THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

**UNIVERSITY OF LONDON** **CO1109 ZB**

**BSc, CertHE and Diploma Examination**

**COMPUTING AND INFORMATION SYSTEMS and CREATIVE COMPUTING**

**Introduction to Java and Object-Oriented Programming**

Thursday 3 May 2018: 10.00 – 13.00

Time allowed: 3 hours

There are **SIX** questions on this paper. Candidates should answer **FOUR** questions. Full marks will be awarded for complete answers to **FOUR** questions. Each question is worth 25 marks. The mark for each part of a question is indicated in [ ] brackets.

Only your first **FOUR** answers, in the order that they appear in your answer book, will be marked.

There are 100 marks available on this paper.

No calculators should be used.

**Question 1**

(a)

(i) Say what kind of loop (if any) it would be most appropriate to use when implementing the following: [4 marks]

(A) Asking the user to enter a valid file name.

(B) Displaying the contents of an `Array`.

(C) Printing the first entry in an `Array`.

(D) Printing the following shape:

```
*
**
***
****
*****
```

(ii) Consider the following *for* loop:

```
for(int i=0;i < 3;i=i+1) System.out.println("*");
```

What is the first part of the *for* loop, `int i=0;`, called? [2 marks]

(A) The initialisation expression.

(B) The final expression.

(C) The guard.

(D) None of the above.

(iii) Consider the class *Empty:*

```
class Empty{
     static int x;
     public static void main(String[] args){
          for(;;) x = 1;
     }
}
```

What will be output when the class is run? [2 marks]

(A) Infinite loop – the program will hang with no output.

(B) No output because the program will not compile.

(C) Run time error.

(D) None of the above.

(b) For each of the following loops, say how many asterisks (*) will be printed. If you think that the loop will continue indefinitely without stopping, write 'infinite loop' in your answer book for that part of the question. [9 marks]

(i)
```
for(int i=0;i<10;i++) System.out.print("*");
```

(ii)
```
int j=21; while (j>6){System.out.print("*");j=j-2;}
```

(iii)
```
for(int i=-2;i<6;i++) System.out.print("*");
```

(iv)
```
int k=3; while (k>3){System.out.print("*");k=k+3;}
```

(v)
```
for(int i=0;i>=0;i++) System.out.print("*");
```

(vi)
```
for(int i=0;i<11;i=i+2) System.out.print("*");
```

(vii)
```
int m=0; while(m<=6){System.out.print("*");m=m+3;}
```

(viii)
```
for(int i=6; i>=6; i++) System.out.print("*");
```

(ix)
```
int p=0; while (true){System.out.print("*");}
```

(c) Consider the *Verifier* class below:

```
import java.util.Scanner;

public class Verifier {
    private static String[] customers =
          new String[]{"Bob", "Rob", "Jing", "Ahmed"};
    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
          verify();
    }

    private static void verify() {
          String customer = getCustomerFromUser();
          if(isValidCustomer(customer))
               System.out.println("Verified customer " + customer);
          else
               System.out.println("Invalid customer " + customer);
    }

    private static String getCustomerFromUser() {
        return getUserInput("Enter customer name: ");
    }
}
```

The class will not compile because it is missing two methods. Write the two missing methods in the *Verifier* class. [8 marks]

NOTE that a customer is a valid customer if their name appears anywhere in the *customers* `Array`, exactly as input by the user.

Below are two examples of output with the methods written correctly:

```
Enter customer name: Ahmed
Verified customer Ahmed

Enter customer name: AHMED
Invalid customer AHMED
```

**Question 2**

(a)

(i) If your program does not compile the best thing to do is: [2 marks]

(A) Correct the final error only and recompile.

(B) Correct the first error only and recompile.

(C) Correct all errors and recompile.

(D) None of the above.

(ii)

```
Class Xda{

     public static void main(String[] Args){
          ;;;;;
     }
}
```

What single error will the compiler find in the *Xda* class above? [2 marks]

(A) *Args* should be *args.*

(B) *Class* should be *class.*

(C) Missing statements in the main method.

(D) None of the above.

