Institiúid Teicneolaíochta, Trá Lí
INSTITUTE OF TECHNOLOGY TRALEE

SUMMER EXAMINATIONS AY 2014 - 2015

Object Oriented Programming 3

Module Code: PROG61005
CRN 43852

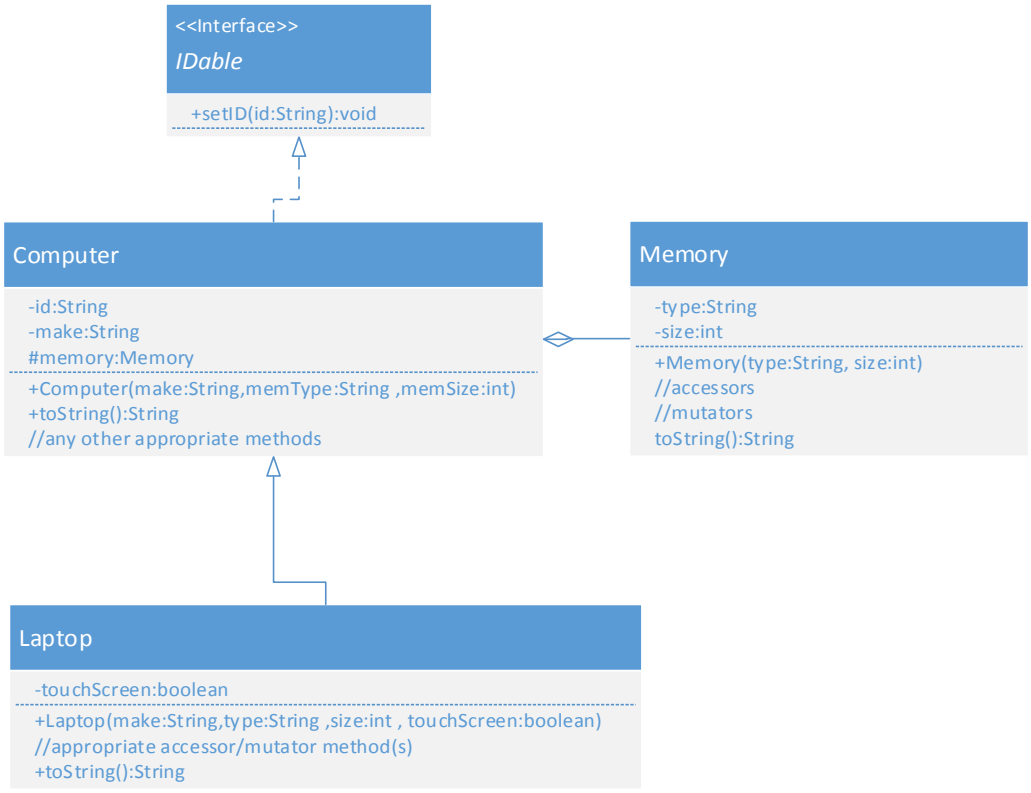
Internal Examiner: Mr. John Walsh
External Examiner: Mr Michael Godley

Duration of Exam: 2 Hours

Instructions to Candidates: **Answer Question 1 and one other question. Each question is 50 marks**

Question 1

In the UML diagram below, IDable is an Interface. Computer, Memory and Laptop are concrete instantiable classes.



a) Explain the differences between abstract and concrete classes. (4 marks)

b) Write the full code for the *IDable* interface. (4 marks)

c) Write the full code, including the class header, attributes and all appropriate methods for the Computer class. (15 marks)

d) Write the full code, including the class header, attributes, and all appropriate methods for the Laptop class. (12 marks)

e) Write an instance method called makeLaptop that will create and return a single Laptop object. The method should prompt the user for the appropriate values. You can assume that the JOptionPane class is available. (7 mark)

f) An application that uses the Computer class is required to create a sorted collection of Computer objects. The collection is **sorted** according to the **id** of the Computer objects. Explain what changes you would make to the Computer class to allow this and write the appropriate code. See the Appendix for helpful information for this part. (8 marks)

