Practice exam:

* many programs contain bugs.
* 2 + 3 \* 4.0 == 14.0
* sqrt is a static method in the Math class object
* False: Java programs are compiled by the Java interpreter. Else True:

- Source code is stored in files with a .java extension.

- Source code is compiled to produce object code.

- Object code is stored in files with a .class extension.

- Java programs are executed by a Java interpreter.

* 9. public void demo (int I, String s)

A. demo( 5, “5.2” );

demo( (int) 3.7, "Hello" );

demo( x++, "Hello".substring(1) );

demo( x / (2 + 3), "A" );

* Int, double, char, and Boolean are all primitive data types. String is an object
* 3. The values stored in the data fields of an object, and the methods of an object, are sometimes called by the terms:

A. state and behaviour

* 4. The header of a method contains information about its:

A. visibility, input parameters, and return type

* for(int x = 1; x <= 5; x++){

System.out.print((x\*x)+” “);

}

Output is: 1 4 9 16 25

* Java contains a fixed number of primitive types.

Any class in java can be a reference type

Data fields can contain primitive values or references.

Local variables can contain primitive values or references.

* A mutator usually accepts input parameters.

A constructor has the same name as the class it is defined in.

An accessor usually includes a return statement.

A private data field can be accessed by methods in the same class.

* Don’t reuse variable names in the same program as it can be confusing

Keep variables as local as possible.

Declare variables at the start of the block.

Consider the preconditions and postconditions of methods.

Make data fields private and use accessor and mutator methods.

* The header of a for loop specifies an initialisation, a condition, and an update.

Any loop can be completely contained within any other loop.

* DrJava is the IDE used in COMP103. IDE stands for Integrated Development Environment
* A program must have at least one class.

A class can contain data fields and methods.

The order of methods in a class does not matter.

The order of statements in a method matters

* 8. int a = 2; char c = ‘a’; double d = 3.2; System.out.println(a + “ “ + c + “ “ + (int) d);

A. prints out: 2 a 3

* 11. MyClass b = a;

A. It creates and initialises a variable b to contain a reference to the same object as a

* 12. if s and t both represent different objects of the string “comp103” then:

A. s.equals(t) and t.equals(s) are true. s = t and s == t are false

* Class files are NOT specific to a particular kind of computer.

The compiler makes class files from source files

The interpreter runs the class files.

The interpreter and the compiler are part of the Java SDK.

The interpreter produces computer specific executable instructions

* "Class members" are members of class objects declared with the keyword static.

Public members of a class may be accessed from / used by other classes.

* (int) (a + b); computes the sum of the two doubles a and b then converts it into an int
* Data fields are part of the class and can be accessed by any method in the class.

If not explicitly initialised data fields are initialised to "zero states" automatically. (D) We specify the inputs that a method expects using formal parameters. (E) Java has reserved words which cannot be used as identifiers for variables.

* public boolean sameSign(int in1, double in2) {// body }

sameSign is the method name and the method returns a Boolean result. The return type is Boolean. In1 and in2 are formal parameters

* Assume that a is a variable referring to an object of type MyClass. What will be printed by the following statement? System.out.println( a );

Whatever string is returned by the toString method in MyClass.

* Service methods typically form the public interface to a class

Support methods typically have private visibility.

Support methods are typically called by service methods.

Service methods may alter the values stored in data fields.

* s.substring(s.indexOf('b'))

part of s from the first occurring 'b' to the end

* public MyClass(int i, int j) // this is a constructor
* The case selector variable can be of type char.

The case selector variable cannot be of type double.

Individual cases can use break to exit, or omit it to continue to the next case.

Switch statements may end with an optional default case.

* The counter variable must be declared.

The counter variable must be initialised.

The counter variable must be updated.

The counter variable must be tested in the loop condition.

* test = x > 1; // is the same as

if (x > 1) test = true;

else test = false;

* int sum = 5, n = 3;

for (int c = 0; c <= n; c++) {

sum += c;

} // sum == 11

* Any drawing in a GUI must be automatically redrawn if windows are moved, uncovered, minimised, opened and so on. Drawing code should all be in, or be called by, a component's paint or paintComponent method.

String Libraries:

* string.length()
* string.indexOf()
* string.substring()
* string.replace()
* string.charAt()
* string.toUpperCase()
* string.toLowerCase()
* string.equals()
* string.equalsIgnoreCase()
* string.startsWith()

Math Libraries:

* Math.PI
* Math.E
* Math.sqrt()
* Math.sin()
* Math.max()
* Math.min()
* Math.abs()
* Math.round()
* Math.ceil()
* Math.floor()
* Math.pow()
* Math.exp()

Scanner:

* Import java.util.Scanner;
* Scanner sc = new Scanner(System.in);
* sc.hasNext< Boolean, Double, Int, Line >();
* sc.next< Boolean, Double, Int, Line >();

Random:

* Import Java.util.Random;
* Random rand = new Random();
* rand.nextInt(int)< + number >;

Switch:

* switch(< int, char >){

case < int1, char1 >:

case < int2, char2 >:

statement(s);

break;

default:

statement(s);

}

While:

* < if, while >(conditions) statement;

< if, while >(conditions){

statement(s)

}

Do:

* do{

statement(s);

} while(condition);

For:

* for(initialisation; condition; update) statement;

for(initialisation; condition; update){

statement(s);

}

variable = (boolean expression) ? < statement1 > : < statement2 >;

// If bool = true statement1 is executed

// If bool = false statement2 is executed

// Expressions are literals, variables, methods, assignments, calculations...

// casting is forcing a variable to be of another type

char c = (char) 97;

int i = (int) 15.32;

Gui:

Import javax.Swing.\*;

Import java.swt.\*;

setColor(Color.< black, ... >);

drawLine(x1, y1, x2, y2);

< drawRect, drawOval, fillRect, fillOval >(x, y, width, height);

drawString(String, x, y);

< drawArc, fillArc >(x, y, width, height, startAngle, arcAngle);

[==, ||, &&, <=, <, >=, >, !=]

DecimalFormat dec = new DecimalFormat("0.##");

dec.format(double);

String [] words = {"The","quick","brown","fox"};

for (String s : words)

System.out.println(s + "!")