3.4

Операторы break и continue.

00:00—00:25 — Введение

Привет! В этом видеоматериале вы познакомитесь с двумя операторами, которые используются внутри циклов break и continue. В переводе с английского break означает «прерывать», а continue — «продолжать». Кто и что прерывает и продолжает будет понятно на примерах, поэтому давайте сразу перейдём к ним.

00:25—04:36 — Пример

Начнём с оператора break. Представьте, что вам нужно закончить выполнение цикла до того, как он завершился сам, поскольку вы уже достигли нужного результата и продолжать цикл нет смысла. Например, у вас есть список лет, и мы хотим понять, есть ли в этом списке какой-то определённый год.
Поскольку списки как способ хранения данных мы ещё не изучали, то просто переберём диапазон лет. Например, с 1922 по 2022. Сделаем это с помощью цикла for.
Пусть изначально год равен 1922, и мы будем продолжать этот цикл до тех пор, пока год не станет равным 2022, будем увеличивать год каждый раз на единицу.
Пока код должен выглядеть так:

```
int firstYear = 1922;
int lastYear = 2022;

for (int year = firstYear; year <= lastYear; year = year +1) {
```

Представим на время, что этим циклом мы перебираем список, который где-то хранится. Мы хотим определить, есть ли в этом

списке какой-то определённый год и, как только мы этот год встретим, перебирать список дальше не нужно и цикл можно прервать, выведя информацию в консоль.

Для того, чтобы это сделать, мы можем воспользоваться оператором break, но сначала давайте создадим переменную, в которой будет храниться искомый год. «Искомый» по-английски — sought, поэтому искомый год будет обозначаться как sought year. Допустим, мы будем искать 1985 год и, соответственно, если текущий год равен искомому году, то можно вывести в консоль сообщение о том, что год найден, после этого можно прервать цикл, используя оператор break. Оператор break прерывает выполнение цикла в тот момент, когда мы его вызываем, — в данном случае мы вызываем его после выполнения условия и сообщения о том, что год найден.

Давайте разовьём эту задачу. Допустим, мы хотим дальше использовать информацию о том, что год найден в списке, и выполнять какой-то код уже после нашего цикла. В этом случае мы можем создать дополнительную переменную, которая изначально будет равна false — это будет переменная типа boolean. Назовём её year exist, что в переводе с английского — «год существует», то есть год найден. Изначально она будет равна false, и мы будем её менять в случае, если нужное значение было найдено, т. е. вместо вывода в консоль мы будем писать year exist=true и в этот момент прерывать цикл.

В результате мы сможем использовать эту переменную уже после цикла и поймём, был ли найден нужный год в процессе выполнения цикла.

Давайте используем эту переменную и для того, чтобы наша программа была более дружелюбна к пользователю — выведем сообщение о том, что год найден, следующим образом:

- сначала создадим две переменных — год найден и год не найден;
- затем можем воспользоваться тернарным оператором.
  Если year exist, то мы используем переменную found, то есть сообщение — «год найден».

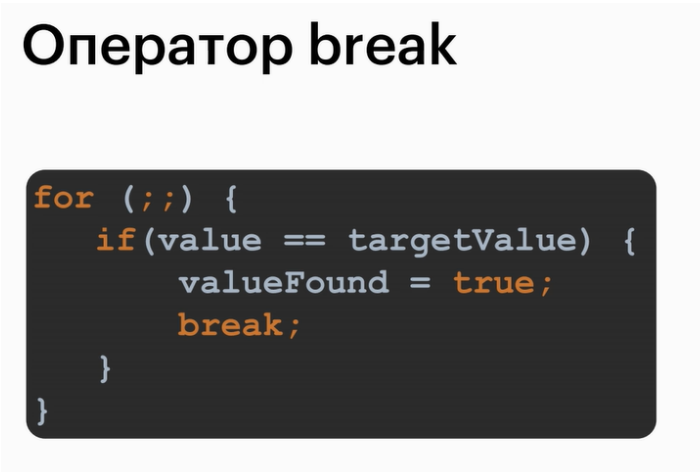
Если year exist=false, то — not found, т. е. «год не найден».
Обратите внимание, что оператор break прерывает цикл. Если после него написать какой-то код, он выполнен не будет.
После добавления переменных и оператора break код должен выглядеть следующим образом:

```
int soughtYear = 1985;

int firstYear = 1922;
int lastYear = 2022;

boolean yearExists = false;

for (int year = firstYear; year <= lastYear; year = year +1) {
     if (year == sought);
     yearExists = true;
     break;
    }
}

String found = “Год найден”;
String notFound = “Год не найден”;
System.out.println(yearExists ? found : notFound);
```

