОПЕРАТОР 1> , <ОПЕРАТОР 2> , <ОПЕРАТОР 3> - любые операторы языка С.

Пример

```
float x, y, pmax;

// инициализация переменных x, y

……….

if (x>=y) pmax=x;

else

      pmax=y;
```

Условный оператор (сокращенная форма)



используется, когда в ветви «нет» не требуется выполнять каких-либо действий

Пример 1

```
float x, y, pmax;

……….

if(x>pmax)     pmax=x; y=x;
```

Пример 2

Рассмотрим еще один пример использования краткой формы условного оператора

```
#include <stdio.h>
main()
{
float  x;  printf (“Введите число: ”); scanf (“%f”, &x);
if (x>0) printf (“Число %f положительное\n”,x);
if (x==0) printf (“Число %f равно нулю\n”,x);
if (x<0) printf (“Число %f отрицательное\n”,x);
}
```

Составной оператор (1)

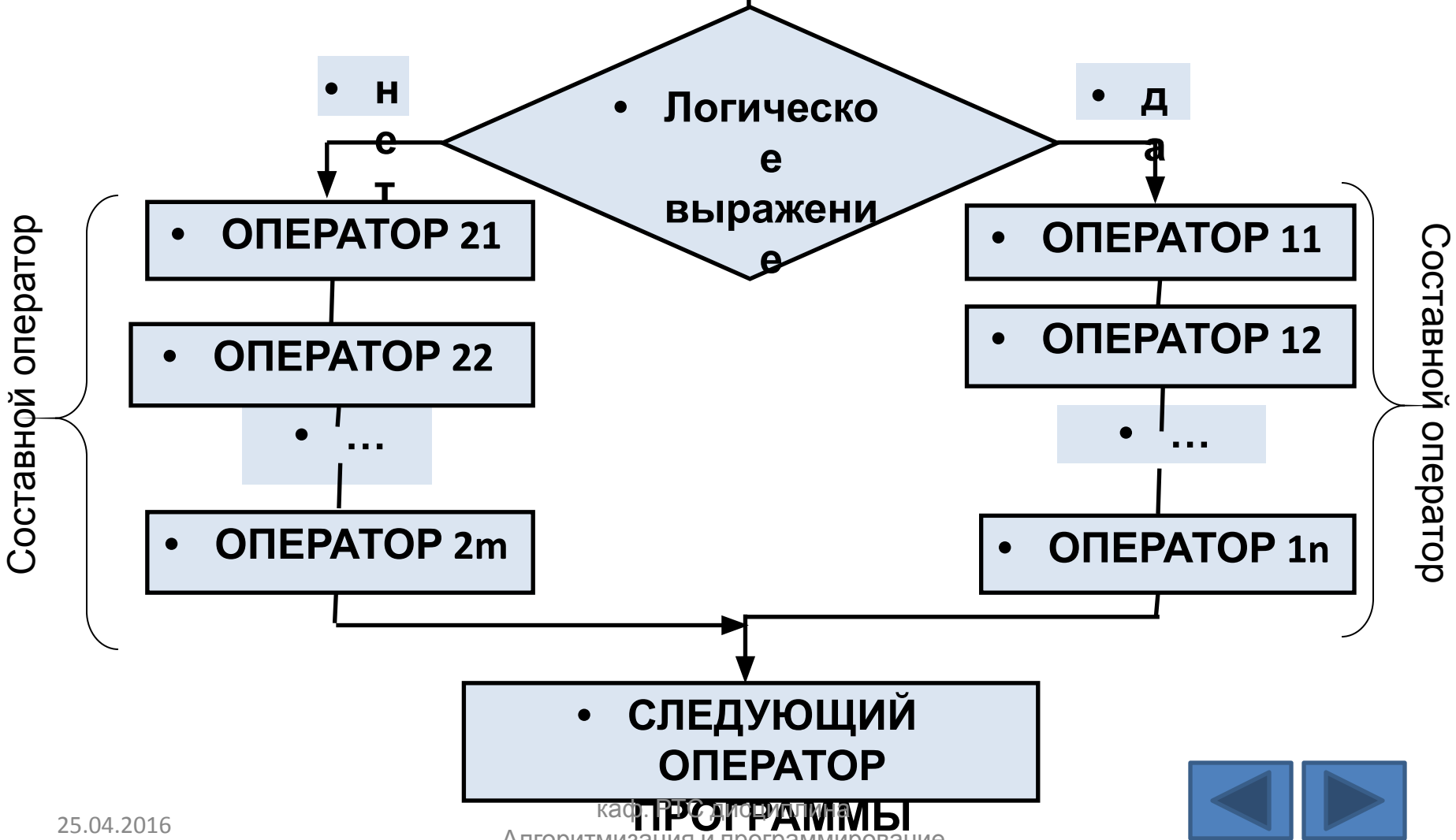
Применяется, когда синтаксис языка допускает использование только одного оператора, а семантика программы требует выполнения последовательности действий.