Question 2

a) Give three similarities and three differences between an ArrayList and a LinkedList. (6 marks)

b) Give a brief explanation of generic programming as used in the Java Collection Framework indicating two advantages of its use. You can include an example as part of your explanation. (6 marks)

c) Considering the UML diagram in Question 1, write a driver class that creates a collection of Computer objects. No duplicates are allowed in the collection. You can assume that two methods exist called **makeComputer** and **makeLaptop** that create and return a single object of the Computer and Laptop class respectively. Use an appropriate structure to allow for multiple Computers and Laptops to be added to the collection depending on the user's choice. See the Appendix for helpful information for this part. (13marks)

d) A method is required that takes the collection created in part c) as a parameter. Using an enhanced for loop, the method creates a filtered collection that

contains all computers or laptops with a memory size greater than 2 MB. In addition the method uses an iterator to print full details of the filtered collection **sorted** according to their **id** value. You can assume that appropriate code exists that compares generic Computer objects. Write the code for the method. (13 marks)

e) Explain the concept of polymorphism and dynamic method binding. You can refer to the code written for part d) as part of your explanation. (6 marks)

f) Write a brief note on scope. You should include an explanation of the different modifiers available in java. (6 marks)

Question 3

a) The HTTP command for requesting a resource from a network server is given by: **GET *resource* HTTP/1.1**
Host: *hostname*
blank line

Write the code to create a String that represents the command to retrieve the resource from the URL: **http://ittralee.ie/en/** (8 marks)

b) An algorithm to retrieve a web page from a server is given by:
Create a socket
Obtain its low level input and output stream
Turn streams into high level I/O streams
Send the request command (as written for part a)
Flush the buffer
Read the server response
Close the socket

Write the code to implement a network client program using this algorithm. See the Appendix for helpful information for this part. (19 marks)

c) Outline briefly two methods that could be used to handle any IO exceptions generated by the code written for part b) above. (6 marks)

d) Indicate what code changes you would make to the code written in part b) to handle any IO exceptions for both methods explained in part c). (7 marks)

e) The Java API provides support for HTTP via the URL and the URLconnection classes. Explain the benefit of using the URLconnection class and indicate the code changes you would make to part b) if you were to use it. See the Appendix for helpful information for this part. (10 marks)

Appendix :

Interface Collection<E>

Method Summary

Methods

Modifier and Type	Method and Description
boolean	add(E e) Ensures that this collection contains the specified element (optional operation).
boolean	addAll(Collection<? extends E> c) Adds all of the elements in the specified collection to this collection (optional operation).
void	clear() Removes all of the elements from this collection (optional operation).
boolean	contains(Object o) Returns true if this collection contains the specified element.
boolean	containsAll(Collection<?> c) Returns true if this collection contains all of the elements in the specified collection.
boolean	equals(Object o) Compares the specified object with this collection for equality.
int	hashCode() Returns the hash code value for this collection.
boolean	isEmpty() Returns true if this collection contains no elements.
Iterator<E>	iterator() Returns an iterator over the elements in this collection.
boolean	remove(Object o) Removes a single instance of the specified element from this collection, if it is present (optional operation).
boolean	removeAll(Collection<?> c) Removes all of this collection's elements that are also contained in the specified collection (optional operation).
boolean	retainAll(Collection<?> c) Retains only the elements in this collection that are contained in the specified collection (optional operation).
int	size() Returns the number of elements in this collection.

Interface Comparable<T>

Method Summary

Methods

Modifier and Type	Method and Description
int	compareTo(T o) Compares this object with the specified object for order.

Interface Comparator<T>

Method Summary

Methods

Modifier and Type	Method and Description
`int`	`compare(T o1, T o2)` Compares its two arguments for order.
`boolean`	`equals(Object obj)` Indicates whether some other object is "equal to" this comparator.