После запуска кода можно убедиться, что всё работает верно.

04:36—04:59 — Синтаксис

Если мы возьмём пример с изготовлением бургеров, то останавливать приготовление бургеров можно тогда, когда что-то случилось — например, засорился механизм подачи соуса.
Вот так пишется оператор break:



- Он пишется всегда внутри цикла.
- После него ставится точка с запятой — (;).

Он прерывает цикл и, если после него вы напишите какой-то код, то он также не будет выполнен.

04:59 — Второй пример

Теперь давайте посмотрим на другую ситуацию. Например, нам надо вывести все 29-е числа всех месяцев високосных лет, потому что только в високосные годы есть 29 февраля. Сначала мы укажем диапазон лет — пусть это будет тот же диапазон. Убираем всё лишнее и оставляем только цикл.

Високосные годы — годы, которые нацело делятся на четыре, поэтому пишем проверку, где знак процента означает остаток от деления. В данном случае на 4:

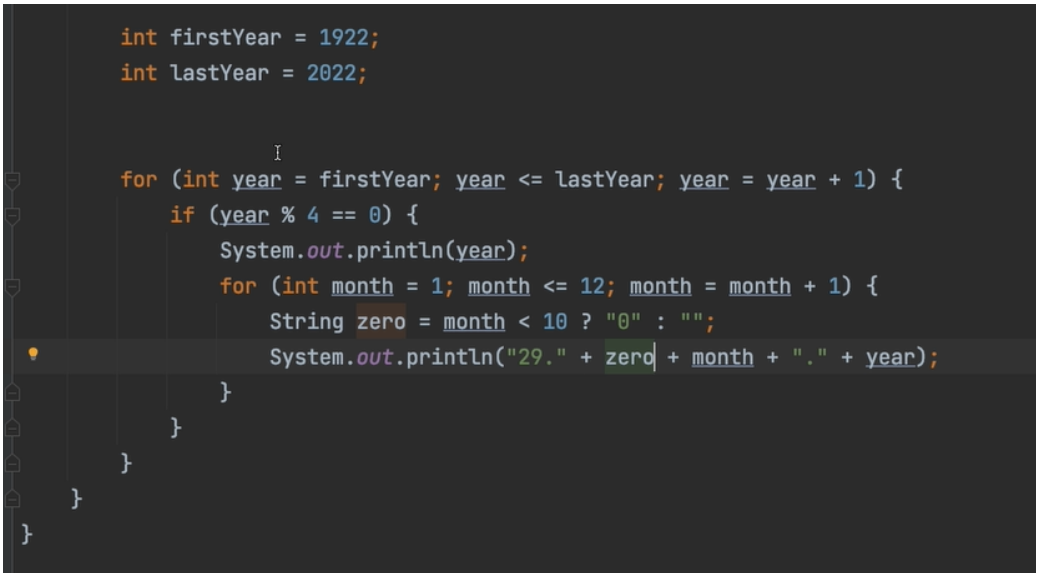
```
if (year % 4 == 0) {
```

Этим условием мы проверяем, делится ли нацело год на 4. Если делится — год високосный.

Внутри напишем код, который будет выводить сначала год, а затем все даты — все 29-е числа в этом году.

Если в строке мы будем ставить месяц, который имеет значение меньше 10, то нужно написать отдельную строку, которая в случае, если месяц меньше 10, будет содержать в себе 0.