(iii) What error will the compiler find in the following program: [2 marks]

```
class Q2{
     static int x;
     static int y;
     static int n = 6;

     public static void main(String[] args){
          if (n<0)
               x=1;
               y=2;
          else{
               x=2;
               y=1;
          }
     }
}
```

(A) `'else' without 'if'`

(B) `reached end of program while parsing`

(C) `unreachable statement`

(D) None of the above.

(iv) What error will the compiler find in the following program: [2 marks]

```
public class D{

     static subtract(int x, int y){
          int temp = x - y;
          return temp;
     }

     public static void main(String[] args){
          int k = subtract(5, 19);
     }
}
```

(A) `variable k should be declared static`

(B) `a static method cannot be referenced in a non-static context`

(C) `return type required`

(D) None of the above.

(b) For each of the following expressions state whether it type checks correctly or not. Write 'yes' in your answer book if it type checks correctly, and write 'no' if it does not. [9 marks]

(i) `Math.abs("Excalibur");`

(ii) `Integer.parseInt(15);`

(iii) `int x = Math.abs("Excalibur".length());`

(iv) `Integer.parseInt("401");`

(v) `"fourZeroOne".compareTo("401");`

(vi) `int y = "fourZeroOne".compareTo("401");`

(vii) `"boy".replace('b',"soup".charAt(0));`

(viii) `Math.max(("hello" + " dog").length(),5);`

(ix) `("boy".replace('b',"soup".charAt(0))).length();`

(c) Write a method to convert a `String` to a capitalised `String`. Your method should take a single `String` parameter, and should return the `String` given to it after making the first `char` a capital (if a member of the alphabet), and the rest of the `chars` (if any) lower case. [8 marks]

For example:

| Parameter | Returns |
|---|---|
| HELLO DOLLY | Hello dolly |
| Hello there | Hello there |
| hello again | Hello again |
| b | B |
| 352 | 352 |

**You may find the following 4 methods from the Java String API helpful:**

## substring

`public String substring(int beginIndex)`

Returns a new string that is a substring of this string. The substring begins with the character at the specified index and extends to the end of this string.

**Examples:**

```
"unhappy".substring(2) returns "happy"
"Harbison".substring(3) returns "bison"
"emptiness".substring(9) returns "" (an empty string)
```

**Parameters:**

`beginIndex` - the beginning index, inclusive.

**Returns**:
the specified substring.

/**********/

## substring

```
public String substring(int beginIndex, int endIndex)
```

Returns a new string that is a substring of this string. The substring begins at the specified beginIndex and extends to the character at index endIndex - 1. Thus the length of the substring is `endIndex-beginIndex`.

**Examples:**

```
"hamburger".substring(4, 8) returns "urge"
"smiles".substring(1, 5) returns "mile"
```

**Parameters:**
`beginIndex` - the beginning index, inclusive.
`endIndex` - the ending index, exclusive.

**Returns:**
the specified substring.

/**********/

## toLowerCase

```
public String toLowerCase()
```

Converts all of the characters in this `String` to lower case using the rules of the default locale.

**Returns:**
the `String`, converted to lowercase.

/**********/

## toUpperCase

```
public String toUpperCase()
```

Converts all of the characters in this `String` to upper case using the rules of the default locale.

**Returns**:
the `String`, converted to uppercase.

**Question 3**

(a)

(i) Explain why the following program will not compile. [2 marks]

```
public class BoolNo{
     public static void main(String[] args){
          if 3/1.2 < 0 System.out.print("error");
     }
}
```

(ii) What is the output of the following? [2 marks]

```
public class Bool3{
     public static void main(String[] args){
          if (false) System.out.print("hello");
          else System.out.print("goodbye");
     }
}
```

(A) `hello`

(B) `goodbye`

(C) No output – the program will not compile.

(D) None of the above.

(iii) What is the output of the following? [2 marks]

```
class Bool5{
     public static void main(String[] args){
          System.out.println(!(!(true)));
     }
}
```

(A) `true`

(B) `false`

(C) No output – the program will not compile.

(D) None of the above.

