Written Exam

**EHSDI Unit: EH102 (Basic Java)**

**Time: 2 hours**

**Student Name:**

Take time to read the questions carefully and write your answers clearly on this paper.

**1 (a). Describe the differences between a long and a float.**

***2 marks***

**1 (b). Explain what is wrong with the following code and how you could fix it:**

***2 marks***

**int** n = 20;

**short** s = n;

# 1 (c). What is the difference between an *explicit* cast and an *implicit* cast?

***2 marks***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 2 uses the following classes:**

**class** A {

**public** A(String msg) { **this**.msg = msg; }

**private** String msg;

}

**class** B **extends** A {

**public** B() {}

**public void** printMessage() { System.out.println(msg); }

}

**2 (a). What is the problem the method printMessage and how could you fix the classes above so that it works?**

***2 marks***

**2 (b). What is the problem with the constructor of class B and how can it be fixed with the super keyword?**

***2 marks***

**2 (c). What is the problem with the following statement?**

***2 marks***

B b = **new** A(“Hello”);

**3. Explain what you understand by *encapsulation*, and why it is an essential part of good software design.**

***3 marks***

**4. List 5 problems you see with the following code. *Note: this code will compile and run correctly.***

***5 marks***

**public class** aMessageClass {

**private String** Message = “Hello”;

//

// Prints a message!

**public void** printmessage(){

System.out.println(Message);

}

}

1.                                                                                                                                                2.                                                                                                                                                3.                                                                                                                                                4.                                                                                                                                                5.

**5. Explain the difference between *class members* (i.e. static) and *instance members*.**

***3 marks***

**6. What is the difference between Integer and int? Why does the following code work:**

***2 marks***

**int** x = 4;  
Integer n = x;

**7 (a). Modify the following definition of a generic class called PolarPoint by adding a *getter* and a *setter* for the variable angle.**

***3 marks***

**class** PolarPoint<R **extends** Number, T> {

**protected** R radius;

**protected** T angle;

}

**7 (b). How would you create an instance of PolarPoint that uses Double for the radius and Float for the angle?**

***2 marks***

**7 (c). What is wrong with the following instantiation of PolarPoint?**

***2 marks***

PolarPoint<String, **int**> p = **new** PolarPoint<String, **int**>();

**Question 8 uses the following classes:**

**abstract** **class** A {

**public** **abstract** **void** foo();

}

**class** B **extends** A {

}

**public class** C {

**public** C() {

A a = **new** B();

}

}

**8 (a). Explain why the code above won’t compile.**

***2 marks***

**8 (b). Rewrite classes A and B so that A is an interface instead of an abstract class, and so that the code will compile.**

***3 marks***

**Question 9 uses the following classes:**

**public class** Divider {

**public double** divide(**double** x, **double** y)

**return** x / y;

}

**public double** reciprocal(**double** x) {

**return** divide(1, x);

}

}

**public class** DivideByZeroException **extends** Exception {

}

**9 (a). Modify the divide method below so that if y is zero, it will throw an DivideByZeroException.**

***3 marks***

**public double** divide(**double** x, **double** y)

{

**return** x / y;

}

**9 (b). Rewrite the reciprocal method so that it will catch the exception from the call to divide and handle it by printing an error message to the console.**

***2 marks***

**public double** reciprocal(**double** x) {

}

**Question 10 uses the following class:**

**public class** Provider {

**protected int** id;

**protected** String name;

**protected** String licenceNumber;

}

**10 (a). If two instances of Provider have the same id and name values, then they are to be considered equal. Write a suitable equals method below.**

***2 marks***

**public boolean** equals(Object obj) {

}

**10 (b). Instances of the Provider class are to be stored in a TreeSet, sorted by id (ascending). Write a suitable compareTo method below.**

***2 marks***

**public int** compareTo(Provider p) {

}

**10 (c). Write a suitable hashCode method below.**

***2 marks***

**public int** hashCode() {

}

**THE END OF THE TEST**