Операторы составного оператора заключаются в фигурные скобки - { и } - и отделяются друг от друга символами «;».

Составной оператор (2)

- Операторы, входящие в него, выполняются последовательно «один за другим».
- Нет ограничений на характер операторов, входящих в составной оператор.
- Может включать в себя и другие составные операторы. Язык Си допускает произвольную глубину их вложенности

Полная схема условного оператора с использованием составных операторов



Соответствующая схеме инструкция в программе будет выглядеть следующим образом:

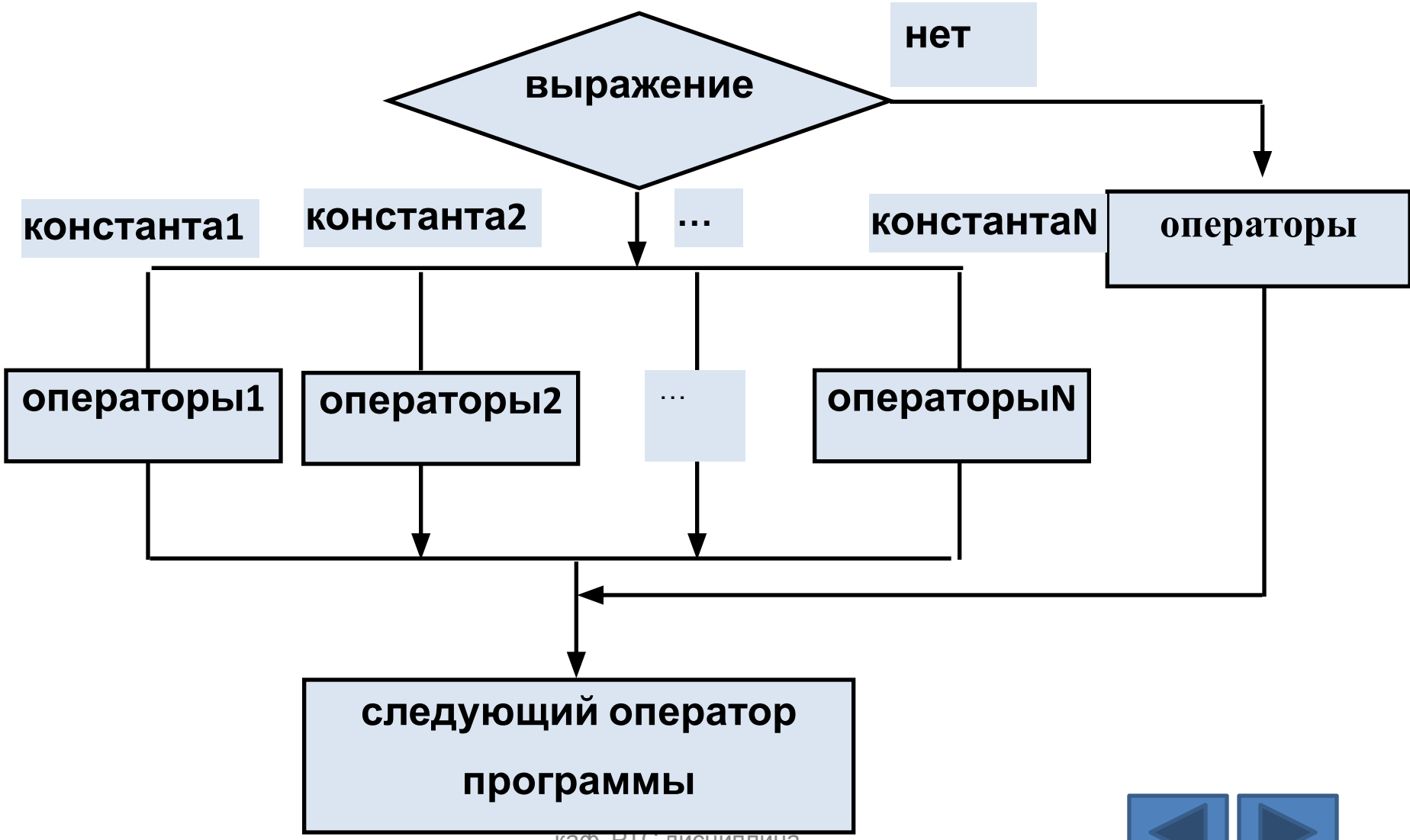
```
if (логическое выражение)
{ ОПЕРАТОР 11;
  ОПЕРАТОР 12;
   …
  ОПЕРАТОР 1n;   }
  else
{ОПЕРАТОР 21 ;
 ОПЕРАТОР 22;
   …
ОПЕРАТОР 2m;   }
СЛЕДУЮЩИЙ ОПЕРАТОР ПРОГРАММЫ ;
```

