1.
The programming languages where the type is checked at compile time are called statically typed languages while the programming languages where the type is checked at runtime are called dynamically typed programming languages.
If a programming language is strict regarding the specifications of data types they are called strongly typed programming languages while if a programming language doesn't demand much regarding the specifications of the data types they are called loosely typed languages.
Java can be considered as both statically typed and dynamically typed languages because the type is checked during both compile time and run time. And java is strict regarding the data type therefore it is a strongly typed programming language.

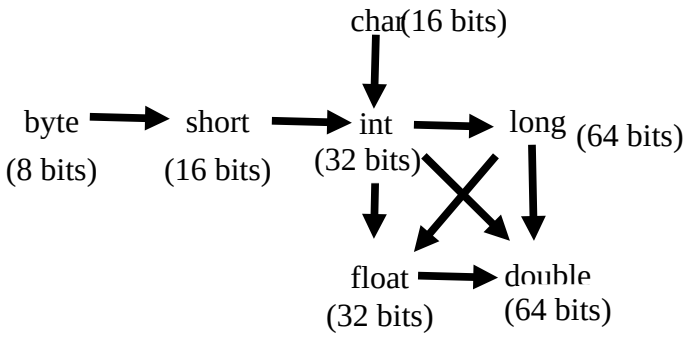
2.
Case sensitive programming languages are the languages where upper case and lower case letters are treated as distinct when using variables , functions and other identifiers. In case insensitive programming languages upper and lower cases are treated equivalently. In case sensitive – insensitive programming languages some identifiers may treat upper case and lower case as distinct and some identifiers may be case insensitive.
Java is a case sensitive language.

3.
Identity conversion is the conversion between two instances of the same data type.
Example:

```
1.int value1 = 10;
  int value2;
  value2 = value1; //identity conversion

2.String name1 = "Kasun";
  String name2;
  name2 = name1;
```

4.
Primitive widening conversion is the conversion from a data type with small bit size to a data type with larger bit size (in the direction of arrow in the following diagram from byte data type to char type). Therefore no information is lost. When converted to float or double data types, precision may be lost because they store bit values according to IEEE- 754 standard.



Example :
1. byte b1 = 10;
short s1 = b1; // widening primitive conversion
2. short s2 = 51;
int intnum = s2; // widening primitive conversion
3. float f1 = 10.5f;
double d1 = f1;

5.
Compile time constants are computed when the code is being compiled but run time constants are computed when the code is running (by the JVM). The value of the run time constants may change every time the code is run but the value of a compile time constant will never be changed after the assignment.

Examples :
final int x = 5; //compile time constant.
final int y = 3* (int) Math.random(); //run time constant. Value of ‘y’ may be changed because value of random number changes every time it runs.

6.
Implicit narrowing primitive conversion is conversion from a data type with higher bit size to a type with lower bit size automatically. Explicit narrowing primitive conversion is done by hard coding when converting from a data type with higher bit size to a data type with lower bit size.

In order to occur implicit narrowing conversion it should be an assignment context and two conditions must be fulfilled.

Condition 1 : Assigning value should be a compile time constant.
Condition 2 : The value should be in the range of the type it assigned.

7.
When converting from a long type to a float type, there may be loss of precision. The range of float is broader than long therefore although it may result in loss of precision, the magnitude is representable according to the IEEE – 754. Therefore there will be no exception when compiling.

8.
In Java, int and double types are set as default data types for integer literals and floating point literals because of the design goals of Java language.

9.
Implicit narrowing primitive conversion are only allowed among byte, char, int, and short because their sizes and ranges are well defined causing less unintended data losses.

10.
Widening and narrowing primitive conversion is that the conversion is occurred in two steps as first widening and then narrowing conversion.