1.

Statically Typed Language

- Compile time type checking.
- Compiled & intermediate languages only.(because , doesn’t have run time)
- For making enterprise application.

In this type of language, once a variable is assigned a type, it can’t be assigned to some other variable of a different type. If we try to do so, the compiler will raise some errors, and we need to fix them. Hence, for a declared variable, the data type is fixed.

Dynamically Typed Language

- Run time type checking.

A dynamically-typed language has the capability to identify the type of each variable during run-time. In these languages, variables are bound to objects at run-time using assignment statements.

Strongly Typed Language

- Strictly care the type of language.
- For making enterprise application.
- Java concerns the type.

A strongly typed programming language is one in which each type of data, such as integers, characters, hexadecimals and packed decimals, is predefined as part of the programming language, and all constants or variables defined for a given program must be described with one of the data types. Certain operations might be allowable only with specific data types.

Loosely Typed Language

- Not strictly care the type of language.

A programming language is loosely typed, or weakly typed,when it does not require the explicit specification of different types of objects and variables. The "looser" typing rules in weakly typed programming languages can produce erroneous or unpredictable results. It can execute implicit type conversions at runtime.

Java is a both statically & dynamically typed language.

2.

Case Sensitive

the phrase used to describe a programming languages ability to distinguish between upper and lower case versions of a letter in the language's character set.

Case Insensitive

Case insensitivity describes a programming language's inability to distinguish between upper and lower case versions of a letter in the language's character set. For example, the letter 'c' is considered to be the same as the letter 'C', even though internally the computer sees a 'c' as 99, and a 'C' as 67.

Case-sensitive means that the computer program only matches values with the same case (lower/upper). Case-insensitive means the program ignores case and matches values regardless of their lower or upper case letters, e g. “My Name” = “my name”).

Java is case-sensitive because it uses a C-style syntax.

3.
A conversion from a type to that same type is permitted for any type.

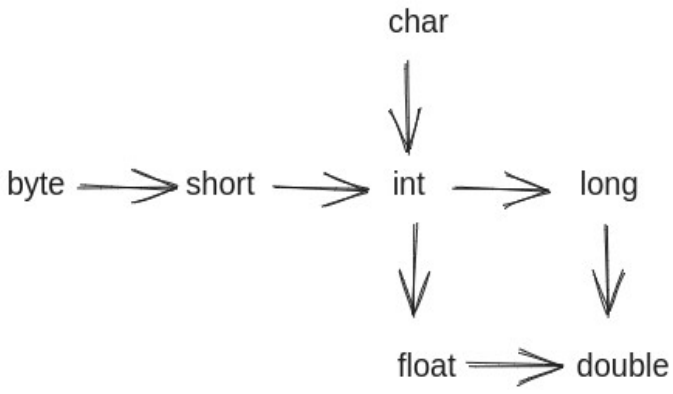
This may seem trivial, but it has two practical consequences. First, it is always permitted for an expression to have the desired type to begin with, thus allowing the simply stated rule that every expression is subject to conversion, if only a trivial identity conversion. Second, it implies that it is permitted for a program to include redundant cast operators for the sake of clarity.

```
public class ex{
   public static void main(String[] args) {

      int x;
      int y=5;

      x=y;
   System.out.println(x);

   }
}
```

```
public class ex{
   public static void main(String[] args) {

      double x;
      double y=10;

      x=y;
   System.out.println(x);

   }
}
```

4.

- `byte` to `short`, `int`, `long`, `float`, or `double`
- `short` to `int`, `long`, `float`, or `double`
- `char` to `int`, `long`, `float`, or `double`
- `int` to `long`, `float`, or `double`
- `long` to `float` or `double`
- `float` to `double`



5.

```
public static void main(String[] args) {
	final int MY_CONST = 10;
	final int MY_CONST2 = 10 * (int) Math.random();

	byte myByte2 = MY_CONST;	//	Compile time constant
	byte myByte3 = MY_CONST2;	//	Run time constant
```

In the compile time constant , compiler can identify the value @ the compile time.

In the run time constant , compiler cannot identify the value @ the compile time .

6.

- Automatic conversion (casting) done by Java compiler internally is called implicit conversion or implicit type casting in java.

  Implicit casting is performed to convert a lower data type into a higher data type.

- The conversion of a higher data type into a lower data type is called narrowing conversion.

  Since this type of conversion is performed by the programmer, not by the compiler automatically, therefore, it is also called explicit type casting in java. It is done to convert from a higher data type to a lower data type.

<u>conditions</u>

- should be compile time constant.
- RHS bit value of the constant should be inside the LHS bit range.

- Cannot assign variables.
- Cannot assign negative values to unsign data types “char”.

7.

From long to double, may result in loss of precision- that is, the result may lose some of the least significant bits of the value. In this case, the resulting floating-point value will be a correctly rounded version of the integer value, using IEEE 754 round-to-nearest mode

9.223372e18 is 9223372000000000000. Compare:

```
9223372036854775807 - the long value
9223372000000000000 - the float value
```

This is because float can sacrifice precision for range of value, because it stores a base value that it raises to an exponent.

8.

The int data type is a 32-bit signed two's complement integer. Its value-range lies between -2,147,483,648 (-2^31) to 2,147,483,647 (2^31 -1) (inclusive). Its minimum value is - 2,147,483,648and maximum value is 2,147,483,647. Its default value is 0.

The int data type is generally used as a default data type for integral values unless if there is no problem about memory.

The double data type is a double-precision 64-bit IEEE 754 floating point. Its value range is unlimited. The double data type is generally used for decimal values just like float. The double data type also should never be used for precise values, such as currency. Its default value is 0.0d.

9.

long, double, float are not involved in automatic narrowing primitive conversion.

There is no 2’s complement in float and double.

byte, short, and char are all smaller in size than int

Implicit narrowing means converting a larger data type to a smaller data type without any explicit casting. Java allows this narrowing conversion for int, byte, short, and char. Implicit narrowing is not allowed for other primitive types like long and float because it can result in a loss of data due to their larger ranges and precision.

10.

byte to char

First, the byte is converted to an int via widening primitive conversion and then the resulting int is converted to a char by narrowing primitive conversion.

Short type value can be converted directly to char value.