**Глава 6. Условный оператор**

**6.1 Понятие алгоритма ветвления**

Разветвляющиеся алгоритмы представляют собой алгоритмы, имеющие несколько альтернативных путей, выбор которых зависит от выполнения определенных условий. Ветвление – это алгоритмическая конструкция, при выполнении которой в зависимости от результата проверки условия (истина или ложь) выполняется одна из двух последовательных команд, называемых ветвями [17].

Оператор if предназначен для реализации алгоритма ветвления. Условная конструкция в C++ всегда записывается в круглых скобках после оператора if. Внутри фигурных скобок указывается тело условия. Если условие истинно, то выполняются команды, которые находятся между фигурными скобками (в теле условия). В противном случае выполняются операции, следующие после слова else. Если после слова if или после слова else необходимо выполнить только один оператор, то фигурные скобки можно опустить. Синтаксис структуры ветвления представлен на рисунке 6.1 [25]

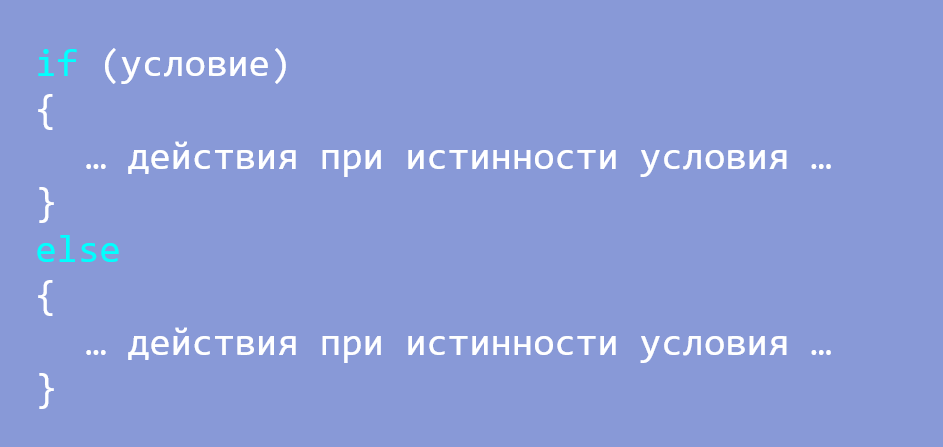


Рисунок 6.1 – Синтаксис структуры ветвления оператора if

Структура оператора if может представлять как полное, так и неполное ветвление. Ветвление называется неполным если отсутствует ветвь else, тогда при ложном условии действие переходит к оператору, находящемуся после ветвления. Структура полного и неполного ветвления показаны на блок-схемах (рисунок 6.2).