Class Socket

Constructor Summary

Constructors

Modifier	Constructor and Description
	`Socket()` Creates an unconnected socket, with the system-default type of SocketImpl.
`protected`	`Socket(SocketImpl impl)` Creates an unconnected Socket with a user-specified SocketImpl.
	`Socket(String host, int port)` Creates a stream socket and connects it to the specified port number on the named host.

Method Summary

Methods

Modifier and Type	Method and Description
`void`	`bind(SocketAddress bindpoint)` Binds the socket to a local address.
`void`	`close()` Closes this socket.
`void`	`connect(SocketAddress endpoint)` Connects this socket to the server.
`void`	`connect(SocketAddress endpoint, int timeout)` Connects this socket to the server with a specified timeout value.
`SocketChannel`	`getChannel()` Returns the unique `SocketChannel` object associated with this socket, if any.
`InetAddress`	`getInetAddress()` Returns the address to which the socket is connected.
`InputStream`	`getInputStream()` Returns an input stream for this socket.
`OutputStream`	`getOutputStream()` Returns an output stream for this socket.

Class Scanner

Constructor Summary

Constructors

Constructor and Description
`Scanner(File source)` Constructs a new `Scanner` that produces values scanned from the specified file.
`Scanner(File source, String charsetName)` Constructs a new `Scanner` that produces values scanned from the specified file.
`Scanner(InputStream source)` Constructs a new `Scanner` that produces values scanned from the specified input stream.

Method Summary

Methods

Modifier and Type	Method and Description
`void`	`close()` Closes this scanner.
`boolean`	`hasNext()` Returns true if this scanner has another token in its input.
`boolean`	`hasNextLine()` Returns true if there is another line in the input of this scanner.
`String`	`nextLine()` Advances this scanner past the current line and returns the input that was skipped.
`long`	`nextLong()` Scans the next token of the input as a `long`.

Class PrintWriter

Constructor Summary

Constructors

Constructor and Description
`PrintWriter(File file)` Creates a new PrintWriter, without automatic line flushing, with the specified file.
`PrintWriter(File file, String csn)` Creates a new PrintWriter, without automatic line flushing, with the specified file and charset.
`PrintWriter(OutputStream out)` Creates a new PrintWriter, without automatic line flushing, from an existing OutputStream.
`PrintWriter(OutputStream out, boolean autoFlush)` Creates a new PrintWriter from an existing OutputStream.

Method Summary

Methods

Modifier and Type	Method and Description
`PrintWriter`	`append(char c)` Appends the specified character to this writer.
`void`	`flush()` Flushes the stream.
`void`	`print(String s)` Prints a string.

Class URL

Constructor Summary

Constructors

Constructor and Description
`URL(String spec)` Creates a `URL` object from the `String` representation.

Method Summary

Methods

Modifier and Type	Method and Description
`URLConnection`	`openConnection()` Returns a `URLConnection` instance that represents a connection to the remote object referred to by the `URL`.

Class URLConnection

Method Summary

Methods

Modifier and Type	Method and Description
`InputStream`	`getInputStream()` Returns an input stream that reads from this open connection.
`long`	`getLastModified()` Returns the value of the `last-modified` header field.
`OutputStream`	`getOutputStream()` Returns an output stream that writes to this connection.

Interface Iterator<E>

Method Summary

Methods

Modifier and Type	Method and Description
`boolean`	`hasNext()` Returns `true` if the iteration has more elements.
`E`	`next()` Returns the next element in the iteration.
`void`	`remove()` Removes from the underlying collection the last element returned by this iterator (optional operation).

Class JOptionPane

Field Summary

Fields

Modifier and Type	Field and Description
`static int`	`CANCEL_OPTION` Return value from class method if CANCEL is chosen.
`static int`	`YES_NO_OPTION` Type used for `showConfirmDialog`.
`static int`	`YES_OPTION` Return value from class method if YES is chosen.

Method Summary

Methods

Modifier and Type	Method and Description
`static int`	`showConfirmDialog(Component parentComponent, Object message)` Brings up a dialog with the options *Yes*, *No* and *Cancel*; with the title, **Select an Option**.