Java Basic

lecture #7. Switch...case, ternary operator

Mentor: <….>

lecture #7. Switch...case, ternary operator

- Switch
  - Switch в Java
  - Синтаксис: Switch-case
  - Важные правила для операторов switch
  - Блок-схема оператора Switch-Case
  - Вложенный Switch
  - switch VS if else
- Ternary operator
- switch vs if else
- practice
  - Выбор языка (ru, en, ua, de)
  - Выбор месяца

Switch в Java

Оператор switch является оператором многостороннего перехода.
Оператор switch выполняет один оператор из нескольких условий.
Это похоже if-else-if.
Выражение может быть примитивными типами данных byte, short, char и int.
Проверяет равенство переменных по нескольким значениям.

Эволюция:

1. <= JDK-6 -> byte, short, char и int
2. >= JDK-7 -> String
3. >= JDK-12 -> switch выражения, в отличие от switch инструкций
4. >= JDK-17 -> Сопоставление с образцом в switch

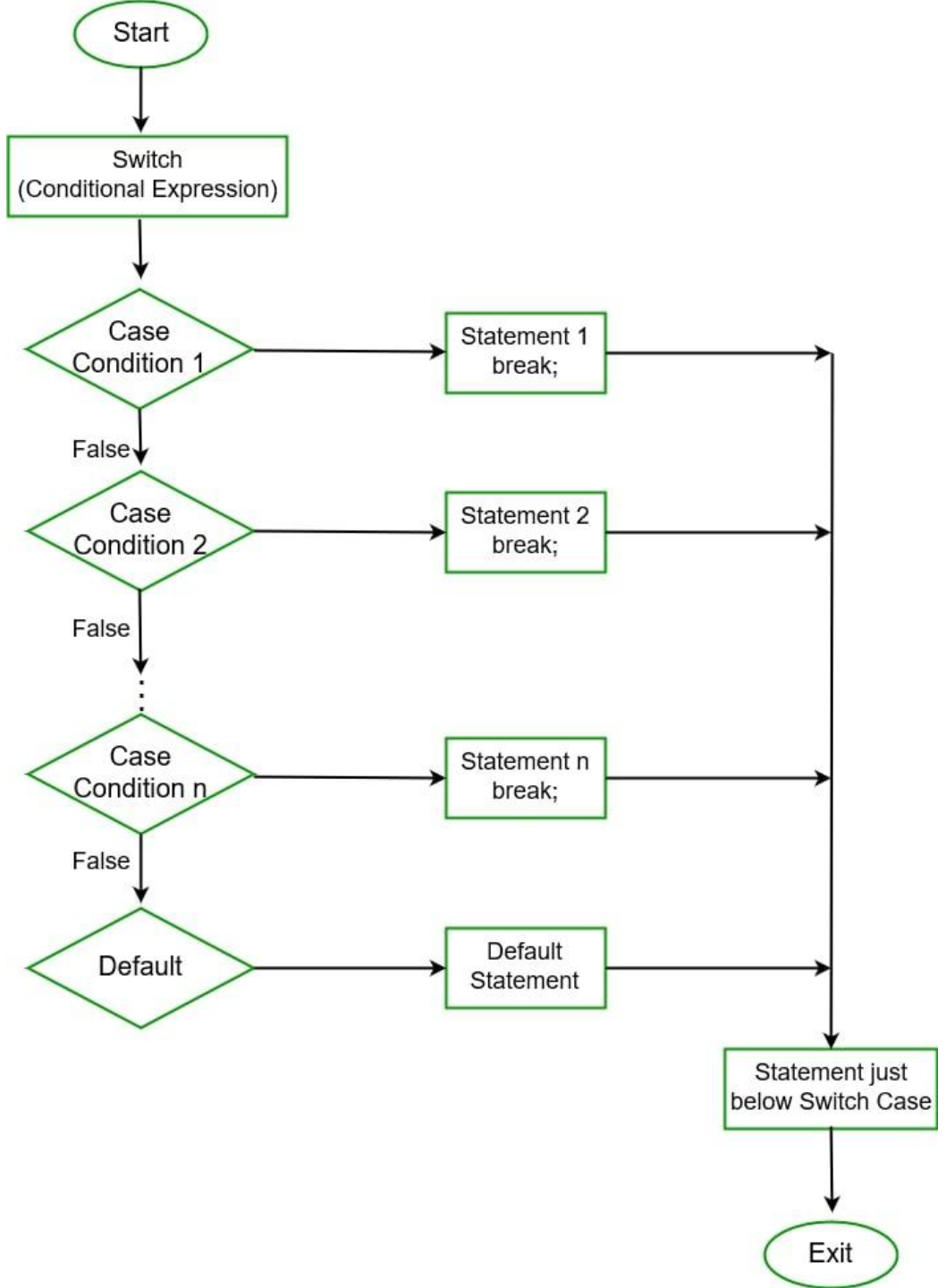
Синтаксис: Switch-case

```
switch(expression)
{
   // case statements values must be of same type of expression
   case 1 :
      // Statements
      break; // break is optional

   case 2 :
      // Statements
      break; // break is optional

   default :
      // Statements
}
```