(iv) Consider class *Bool4:*

```
import java.util.Scanner;

public class Bool4{

     public static void main(String[] args){
          Scanner in =new Scanner(System.in);
          System.out.print("Enter Number>");
          int t=in.nextInt();
          if (t==0 && t==3)
     System.out.println("hello");
     }
}
```

Which of the following describes the behaviour of *Bool4* when the user enters 3? [2 marks]

(A) The program will output `hello`

(B) The program will output nothing at all.

(C) The program will output `0`

(D) The program will stop with a run-time error.

(E) None of the above.

(b)

(i) Give the output of the *SopCon* class: [3 marks]

```
class SopCon{

     public static void main(String[] args){
          System.out.println("3+3");
          System.out.println(3+3);
          System.out.println("3"+"3");
     }
}
```

(ii) Give the output of the *Division* class: [3 marks]

```
class Division{
     public static void main(String[] args){
          System.out.println(5/2);
          System.out.println(5.0/2.0);
          System.out.println(5/2.0);
     }
}
```

(iii) Say which two of the three numbered statements in the *Precedence* class below, will have the same output: [3 marks]

```
class Precedence{
     public static void main(String[] args){
     /*1*/      System.out.println(3*3+1);
     /*2*/      System.out.println((3*3)+1);
     /*3*/      System.out.println(3*(3+1));
     }
}
```

(c) Write a class that does the following: [8 marks]

- First: it outputs to the user:
  `Human beings glow in the dark.`
  `Enter 'true' or 'false' to answer:`
- Second: it reads in the user's answer
- Third: it outputs to the user:
  `A piece of paper can only be folded in half 8 times.`
  `Enter 'true' or 'false' to answer:`
- Fourth: it reads in the user's answer
- Finally it outputs to the user **one** of the following:
  "You are so right!" when the user has entered "true" or "TRUE" or "True" or "TruE" or anything else that can be parsed to the boolean value *true,* in answer to both questions.
  "You are half right" when the user has answered *true* to one of the questions, and *false* (or something that cannot be parsed to *true*) to the other.
  "Wrong answers!" (when the user's answer to both questions cannot be parsed to *true*, *i.e.* the user is assumed to have answered *false* to both questions).

You may find the following extract from the Java `Boolean` API helpful:

## parseBoolean

```
public static boolean parseBoolean(String s)
```

Parses the string argument as a boolean. The `boolean` returned represents the value `true` if the string argument is not `null` and is equal, ignoring case, to the string "true".

**Example:** `Boolean.parseBoolean("True")` returns `true`.
**Example:** `Boolean.parseBoolean("yes")` returns `false`.

**Parameters:**
`s` - the `String` containing the boolean representation to be parsed.

**Returns:**
the boolean represented by the string argument.

**Question 4**

(a) Consider the following class:

```
import java.io.*;
public class filey{

  public static void bling(String s) throws Exception{
     BufferedReader in =new BufferedReader(new FileReader(s));
     int t=in.read();
     while (t!=-1){
          System.out.print((char)t);
          t=in.read();
     }
  }

  public static void main(String[] args) throws Exception{
     bling(args[0]);
  }
}
```

(i) When does the variable *t* get the value -1? [2 marks]

(A) When the program starts.

(B) When the *bling()* method reads in the end of file character.

(C) When the file closes.

(D) None of the above.

(ii) If the *args[0]* variable contains the `String "filey.java"` what will the program do? [2 marks]

(A) The program will output nothing.

(B) The program will stop with a run-time error.

(C) The program will output `0`

(D) The program will output the text contained in the text file 'filey.java'.

(E) None of the above.

(iii) What would happen if we ran *filey.java* with an empty input file? [2 marks]

(A) The program would hang waiting for input.

(B) There would be no output as the condition `t!=-1` would be false at the beginning.

(C) The program would end with an `InputMismatchException`

(D) None of the above.

(iv) What is the name of the exception that would be thrown if *filey.java* was run with the input parameter having the name of a file that did not exist? [2 marks]

(A) `InputMismatchException`

(B) `NumberFormatException`

(C) `FileNotFoundException`

(D) `EOFException`

(E) None of the above.

