LONDON CAPITAL COMPUTER COLLEGE

Diploma in Database Developer (991) - JDeveloper

Prerequisites: Basic knowledge of basic programming.	**Corequisites:** A pass or higher at Diploma level
Aim: This course takes candidates through basic Java syntax, how to design simple programs and classes used in the development of Java applications and applets.	
Required Materials: Student study materials	**Supplementary Materials:** Recommended textbooks and lecture notes.
Special Requirements: This is a hands-on course, hence practical use of computers is essential. Requires intensive lab work outside of class time.	
Intended Learning Outcomes:	**Assessment Criteria:**
1. Discover the history of Java. Identify the basics of the Java environment. Define object-oriented programming.	1.1 Familiarise with different types of programming languages 1.2 Describe a typical java development environment 1.3 Describe java's role in developing distributed client/server applications for the internet and web 1.4 Define object-oriented design with the UML and design patterns.
2. Introduce simple examples to illustrate Java features. Define Java applications and primitive data types.	2.1 Describe how to write simple java applications 2.2 Demonstrate how to use input and output statements 2.3 Familiarise with primitive types 2.4 Describe basic memory concepts 2.5 Describe how to use arithmetic operators 2.6 Describe arithmetic-operator precedence 2.7 Demonstrate how to write decision-making statements 2.8 Describe how to use relational and equality operators.
3. Define Java Development kit. Discuss Java Applets. Define how to draw strings and lines.	3.1 Differentiate between applets and applications 3.2 Illustrate some of java's exciting capabilities through the Java software development kit's demonstration applets 3.3 Demonstrate how to write simple java applets 3.4 Illustrate how to write a simple hypertext markup language (html) document to load an applet into the **appletviewer** or a web browser and execute the applet 3.5 Describe the difference between variables and references.
4. Describe the structured-programming principle. Define how control structures help	4.1 Describe basic problem-solving techniques

build and manipulate objects.	4.2	Describe how to develop algorithms through the process of top-down, stepwise refinement
	4.3	Describe how to use the **if** and **if**…**else** selection statements to choose among alternative actions
	4.4	Illustrate how to use the **while** repetition statement to execute statements in a program repeatedly
	4.5	Describe counter-controlled repetition and sentinel-controlled repetition
	4.6	Illustrate how to use the assignment, increment and decrement operators.
5. Describe counter-controlled repetition. Define logical operators.	5.1	Describe how to use the **for** and **do**…**while** repetition statements to execute statements in a program repeatedly
	5.2	Illustrate multiple selection using the **switch** selection statement
	5.3	Describe how to use the **break** and **continue** program control statements
	5.4	Describe how to use the logical operators.
6. Define methods in Java. Describe method declarations and method overloading.	6.1	Describe how to construct programs modularly from small pieces called *methods*
	6.2	Define the common math methods available in the java API
	6.3	Describe how to create new methods understand the mechanisms for passing information between methods
	6.4	Describe simulation techniques that use random-number generation
	6.5	Illustrate how the visibility of declarations is limited to specific regions of programs
	6.6	Describe how to write and use methods that call themselves.
7. Differentiate arrays and variables. Describe the process of declaring, creating and referencing arrays.	7.1	Define the array data structure
	7.2	Describe the use of arrays to store, sort and search lists and tables of values
	7.3	Define how to declare an array, initialise an array and refer to individual elements of an array
	7.4	Describe how to pass arrays to methods
	7.5	Illustrate how to declare and manipulate multidimensional arrays.
8. Define Object Oriented Programming (OOP). Describe data encapsulation and methods. Discuss how to create and use objects.	8.1	Describe encapsulation and data hiding;
	8.2	Describe the notions of data abstraction and abstract data types (ADTs);
	8.3	Describe how to create java ADTs—namely, classes
	8.4	Describe how to create and use objects
	8.5	Define how to control access to instance variables and methods
	8.6	Define the use of the **this** reference

	8.7 Describe how to use class variables and methods
9. Define inheritance. Define superclass and subclasses. Identify the relationship between them.	9.1 Describe how inheritance promotes software reusability 9.2 Define the notions of superclasses and subclasses 9.3 Define access modifier **protected** 9.4 Illustrate how to access superclass members with **super** 9.5 Describe the use of constructors and finalisers in inheritance hierarchies 9.6 Demonstrates the mechanics of inheritance.
10. Define polymorphism. Understand relationships among objects in an inheritance hierarchy.	10.1 Describe the concept of polymorphism 10.2 Illustrate how to use overridden methods to effect polymorphism 10.3 Distinguish between abstract and concrete classes 10.4 Identify how to declare **abstract** methods to create abstract classes 10.5 Define how polymorphism makes systems extensible and maintainable 10.6 Determine an object's type at execution time.
11. Define character and string classes in Java.	11.1 Demonstrate how to create and manipulate nonmodifiable character string objects of class **string** 11.2 Demonstrate how to create and manipulate modifiable character string objects of class **stringbuffer** 11.3 Illustrate how to create and manipulate objects of class **character** 11.4 Illustrate how to use a **stringtokenizer** object to break a **string** object into tokens.
12. Define Java files and streams. Illustrate how to create, read and update sequential-access files.	12.1 Describe how to create, read, write and update files 12.2 Be able to use class **file** 12.3 Describe the java streams class hierarchy 12.4 Be able to use the **fileinputstream** and **fileoutputstream** classes 12.5 Be able to use a **jfilechooser** dialog to access files and directories 12.6 Be able to use the **objectinputstream** and **objectoutputstream** classes 12.7 Be able to use class **randomaccessfile** 12.8 Familiarise with sequential-access and random-access file processing.

Recommended Learning Resources:
JDeveloper

Text Books	• Effective Java: A Programming Language Guide by Joshua Bloch. ISBN-10: 0321356683 • Java: How to Program by Harvey & PaulDeitel & Deitel. ISBN-10: 0132222205
Study Manuals 	BCE produced study packs
CD ROM 	Power-point slides
Software 	Java Programming Language