1 Types

1.1 Primitive Types

prim. type	corresp. class	ex. literals	default
boolean	Boolean	true, false	false
char	Character	A, 3, \n	\u0000
int	Integer	37, -3, 12345	0
double	Double	3.1416, 1E-10	0.0

1.2 Reference Types

- arrays:
  Example: `int[], Ball[], double[][]`
  Create Object: `int[] a = new int[10];`
  Initialize: `double[][] data = {{1,3},{4,8}};`
  Indexing: `a[i], 0 <= i < a.length`.

- class types:
  The object must be created with a constructor:
  `Ball b = new Ball(10,20,Color.RED);`
  `LifeModel model = new LifeModel(50,50);`

- interface types:
  Declare methods with result type, name and parameter types but without implementation.
  Objects of this type cannot be constructed but classes can implement interfaces.

2 Variables

Variables must be declared: `int x; double[] ys;`

Initialization
Instance variables and array elements will be initialized with

- the default value for the primitive type
- `null` for reference types.

Local variables must be initialized with the declaration.

3 Expressions

Expressions are build from variables, literals, operators and method calls.

Binary operators

operator	arg type	res type	comment
*, /, %	number	number	
+, -	number	number	
+	String	String	concatenation
<, <=, >, >=	number	boolean	
==, !=	any	boolean	
&&	boolean	boolean	logical and
||	boolean	boolean	logical or

number means a numeric primitive type.
any means arbitrary type.

Unary operators

operator	arg type	res type	comment
!	boolean	boolean	arithmetic negation
-	number	number	logical negation

String concatenation
If we have type String `a + b` means concatenation. `b` is first converted to `String`.

Type convertion
Convertion of on expression of type `int` to `double` happens automatically when is needed. In the opposite direction we need an explicite convertion: (int) e.

The conversion from a class to its super class is automatic. The opposite requres an explicit conversion: (SubClass) o.

4 Method calls; parameter passing

- The values of the arguments are calculated before the method call;
- The parameters are initialized to these values before the method body is executed. For reference types, it means that the reference and not the value is copied.
- The result of the function goes back to the call place and is copied to the destination variable.

5 Statements

l-expr = r-expr;	method-name($e_1, \ldots e_n$);
var++;	var−;
break;	continue;
return;	return expr ;

if (test) {
statements
}
	if (test) {
	statements
	} else {
	statements
	}

while (test) {
statements
}
	do {
	statements
	} while (test);

for (init; test; upd) {
statements
}
	for (type var : expr) {
	statements
	}

switch (expr) {
case lit1: stmt
...
case litN: stmt
}
	try {
	statements
	} catch (exc-type var) {
	statements
	}

6 Classes

- Library of functions (ex. Math, Arrays) - contains only static methods (functions, procedures)
- Templates from which objects are created (most classes). Usually does not contain static methods. Instead there are instance variables, constructors and methods.
- The main class in an application contains:

  `public static void main(String[] args)`

7 Libraries of Functions

All methods in this section (and only in this) are static.

java.lang.Math
Among others: `abs`, `max`, `min`, `sin`, `cos`, `exp`, `log`, `pow`, `sqrt`, `round`, `floor`, `random`.

java.util.Arrays

```
int binarySearch(X [] a, X key)
X [] copyOf(X [] orig, int len)
boolean equals(X [] a1, X [] a2)
void fill(X [] a, X val)
void sort(X [] a)
String toString(X [] a)
```

java.lang.Character

```
int digit(char ch, int radix)
char forDigit(int dig, int radix)
boolean isDigit(char ch)
boolean isLetter(char c)
char toLowerCase(char ch)
char toUpperCase(char ch)
```

java.lang.Double

```
double parseDouble(String str)
```

java.lang.Integer

```
int parseInt(String str)
```

java.lang.System

```
InputStream in
PrintStream out
PrintStream err
void exit(int status)
long currentTimeMillis()
```

8 Object and String

java.lang.Object

```
boolean equals(Object obj)
String toString()
```

java.lang.String

```
implements Comparable
char charAt(int index)
byte[] getBytes()
int indexOf(String str)
int length()
String subString(int beginIndex)
String toLowerCase()
String toUpperCase()
```

9 Input and Output

java.util.Scanner

```
Scanner(File source) throws FileNotFoundException
Scanner(InputStream source)
boolean hasNextX()
X nextX()
String nextLine()
boolean hasNextLine()
```

Here `X` is one of the primitive types, i.e., Int, Double, Float, etc.

java.io.PrintStream

```
PrintStream(File file) throws FileNotFoundException
void print(X x)
void println(X x)
print(String str)
println(String str)
```

10 Collections

interface java.util.Iterator

```
boolean hasNext()
Object next()
void remove() // optional method
```

interface java.lang.Iterable

```
Iterator iterator()
```

interface java.util.Collection
extends Iterable

```
boolean add(T e)
int size()
```

interface java.util.List
extends Collection

```
void add(int index, Object element)
Object get(int index)
Object set(int index, Object element)
Object remove(int index)
```

class java.util.ArrayList
implements List

class java.util.LinkedList
implements List

interface Map

```
Object get(Object key)
Object put(Object key,Object value)
Object remove(Object key)
```

class HashMap
implements Map