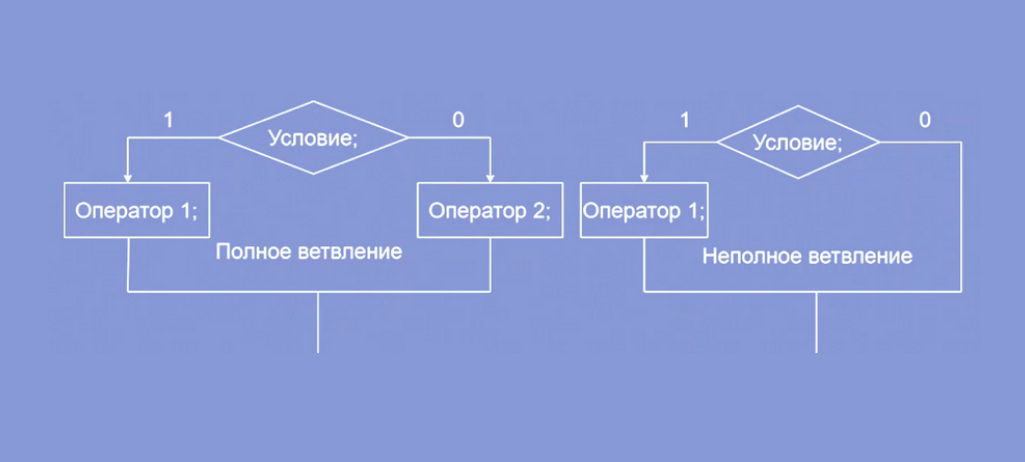


Рисунок 6.2 – Блок-схемы полного и неполного ветвления

Условие в операторе if не обязательно должно быть выражением. Это может быть любая целочисленная переменная, причем если значение переменной отлично от нуля, то выполнение алгоритма пойдет по ветви истина, иначе по ветви ложь. Также это может быть логическая переменная, которая имеет тип данных bool [14].

Для закрепления материала предлагается выполнить задачи, представленные ниже.

Задача 1:

Программа выполняется два раза с входными следующими данными: value =- 10 и value = 9. Необходимо объяснить, что будет выведено в результате выполнения программы (рисунок 6.3).

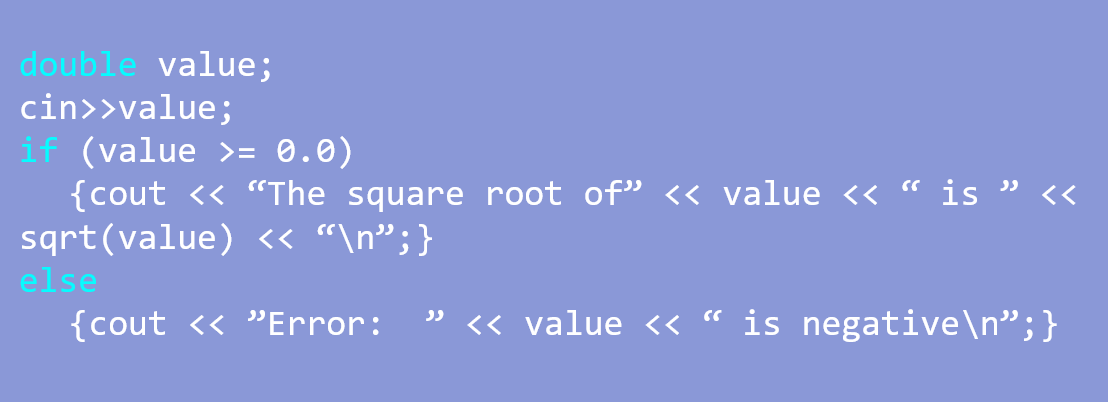


Рисунок 6.3 – Фрагмент кода программы задания №1

Задача 2:

Программа выполняется два раза с входными следующими данными: a = 5 и a = 16. Необходимо объяснить, что будет выведено в результате выполнения программы (рисунок 6.4).

