LONDON CAPITAL COMPUTER COLLEGE

Advanced Diploma in Programming (602) – Advanced Java Programming

Prerequisites: Programming experience in C for at least six months.	**Corequisites:** A pass or higher in Diploma in Programming or equivalence.
Aim: This course exposes the candidates to advanced Java features such as JavaBeans, Servlet Programming, the Java database connectivity, remote method invocation, and swing. Other major topics in this course include network programming serialization, properties, security, the collection classes and architectures. At the end of the course, candidates will be able to: distinguish between the various phases of construction of objects; choose the correct data structures from the Java collections classes, extend appropriate I/O classes in order to create a new I/O class, distinguish among various thread problems and provide the correct programming solution; create a Java Bean component and execute it within the BeanBox, write TCP/IP client server applications using the sockets, execute methods on a remote object and use the results that are returned from the method, write Java Servlets to implement HTML form processing; write Java applications using the JDBC to make database independent queries; and use many of the superior capabilities of the swing components.	
Required Materials: Recommended Learning Resources.	**Supplementary Materials:** Lecture notes and tutor extra reading recommendations.
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. Discuss Java's graphic capabilities.	1.1 Describe graphics contexts and graphics objects 1.2 Describe and be able to manipulate colors 1.3 Describe and be able to manipulate fonts 1.4 Define how to use graphics methods to draw lines, rectangles, rectangles with rounded corners, three-dimensional rectangles, ovals, arcs and polygons 1.5 Define how to use methods of class graphics2d from the java2d api to draw lines, rectangles, rectangles with rounded corners, ellipses, arcs and general paths 1.6 Analyse how to specify paint and stroke characteristics of shapes displayed with graphics2d.
2. Define Graphical User Interface (GUI). Describe how users interact with GUI components via the mouse and keyboard.	2.1 Describe the design principles of Graphical User Interfaces (GUI) 2.2 Demonstrate how to build graphical user interfaces 2.3 Explain the packages containing GUI-related components, event-handling classes and interfaces 2.4 Describe how to create and manipulate buttons, labels, lists, text fields and panels 2.5 Describe mouse events and keyboard events 2.6 Define how to use layout managers.

3. Discuss advanced GUI components, including text areas, sliders and menus.	3.1 Describe how to create and manipulate text areas, sliders, menus, popup menus and windows 3.2 Describe how to create customised jpanel objects 3.3 Demonstrate how to change the look-and-feel of a GUI, using swing's pluggable look-and-feel (plaf) 3.4 Describe how to create a multiple-document interface with jdesktoppane and jinternalframe 3.5 Describe how to use additional layout managers.
4. Describe exception handling. Define the uses of exception handling.	4.1 Define exception and error handling 4.2 Describe how to use try, throw and catch to detect, indicate and handle exceptions, respectively 4.3 Describe how to use the finally clause to release resources 4.4 Define the java exception hierarchy 4.5 Describe how to declare new exception classes 4.6 Describe how to create chained exceptions.
5. Define multi-threading and the thread states.	5.1 Describe multithreaded programming 5.2 Demonstrate how multithreading can improve program performance 5.3 Describe the life cycle of a thread 5.4 Define thread priorities and scheduling 5.5 Describe how to create, manage and destroy threads 5.6 Describe thread synchronization 5.7 Describe daemon threads 5.8 Demonstrate how to stop and suspend threads
6. Define networking. Describe communication over the internet and how to read a file on a Web server.	6.1 Outline java networking with URLs, sockets and datagrams 6.2 Describe how to implement java networking applications by using sockets and datagrams 6.3 Describe how to implement java clients and servers that communicate with one another 6.4 Identify how to implement network-based collaborative applications 6.5 Define how to construct a multithreaded server.
7. Describe multimedia images, animation and audio. Describe how to create image maps and play audio files.	7.1 Describe how to get and display images 7.2 Demonstrate how to create animations from sequences of images 7.3 Identify how to create image maps 7.4 Describe how to get, play, loop and stop sounds, using an AudioClip.
8. Define dynamic data structures. Describe the operations of linked lists, stacks, queues and binary trees.	8.1 Describe how to form linked data structures using references, self-referential classes and recursion 8.2 Identify how to create and manipulate

		dynamic data structures, such as linked lists, queues, stacks and binary trees
	8.3	Describe various important applications of linked data structures
	8.4	Describe how to create reusable data structures with classes, inheritance and composition.
9. Define Java utilities package.	9.1	Describe containers, such as classes Vector and Stack, and the Enumeration interface
	9.2	Describe how to use Hashtable objects
	9.3	Define how to use persistent hash tables manipulated with objects of class Properties
	9.4	Describe how to use bit manipulation to process the individual bits in integer data
	9.5	Demonstrate how to use BitSet objects
10. Define Java collections framework.	10.1	Describe what collections are
	10.2	Demonstrate how to use class arrays for common array manipulations
	10.3	Describe how to use the collections-framework implementations
	10.4	Demonstrate how to use collections-framework algorithms to manipulate various collections
	10.5	Describe how to use the collections-framework interfaces to program polymorphically
	10.6	Identify how to use iterators to "walk" through the elements of a collection
	10.7	Describesynchronization wrappers and modifiability wrappers.
11. Identify how to connect to a database.	11.1	Describe relational databases
	11.2	Define basic database queries using sql
	11.3	Demonstrate how to use the classes and interfaces of package java.sql to manipulate databases.
	11.4	Define database management system and structured query language.
	11.5	Describe Java database connectivity.
12. Define Java servlets. Describe networking capabilities.	12.1	Describe how to execute servlets with the apache tomcat server
	12.2	Identify how to respond to HTTP requests from an httpservlet
	12.3	Describe how to redirect requests to static and dynamic web resources.
	12.4	Analyse to create and deploy javaserver pages
	12.5	Describe how use JSP's implicit objects and scriptlets to create dynamic web pages
	12.6	Define how to specify global JSP information with directives
	12.7	Describe how to use actions to manipulate javabeans in a JSP, to include resources dynamically and to forward requests to other JSPs.

13. Define JavaServer Pages. Describe the JavaServer Pages key components.	13.1 Describe how to create and deploy javaserver pages. 13.2 Demonstrate how to use JSP's implicit objects and scriptlets to create dynamic web pages. 13.3 Specify global JSP information with directives. 13.4 Use actions to manipulate javabeans in a jsp, to include resources dynamically and to forward requests to other JSPs.

Recommended Learning Resources: Advanced Java Programming

Text Books	• Java Programming: Advanced Topics by Joe Wigglesworth, Paula McMillan and T. Wigglesworth. ISBN-10: 0619159685 • Advanced Java: How to Program by Harvey M. Deitel, Paul J. Deitel and Sean E. Santry. ISBN-10: 0130895601 • 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