39. ↑ Анонимные методы и шаблоны появились в Delphi 2009

42. ↑ Создание объектов на стеке/Delphi. В Delphi имеется 2 объектных модели - старая (унаследована из Turbo Pascal) и новая. Создание объектов на стеке возможно только в старой объектной модели
43. ↑ Поддержка Unicode в идентификаторах/C++. Доступно в компиляторах от MS, начиная с MSVS 2005

47. ↑ Контроль выхода за границы массива/C++. В STL входит шаблон std::vector, который следует использовать вместо массивов языка Си, оставленых для совместимости. Благодаря возможности перегрузки оператора индексации имеется возможность встраивать в него контроль границ (в std::vector также поддерживается контроль границ в итераторах).
48. ↑ Контроль выхода за границы массива/Java. При выходе за границы массива в Java автоматически генерируется и выбрасывается исключение ArrayIndexOutOfBoundsException.

51. ↑ Сборка мусора/Ada. Только на некоторых платформах (.NET и JVM) или при помощи библиотек (AdaCL:GC). Тем не менее, практически все программы на Ada могут работать как с ним, так и без него. В этом смысле к сборке мусора применительно к Аде следует относиться не как к инженерному решению, а как к оптимизации управления памятью.
52. ↑ [1] [2] Сборка мусора/C и C++. В стандарте языка и в стандартных библиотеках нет сборки мусора. Однако существуют сборщики мусора для C и C++ в виде библиотек. Например, BoehmGC
53. ↑ Сборка мусора/Delphi. Если не считать Delphi.net

55. ↑ Целые числа произвольной длины/Java. Для представления таких чисел в стандартном наборе присутствуют специальные классы - BigInteger и BigDecimal

64. ↑ Именованные параметры в Delphi могут использоваться при вызове OLE:
```
Word.Openfile(filename='1.doc')
```

67. ↑ Локальные функции/C. Поддерживаются в компиляторе gcc как нестандартное расширение языка. [6]
68. ↑ Локальные функции/Java. Внутри метода можно определять безымянные (анонимные) локальные классы, которые фактически позволяют создавать экземпляры объектов, перекрывающие методы своего класса.
69. ↑ Появились в Delphi2009, как анонимные функции
70. ↑ Кортежи/C++. В C++ кортежи реализуются в стандартной библиотеке (появились в TR1 (англоязычный раздел), до этого была в boost'e)).

73. ↑ "Query Comprehension" можно считать за List Comprehension только с большой натяжкой
74. ↑ Цикл foreach/Ada. Методы Iterate и Reverse_Iterate различных контейнеров, входящих в библиотеку Ada.Containers, являющуюся неотъемлимой частью языка.
75. ↑ Цикл foreach/C++. Алгоритм for_each входит в библиотеку STL, являющуюся неотъемлимой частью языка.

79. ↑ Список изменений в языке Delphi с 7 версии
80. ↑ Информация о типах в runtime/Ada. Точный тип узнать можно (Ada.Tags), но полной поддержки отражения в языке нет. Можно узнать имя, предков, интерфейсы, сериализовать объект. Нельзя запросить список методов.
81. ↑ [1] [2] Информация о типах в runtime/C++. Можно сравнить типы на точное совпадение, узнать имя типа (typeid) или его размер (sizeof). Однако полноценной поддержки отражения в языке нет - нельзя перечислить предков типа, члены данных или методы либо сериализовать объект. Статически можно привязывать и получать дополнительную информацию о типах с помощью специализированных шаблонов, как, например, std::numeric_limits. Все перечисленные средства работают и с аргументами шаблонов.
82. ↑ Инструкция goto/Java. Является зарезервированным словом.

88. ↑ Блок finally/Ada. В стандарте языка finally нет, но существуют библиотеки, реализующие функционал finally. Используются крайне редко, это скорее proof of concept(англ.).

91. ↑ При помощи нескольких последовательных catch

93. ↑ Легковесные процессы/Java. Вплоть до Java 1.1.

96. ↑ Контрактное программирование/Java. На основе аннотаций Java 5, используя библиотеку OVal и аспектный компилятор AspectJ.
97. ↑ [1] [2] [3] [4] [5] Реализуется сторонними библиотеками
98. ↑ Только совместное использование посредством виртуального наследования