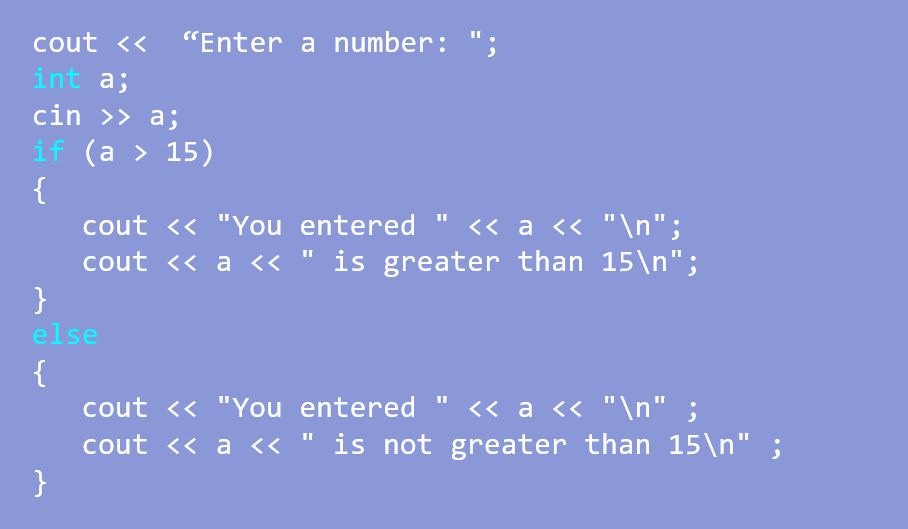


Рисунок 6.4 – Фрагмент кода программы задания №2

**6.2 Понятие вложенного условного оператора**

Конструкция, в которой внутри одного условия необходимо проверить другое условие, называется вложенным условным оператором. Вложенные условные операторы помогают сделать программу более гибкой. Примером использования конструкции вложенного условного оператора является нахождение минимального числа из трех чисел (рисунок 6.5).

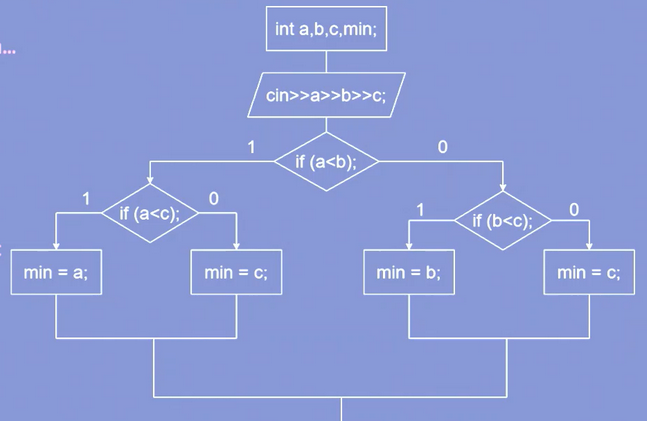


Рисунок 6.5 – Блок-схема алгоритма нахождения минимального числа с использованием вложенного условного оператора

Задача 3:

Программа выполняется два раза с входными следующими данными: num=10.5 и num = 10. Необходимо объяснить, что будет выведено в результате выполнения программы (рисунок 6.6).

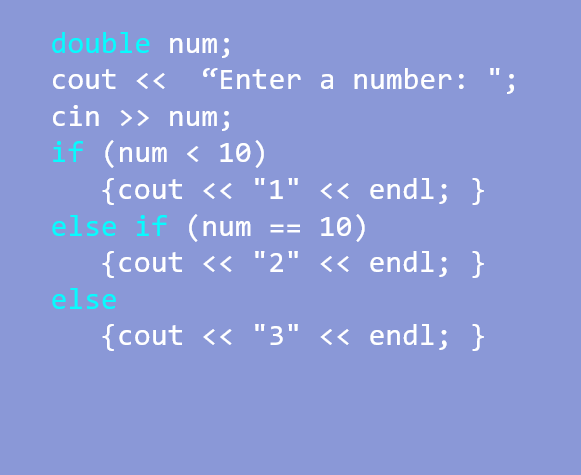


Рисунок 6.6 – Фрагмент кода программы задания №3

Задача 4:

Программа выполняется два раза с входными следующими данными: a = 6, a = 15 и a = 17. Необходимо объяснить, что будет выведено в результате выполнения программы (рисунок 6.7).

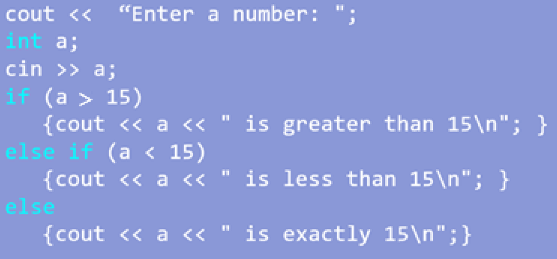


Рисунок 6.7 – Фрагмент кода программы задания №4