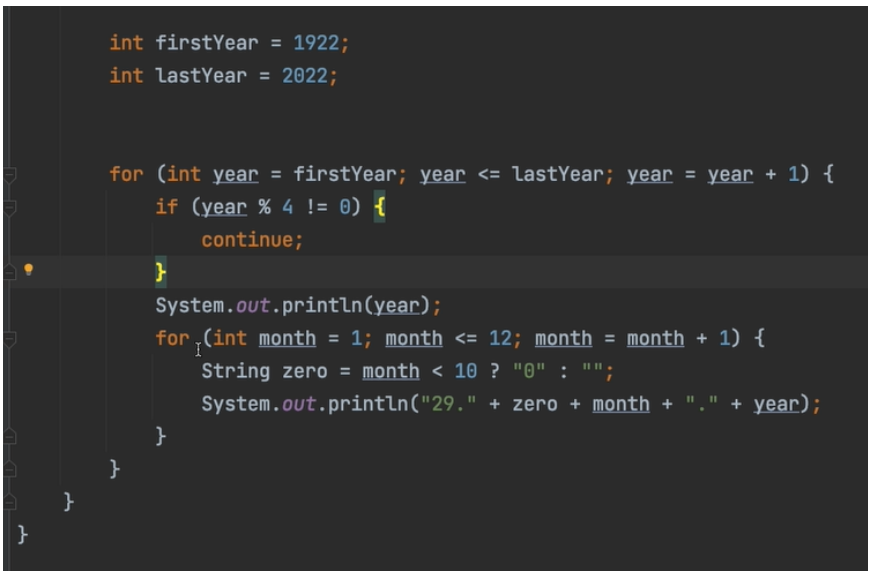
Пока код должен выглядеть так:



И всё с этим кодом было бы хорошо, если бы не высокая вложенность. Мы уже говорили о том, что программный код должен быть максимально поддерживаемым, и хорошей практикой является избегание больших уровней вложенности в коде.

Здесь в цикле находится условие If, а в нём ещё один цикл — итого 3 уровня вложенности. Их можно сократить с помощью специального оператора — continue. С его помощью можно написать обратное условие — если год не делится на 4, то в этот момент можно вызвать оператор continue, а затем весь оставшийся код сдвинуть на 1 уровень влево.

Что здесь будет делать оператор continue? По сути, он будет прерывать выполнение кода на текущей итерации, т. е. дальнейший ход в текущей итерации цикла выполняться не будет и сразу будет переходить к следующей итерации цикла. Таким образом, написанный нами код стал проще:



```
        int firstYear = 1922;
        int lastYear = 2022;


        for (int year = firstYear; year <= lastYear; year = year + 1) {
            if (year % 4 != 0) {
                continue;
            }
            System.out.println(year);
            for (int month = 1; month <= 12; month = month + 1) {
                String zero = month < 10 ? "0" : "";
                System.out.println("29." + zero + month + "." + year);
            }
        }
    }
}
```

Оператор continue удобен в тех случаях, когда вам не нужно выполнять код из текущей итерации, а нужно сразу перейти к следующей итерации цикла.
Если вернуться к примеру с аппаратом, который готовит бургеры, то по такому принципу можно построить механизм подачи новой булочки. Вспомним, что в аппарате осуществляется контроль каждого действия, и подача булочки определяется по изменению веса платформы, на которую эта булочка кладётся. Если механизм кладёт очередную булочку на подставку, а вес подставки не меняется, это означает, что что-то пошло не так. В этот момент можно прервать дальнейшее приготовление бургера и перейти к началу следующей операции, то есть попытаться снова положить булочку. Так же и с остальными этапами контроля качества — если на каком-то этапе что-то пошло не так, например, налили слишком много соуса, то этот этап тут же можно прервать, выбросить неудачный бургер и начать процесс сначала.

Оператор continue предназначен для прерывания текущей итерации цикла и перехода к следующей.

Итак, в этом видеоматериале вы познакомились с работой двух операторов, используемых в циклах, — break и continue. Их можно использовать в любых циклах, будь то for, while или do while.
Оператор break полезен тогда, когда вы хотите прервать цикл, а оператор continue — тогда, когда хотите прервать только текущую итерацию и перейти к следующей.