(b)

(i) What is the output of program *Yez*, below? [2 marks]

```
public class Yez{

     public static void main(String[] args){
          double x = 5.5; double y = 2.0;
          try{
               double z = x/y;
               System.out.println("division
done");
          }
          catch(Exception e){
               System.out.println("not a
number");
          }
     }
}
```

(ii) What is the output of program *Ez*, below? [2 marks]

```
public class Ez{
     public static void main(String[] args){
          try{
               Integer.parseInt("delta");
               System.out.println("epsilon");
          }
          catch(Exception e){
               System.out.println("zeta");
          }
     }
}
```

(iii) Consider the class *ArrayQ*, below:

```
class ArrayQ{
     public static void main(String[] args){
          int k=3;
          int[] num= new int[k];
          for(int i=0;i<k;i++)num[i]=i+1;
          for(int i=0;i<k;i++)System.out.println(num[i]);
     }
}
```

Which one of the following statements about the *ArrayQ* class is correct? [2 marks]

(A) The program will not compile.

(B) The program will compile but when it is run it will crash with an 'array out of bounds' exception.

(C) The program will compile and run but output nothing.

(D) The program will compile and run and output:

```
0
1
2
3
```

(E) None of the above.

(iv) Name the exception that will be thrown by the following: [3 marks]

```
class T1{
     public static void main(String[] args){
          int x=Integer.parseInt("dl;fkas;lkf");
     }
}
```

(c) The file *file1.txt* has the following contents:

```
5
6
15
7
0
1
10
```

Write a method with the heading [8 marks]
`public static void multiplyFileContents()`
that does the following:

- Opens *file1.txt* to read from.
- Opens a file called *file2.txt* to write to.
- Reads each number from *file1.txt*, multiplies it by 10, and writes it to *file2.txt*.
- Writes numbers to *file2.txt* one to a line.
- Closes both files.
- Handles any potential exceptions with try/catch.
- If an exception occurs the catch block prints a message about it.

**Question 5**

(a)

(i) Java has simple (primitive) variables and reference variables. Say which of the following are reference variables: [2 marks]

```
Boolean
char
String
double
```

(ii) Say whether the following two statements are TRUE or FALSE: [2 marks]

(A) When you make an assignment to a simple (primitive) variable, the variable does not hold the value assigned, but instead points to it in memory.

(B) Java allows two bytes (=16 bits) to store characters. This means that in Java we can represent $2^{16}$ different characters.

(iii) Say what the output of the *IntParam* class will be: [2 marks]

```
class IntParam{
     public static void main(String [ ] args){
          int n=3;
          p(n);
          System.out.println(n);
     }

     static void p(int m){
          m=m+6;
     }
}
```

(iv) Say what the output of the *ArrayParam* class will be: [2 marks]

```
class ArrayParam{
     public static void main(String [ ] args){
          int[]a=new int[1];
          a[0]=1;
          p(a);
          System.out.println(a[0]);
     }

     static void p(int [ ] m){
          m[0]=5;
     }
}
```

(b)

(i) Consider the class *Try*

```
import java.io.*;
class Try{
     public static void main(String[] args)
                throws IOException{
          System.out.println("please type something in,
                followed by 'enter'");
          int c=System.in.read();
          System.out.println("You typed in " +(char)c);
     }
}
```

Identify casting in the *Try* class. [2 marks]

(ii) Consider the class *Try1*

```
import java.io.*;
class Try1{
     public static void main(String[] args) throws
     IOException{
          System.out.println("please type something in,
               followed by 'enter'");
          int c=System.in.read();
          System.out.println("You typed in " + c);
     }
}
```

Say which one of the following is true: [2 marks]

(A) If the user enters 'k' the output will be '107'.

(B) If the user enters '107' the output will be 'k'.

(C) If the user enters 'k' the output will be 'k'.

(D) None of the above.

