TODO -

- Conversion Numeric Types - Core Java - pg 56
- Exception hierarchy - Core Java - pg 374
- Collections - Core Java - pg 496
- Collection interfaces - Core Java - pg 493

Table 3.1 Types (page 43)

Type	Storage Requirement	Range(inclusive)
byte	1 byte	-128 to 127
short	2 bytes	-32,768 to 32,767
int	4 bytes	-2,147,483,648 to 2,147,483,647 (just over 2 billion)
long	8 bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

Table 3.2 Floating-Point Types (page 44)

Type	Storage Requirement	Range(inclusive)
float	4 bytes	Approximately +- 3.40282347E+38F (6-7 significant decimal digits)
double	8 bytes	Approximately +- 1.79769313486231570E+308F (6-7 significant decimal digits)

Table 3.3 - Escape Sequences for Special Characters (page 46)

Escape Sequence	Name	Unicode Value
\b	Backspace	\u0008
\t	Tab	\u0009
\n	Linefeed	\u000a
\r	Carriage return	\u000d
"	Double quote	\u0022
'	Single quote	\u0027
\	Backslash	\u005c

Table 3.4 - Operator Precedence (page 61)

Operators	Associativity
[] , () (method call)	Left to right
! ~ ++ -- + (unary) - (unary) () (cast) new	Right to left
* / %	Left to right

Operators	Associativity
+ -	Left to right
<< >> >>>	Left to right
< <= > >= instanceof	Left to right
== !=	Left to right
&	Left to right
^	Left to right
|	Left to right
&&	Left to right
||	Left to right
?:	Right to left
= += -= *= /= %= &= |= ^= <<= >>= >>>=	Right to left

Table 6.2 Common Functional Interfaces (page 337)

Functional Interface	Parameter Types	Return Type	Abstract Method Name	Description	Other Methods
Runnable	none	void	run	Runs an action without arguments or return value	
Supplier	none	T	get	Supplies a value of type T	
Consumer	T	void	accept	Consumes a value of type T	andThen
BiConsumer<T, U>	T,U	void	accept	Consumes values of types T and U	andThen
Function<T,R>	T	R	apply	A function with argument of type T	compse, andThen, identity
BiFunction<T,U,R>	T,U	R	apply	A function with arguments of types T and U	andThen
UnaryOperator	T	T	apply	A unary operator on the type T	compose, andThen, identity
BinaryOperator	T,T	T	apply	A binary operator on the type T	andThen, maxBy, minBy
Predicate	T	boolean	test	A boolean-valued function	and, or, negate, isEqual

Functional Interface	Parameter Types	Return Type	Abstract Method Name	Description	Other Methods
BiPredicate<T, U>	T,U	boolean	test	A boolean-valued function with two arguments	and, or, negate

Table 6.5 Functional Interfaces for Primitive Types (page 338)

`p,q is int, long, double`

`P,Q is Int, Long, Double`

Functional Interface	Parameter Types	Return Type	Abstract Method Name
BooleanSupplier	none	boolean	getAsBoolean
PSupplier	none	p	getAsP
PConsumer	p	void	accept
ObjPConsumer	T,p	void	accept
PFunction	p	T	apply
PToQFunction	p	q	applyAsQ
ToPFunction	T	p	applyAsP
ToPBiFunction<T,U>	T,U	p	applyAsP
PUnaryOperator	p	p	applyAsP
PBinaryOperator	p,p	p	applyAsP
PPredicate	p	boolean	test

Table 9.1 Concrete Collections in the Java Library

Collection Type	Description
ArrayList	An indexed sequence that grows and shrinks dynamically
LinkedList	An ordered sequence that allows efficient insertion and removal at any location
ArrayDeque	A double-ended queue that is implemented as a circular array
HashSet	An unordered collection that rejects duplicates
TreeSet	A sorted set
EnumSet	A set of enumerated type values
LinkedHashSet	A set that remembers the order in which elements were inserted
PriorityQueue	A collection that allows efficient removal of the smallest element
HashMap	A data structure that stores key/value associations
TreeMap	A map in which the keys are sorted

Collection Type	Description
EnumMap	A map in which the keys belong to an enumerated type
LinkedHashMap	A map that remembers the order in which entries were added
WeakHashMap	A map with values that can be reclaimed by the garbage collector if they are not used elsewhere
IdentityHashMap	A map with keys that are compared by ==, not .equals()

Table A.1 - Java Keywords (page 839-842 Appendix A)

Keyword	Meaning
abstract	An abstract class or method
assert	Used to locate internal program error
boolean	The Boolean typ
break	Breaks out of a switch or loop
byte	The 8-bit integer type
case	A case of a switch
catch	The clause of a try block catching an exception
char	The Unicode character type
class	Defines a class type
const	Not used
continue	Continues at the end of a loop
default	The default clause of a switch, or a default method in an interface
do	The top of a do/while loop
double	The double-precision floating-number type
else	The else clause of an if statement
enum	An enumerated type
exports	Exports a package of module (restricted)
extends	Defines the parent class of a class, or an upper bound of a wildcard
final	A constant, or a class or method that cannot be overridden
finally	The part of a try block that is always executed
float	The single-precision floating-point type
for	A loop type
goto	Not used
if	A conditional statement

Keyword	Meaning
implements	Defines the interface(s) that a class implements
import	Imports a package
instanceof	Tests if an object is an instance of a class/interface
int	The 32-bit integer type
interface	An abstract type with methods that a class can implement
long	The 64-bit long integer type
native	A method implemented by the host system
new	Allocated a new object or array
null	A null reference(note that null is technically a literal, not a keyword)
module	Declared a module (restricted)
open	Modifies a module declaration (restricted)
opens	Opens a package of a module (restricted)
package	A package of classes
private	A feature that is accessible only by methods of this class
protected	A feature that is accessible only by methods of this class, its children, and other classes in the same package
provides	Indicates that a module uses a service (restricted)
public	A feature that is accessible by methods of all classes
return	Returns from a method
short	The 16-bit integer type
static	A feature that is unique to a class or interface, not to instances of a class
strictfp	Use strict rules for floating-point computations
super	The superclass object or constructor, or a lower bound in a wildcard
switch	A selection statement
synchronized	A method or code block that is atomic to a thread
this	The implicit argument of a method, or a constructor of this class
throw	Throws an exception
to	A part of an exports or opens declaration (restricted)
throws	The exceptions that a method can thro
transient	Marks data that should not be persistent
transitive	Modifies a requires declaration (restricted)

Keyword	Meaning
try	A block of code that traps exceptions
uses	Indicates that a module uses a service (restricted)
var	Declares a variable whose type is inferred (restricted)
void	Denotes a method that returns no value
volatile	Ensures that a field is coherently accessed by multiple threads
with	Defines the service class in a provides statement (restricted)
while	A loop