**Билет 18. Прусаченкова В.Д.**

1. Как преобразовать строку в число?
2. Как сравнить значения двух строк?
3. Что является монитором при вызове нестатического и статического метода?
4. Параметры всегда возвращаются в виде символьных строк. Поэтому символьные строки нужно преобразовать в числовой тип, если они применяются именно для этого. Это делается стандартным способом с помощью соответствующего метода, например, parselnt () из класса Integer. Так, если требуется добавить параметр размера шрифта, с этой целью можно написать следующий код HTML-разметки:

<applet code="FontParamApplet.class" width="200" height="200">

<param name="font" value="Helvetica"/>

<param name="size" value="24"/>

</applet>

А в приведенном ниже фрагменте кода показано, каким образом считывается целочисленный параметр.

public class FontParamApplet extends JApplet

{

public void init()

{

String fontName = getParameter("font”);

int fontSize = Integer.parselnt(getParameter("size"));

}Кроме того, чтобы гарантировать правильность заданных параметров, следует

проверить, не пропущено ли значение параметра s iz e . Для этого достаточно сравнить его с пустым значением n u ll, как показано в приведенном ниже примере кода.

int fontsize;

String sizestring = getParameter("size");

if (sizestring == null) fontsize = 12;

else fontsize = Integer.parselnt(sizestring);

1. Чтобы сравнить символьные строки по длине, достаточно определить класс, реализующий интерфейс Comparator<String>:

class LengthComparator implements Comparator<String> {

public int compare(String first, String second) {

return first.length() - second.length();

}

}

Чтобы произвести сравнение, фактически требуется получить экземпляр данного класса следующим образом:

Comparator<String> сощр = new LengthComparator();

if (comp.compare(words[i], words[j]) >0) ...

Сравните этот фрагмент кода с вызовом words [i] .compareTo (words [ j ]). Метод compare () вызывается для объекта компаратора, а не для самой символьной строки.

Несмотря на то что у объекта типа LengthComparator отсутствует состояние, его экземпляр все же требуется получить, чтобы вызвать метод compare ( ) . Ведь этот метод не является статическим.

1. Для нестатического метода — текущий объект this. Для статического метода — объекта типа Class, соответствующий классу, в котором определен этот метод.

Статическими называют методы, которые не оперируют объектами. Например,

метод pow () из класса Math является статическим. При вызове метода Math. pow (х, а)

вычисляется степень числа ха. При выполнении этого метода не используется ни

один из экземпляров класса Math. Иными словами, у него нет неявного параметра this. Это означает, что в статических методах не используется текущий объект

по ссылке this. Неявный параметр в объявлении метода не приводится. В каждом методе ключевое слово this обозначает неявный параметр. По желанию метод raiseSalary () можно было бы переписать следующим образом:

public void raiseSalary(double byPercent)

{

double raise = t h i s . salary \* byPercent / 100;

t h i s . salary += raise;

}

Некоторые предпочитают именно такой стиль программирования, поскольку

в нем более отчетливо различаются поля экземпляра и локальные переменные.