(iii) Write a simple class that will display the `char` that has the unicode value 99. [2 marks]

(iv) Which one of the following is the most likely output of the *Chars* class? [3 marks]

```
public class Chars{

     public static void main(String [ ] args){
          System.out.println('6'+'B');
          System.out.println((char)('6'+'B'));
     }
}
```

(A)
```
120
x
```

(B)
```
6B
x
```

(C) No output – compilation error.

(D) No output – runtime error.

(c) Consider the *Q5* class, below: [8 marks]

The *Q5* class is run from the command line. The class attempts to parse the `String` entered by the user to an `int`. If successful it prints a message to the user based on the size of the number entered by the user. If unsuccessful, the program enters a loop, asking the user to enter a number. The loop continues until the user enters a `String` that can be parsed to an `int`. When the loop ends the class prints a message to the user based on the size of the number entered by the user.

The program has some repeated statements. Rewrite the class to remove repetition. Your revised *Q5* class should behave in the same way as the original class.

```
import java.util.Scanner;

public class Q5{

  public static void askUserForNumber(String a) {
     boolean validInput = false;
     try {
          int number = Integer.parseInt(a);
          validInput = true;
          if (number < 100) System.out.println
                    ("Your number is small.");
          else if (number > 1000000) System.out.println
                    ("Your number is big.");
          else System.out.println
                    ("Your number is nothing special.");
     }
     catch (Exception e) {
          System.out.println("\""+a+"\" is not
                     a valid number.");
     }

     Scanner scanner = new Scanner(System.in);
     String input = "";
     while(!validInput) {
          try {
            System.out.print("Enter a number: ");
            input = scanner.nextLine();
            int number = Integer.parseInt(input);
            validInput = true;
            if (number < 100) System.out.println
                    ("Your number is small.");
            else if (number > 1000000) System.out.println
                    ("Your number is big.");
            else System.out.println
                    ("Your number is nothing special.");
          }
          catch (Exception e) {
            System.out.println("\""+a+"\" is not
                     a valid number.");
          }
     }
  }

  public static void main(String[] args) {
     askUserForNumber(args[0]);
  }
}
```

**Question 6**

(a) Answer true or false to each of the following statements: [8 marks]

(A) A constructor must have the same name as the name of its containing class.

(B) Constructors do not have return types.

(C) A constructor is typically used to give values to the object's instance variables.

(D) A class can have up to three constructors.

(E) When one class extends another, the keyword `super` can be used in a constructor of the extending class to access a constructor in the class that is being extended.

(F) An instance method has the keyword `static` in its heading.

(G) `Person extends SentientBeing` means that the *Person* class has all the fields and instance methods of the *SentientBeing* class.

(H) An inheriting class can redefine instance methods from its parent class by overriding them.

(b) The class *ToDoList* has the following output from the test statements in its main method:

```
Action: Submit 109 cwk with high importance
Action taken status: true
```

Consider the following numbered (from 1 to 11) jumbled up fragments of program code from the class *ToDoList*:

```
1.          ToDoList toDoList = new ToDoList("Submit 109
                 cwk","high",true);
            System.out.println(toDoList);
       }

2.  public class ToDoList {

       private String item;
       private String importance;

3.     public static void main(String[] args) {
            //test statements
```

```
4.   public String toString(){
          String s = ("Action: "+item+" with ");
          s = s+(importance+" importance");
          s = s+("\nAction taken status: "+done);

5.   private boolean done;

6.   public boolean isDone() {


7.        return done;
     }

8.        this.item = item;
          this.importance=i;
          this.done = done;
     }

9.        return s;
     }

10.  public ToDoList(String item,String i,boolean
done){

11.  }
```

Using **all** the code fragments (and nothing else) assemble the fragments in the right order such that the *ToDoList* class will compile and give the above output when run. [9 marks]

Your program should follow convention in that the order of appearance of the different elements should be: instance variables; constructor; instance methods; and finally, the main method.

(c) Consider the *Exists* class below:

```
public abstract class Exists {
     abstract boolean isContained(int[]x, int y);
}
```

Write a class *IntExists* that extends *Exists* and implements its *isContained(int[ ], int)* method. [8 marks]

The method should return true if the `int` search item is found in the array, and false if the search item is not found in the array.

**END OF PAPER**