Условные конструкции

№ урока: 4 **Курс:** Java Starter

Средства обучения: Компьютер с установленной IntelliJ IDEA

Обзор, цель и назначение урока

Рассмотрение операторов ветвления для построения условных конструкций.

Изучив материал данного занятия, учащийся сможет:

- Понимать работу операторов ветвления.
- Использовать условные конструкции: `if-else`
- Использовать тернарный оператор
- Использовать оператор многозначного выбора `switch-case`.

Содержание урока

1. Рассмотрение понятия ветвления в программировании.
2. Обзор операторов ветвления.
3. Рассмотрение случаев применения условных конструкций.
4. Рассмотрение примера: Условная конструкция – `if` (с одной ветвью).
5. Рассмотрение примеров: Условная конструкция – `if-else` (с двумя ветвями).
6. Рассмотрение примера: Условная конструкция – `if-else` (с несколькими ветвями). Каскад условных операторов.
7. Рассмотрение примеров: Тернарная условная операция (`? :`).
8. Рассмотрение примера: Ограничения, связанные с типобезопасностью.
9. Рассмотрение примера: Вложенные тернарные операторы.
10. Рассмотрение примеров: Оператор многозначного выбора – `switch-case` (переключатель).
11. Рассмотрение примеров: Проваливание в переключателях.

Резюме

- Оператор ветвления (условная конструкция, условный оператор) – оператор, конструкция языка программирования, обеспечивающая выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд (наборов команд) в зависимости от значения некоторого выражения.
- В Java существует три основные формы условной конструкции:
    1. условный оператор (`if-else`)
    2. тернарный оператор (`? :`)
    3. оператор многозначного выбора (переключатель, `switch-case`).
- Условный оператор `if` реализует выполнение определённых команд при условии, что используемое логическое выражение в условии, принимает значение `true`.
- Если использовалась конструкция `if-else`, и результатом условия было значение `true`, то выполнится только тело оператора `if`, а тело блока `else` останется не выполненным.
- После выполнения оператора `if` управление передается следующему оператору.
- Оператор, выполняемый после проверки условия, может быть любого типа, включая другой оператор `if`, вложенный в оригинальный оператор `if`. Во вложенных операторах `if` предложение `else` принадлежит к последнему оператору `if`, у которого нет соответствующего `else`.
- Если тело блока `if` или `else` состоит из одного выражения, то операторные скобки можно опустить
- Тернарная условная операция (записывается как `? :`) – операция, возвращающая свой второй или третий операнд в зависимости от значения логического выражения, заданного первым операндом.
- Тернарный оператор `[ ? : ]`, является сокращенной формой конструкции `if... else`.

CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Тернарный оператор состоит из следующих операндов:
  (условие) ? (блок истинности или то) : (блок иначе) ;
- Алгоритм работы тернарного оператора:
  (логическое выражение) ? выражение 1 : выражение 2
  1. Вычисляется логическое выражение (условие).
  2. Если логическое выражение истинно, то вычисляется значение выражения выражение 1 (блока истинности), в противном случае – значение выражения выражение 2 (блока иначе).
  3. Вычисленное значение возвращается.
- Тернарный оператор обязательно должен возвращать значение, иначе будет ошибка.
- Либо блок истинности и блок иначе должны быть одинакового типа, либо должно существовать неявное преобразование из одного типа в другой.
- Конструкция переключателя `switch-case` имеет несколько (две или более) ветвей. Переключатель выполняет одну заданную ветвь в зависимости от значения вычисляемого ключевого выражения. Принципиальным отличием этой конструкции от условного оператора является то, что выражение, определяющее выбор исполняемой ветви, допускает использование не логических значений.
- Для пустых операторов `case` разрешено "проваливание" от одного оператора к другому.
- В каждом операторе `case` указывается постоянное значение. Выполняется тело того оператора `case`, постоянное значение которого, соответствует значению выражения селектора оператора `switch`.
- Если постоянное выражение оператора `case` не содержит соответствующего значения, выполняется блок `default`, если таковой имеется. Если блок `default` отсутствует, происходит выход за пределы оператора `switch`.
- Каждый блок `case`, как и блок `default`, в котором содержатся выполняемые операторы, должен завершаться оператором перехода `break`, `return` или `throw`
- Выполнение сравнения значения выражения селектора с постоянными значениями операторов `case` с первого оператора и продолжается по списку, обычно до достижения оператора перехода. В этой точке управление передается за пределы оператора `switch` или переходит к другому оператору `case`, если операторы перехода или тело оператора `case` отсутствовали, и так до того оператора `case`, у которого будет присутствовать тело и оператор перехода. Такая техника называется ***проваливанием***.
- Блок `default` может быть создан в любом месте тела переключателя `switch-case`. Исключением является тело операторов `case`.

Закрепление материала

- Обязательно ли оператор `if` должен использоваться вместе с оператором `else`?
- Обязательно ли создавать блок `default` в переключателе `switch`?
- Допустимо ли вложение тернарных операторов?
- Значения какого типа можно передавать в качестве параметра `if`()?
- Обязательно ли в переключателе `switch-case` использовать оператор перехода `break`?
- Может ли `switch-case` иметь только блок `default`?
- Что такое техника проваливания в операторе `switch-case`?

Дополнительное задание

Задание 1
Используя IntelliJ IDEA, создайте класс **Translator**.
Напишите программу русско-английский переводчик. Программа знает 10 слов о погоде. Требуется, чтобы пользователь вводил слово на русском языке, а программа давала ему перевод этого слова на английском языке. Если пользователь ввел слово, для которого отсутствует перевод, то следует вывести сообщение, что такого слова нет.

CyberBionic systematics
CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Самостоятельная деятельность учащегося

Задание 1
Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2
Используя IntelliJ IDEA, создайте класс **Calculator**.
Напишите программу – консольный калькулятор.
Создайте две переменные с именами `operand1` и `operand2`. Задайте переменным некоторые произвольные значения. Предложите пользователю ввести знак арифметической операции. Примите значение, введенное пользователем и поместите его в строковую переменную `sign`.
Для организации выбора алгоритма вычислительного процесса, используйте переключатель `switch`. Выведите на экран результат выполнения арифметической операции.
В случае использования операции деления, организуйте проверку попытки деления на ноль. И если таковая имеется, то отмените выполнение арифметической операции и уведомите об ошибке пользователя.

Задание 3
Используя IntelliJ IDEA, создайте класс **Interval**.
Напишите программу определения, попадает ли указанное пользователем число от 0 до 100 в числовой промежуток [0 - 14] [15 - 35] [36 - 50][50 - 100]. Если да, то укажите, в какой именно промежуток. Если пользователь указывает число, не входящее ни в один из имеющихся числовых промежутков, то выводится соответствующее сообщение.

Рекомендуемые ресурсы

Оператор if-then-else
https://docs.oracle.com/javase/tutorial/java/nutsandbolts/if.html

Оператор switch
https://docs.oracle.com/javase/tutorial/java/nutsandbolts/switch.html

CyberBionic systematics
CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com