Оператор варианта (выбора)

Является обобщением условного оператора для произвольного числа альтернатив.

Если необходимо выбрать один из нескольких вариантов, то вместо вложенных конструкций **if** удобнее применять оператор множественного выбора (оператор-переключатель) **switch**.

Блок-схема оператора выбора



Прядок работы оператора варианта

- Вычисляется выражение в скобках за ключевым словом **switch**
- Просматривается список меток (**case** константа1 и т. д.) до тех пор, пока не находится метка, соответствующая значению выражения
- Выполняются действия соответствующей ветви **case**
- Если значение выражения не соответствует ни одной из меток **case,** выполняются операторы ветви **default**

Основная форма оператора :

```
switch (выражение) {
 case константа1:
последовательность операторов
break;
case константа2:
последовательность операторов
break;
…
case константаN:
последовательность операторов
break;
default
последовательность операторов
}
```

Пример «Список писателей»

```
#include <stdio.h>
//Пример оператора switch с использованием break
main()
{    char ch;
printf ("Введите заглавную букву русского алфавита:");
ch=getchar();
if(ch>='A' && ch<='Я')
switch(ch)
```

```
{ case 'А':   printf ("Алексеев \n");  break;
  case 'Б':   printf(" Булгаков \n");  break;
  case 'В':   printf  (" Волошин \n");  break;
  default:
printf ("Нет в списке писателя, чья фамилия начинается с
    этой буквы \n");  break;    }
else printf ("Надо было ввести заглавную русскую букву\n"):
}
```

Пример
Нечетные цифры

Приведенная ниже программа выводит на экран названия нечетных цифр, не меньших заданной

```
#include <iostream.h>

void main()

{int n;

cout<<"\nВведите любую десятичную цифру: ";

 cin >> n;

 switch (n)
```

```
{case 0:  case 1:  cout << “один, “;
 case 2:  case 3:  cout << “три, “;
 case 4:  case 5:  cout << “пять, “;
 case 6:  case 7:  cout << “семь, “;
 case 8:  case 9:  cout << “девять, “; break;
   default :  cout << “ \nНеверный ввод данных “;
  }
}
```

Итоги
Рассмотренные вопросы:

- Программирование разветвленных алгоритмов
- Условный оператор
- Составной оператор
- Оператор варианта

Библиографический список

- Подбельский В.В., Фомин С.С. Курс программирования на языке Си: учебник. М.: ДМК Пресс, 2012. – 384 с.
- Павловская Т.А. C/C++. Программирование на языке высокого уровня: учебник для студентов вузов, обучающихся по направлению "Информатика и вычисл. техника" СПб.: Питер, 2005. - 461 с.
- Павловская Т. А., Щупак Ю. А. C++. Объектно-ориентированное программирование. Практикум. Практикум. — СПб.: Питер, 2006. — 265 с: ил.
- Березин Б.И. Начальный курс C и C++ / Б.И. Березин, С.Б. Березин. - М.: ДИАЛОГ-МИФИ, 2001. - 288 с
- Каширин И.Ю., Новичков В.С. От С к С++. Учебное пособие для вузов. – М.: Горячая линия – Телеком, 2012. – 334 с.

Автор:

Саблина Наталья Григорьевна

Ст. преподаватель

каф. РТС УрФУ