Важные правила для операторов switch

1. Может быть любое количество случаев, но повторяющиеся значения случаев не допускаются.
2. Значение для case должно иметь тот же тип данных, что и переменная в switch.
3. Значение для case должно быть постоянным. Переменные не допускаются.
4. Оператор break используется внутри switch для завершения последовательности операторов.
5. Каждый оператор case может иметь оператор break, который является необязательным. Когда управление достигает оператора break , оно переходит к элементу управления после выражения switch. Если оператор break не найден, выполняется следующий случай.
6. Оператор default является необязательным и может появляться в любом месте внутри блока переключателя. В случае, если он не в конце, то после оператора default необходимо оставить оператор break, чтобы пропустить выполнение следующего оператора case

Блок-схема оператора Switch-Case



Вложенный Switch

Мы можем использовать Switch как часть последовательности операторов внешнего Switch.

Это называется вложенным Switch. Поскольку оператор switch определяет свой собственный блок, между константами case во внутреннем Switch и во внешнем Switch не возникает конфликтов.

```
switch (year) {
        case 1:
        …
        break;
        case 2:
        // Вложенный Switch
            switch (Branch) {
        case “TelRan Berlin":
              case “TelRan Israel":
        …
        break;
        case “TelRan USA":
                    …
                    break;

        default:
        …
        }
}
```

switch VS if else

Оператор switch обычно более эффективен, чем набор вложенных if’ов.
Что использовать, операторы if-then-else или оператор switch?

switch	if-else
проверяет выражения, основанные только на одном целом, перечисляемом значении или объекте	может проверять выражения на основе диапазонов значений или условий
оператор switch при компияции, компилятор создает «таблицу переходов», которую он будет использовать для выбора пути выполнения в зависимости от значения выражения, потому что компилятор знает, что case-константы все одного типа	в то время как в случае if-выражений компилятор таких знаний не имеет и это может работать дольше
операторы switch отлично подходят для фиксированных значений данных	условные переходы if-else отлично подходят для переменных условий, которые приводят к логическому значению
оператор switch может оказаться быстрее все элементы получают одинаковое время доступа	для достижения последнего элемента требуется гораздо больше времени, поскольку оценивается каждое предыдущее условие
переключатель выглядит намного чище	спорный момент, if в некоторых случаях выглядит опрятнее

Не используйте switch в своем коде!
☺️

Особенности

Выражение или оператор?

До Java12 switch был оператором - императивной конструкцией, которая просто выполняет соответствующий блок

После Java12 - можно использовать как выражение, которое возвращает значение

Двоеточия или стрелки? (: vs ->)

```
int day = 17;
String dayStr;

switch (day) {
    case 17:
        dayStr = "понедельник";
        break;
    case 18:
        dayStr = "вторник";
        break;
    default:
        dayStr = "error";
}

System.out.println(dayStr);
```

```
int day = 17;

String dayStr = switch (day) {
    case 17 -> dayStr = "понедельник";
    case 18 -> dayStr = "вторник";
    default -> dayStr = "error";
};

System.out.println(dayStr);
```

Ternary operator

Тернарный оператор Java — единственный условный оператор, который принимает три операнда.
Это однострочная замена инструкции if-then-else.
Мы можем использовать тернарный оператор вместо условий if-else.
Условный оператор занимает меньше места и помогает писать операторы if-else кратчайшим возможным способом.

Синтаксис:

```
variable = Expression1 ? if true : else;
```

```
If (Expression1)
{
   variable = Expression2;
}
else
{
   variable = Expression3;
}
```