

THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

UNIVERSITY OF LONDON

CO1109 ZA

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.

© University of London 2018

Question 1

(a)

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

- (A) Asking the user to enter a number between 0 and 9 inclusive.
- (B) Adding together all of the items in an int Array.
- (C) Printing the last 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 second part of the *for* loop, *i < 3*, 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(;;) System.out.println("Hello world");  
    }  
}
```

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

- (A) No output because the program will not compile.
- (B) Infinite loop – the standard output will fill up with “Hello world”, one to a line, and this will continue indefinitely.
- (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<9;i++) System.out.print("*");`
 - (ii) `int j=18; while (j>6){System.out.print("*");j=j-2;}`
 - (iii) `for(int i=-2;i<4;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<9;i=i+2) System.out.print("*");`
 - (vii) `int p=0; while(p<=8){System.out.print("*");p+=2;}`
 - (viii) `for(int i=5; i>=5; i++) System.out.print("*");`
 - (ix) `int m=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 [8 marks] the two missing methods in the *Verifier* class.

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 first error only and recompile.
- (B) Correct the final 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) *Public* should be *public*.
- (B) *Args* should be *args*.
- (C) Missing statements in main.
- (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) unreachable statement

(B) reached end of program while parsing

(C) 'else' without 'if'

(D) None of the above.

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

```
public class Da{  
    static int k;  
  
    static int add(int x, int y){  
        int temp = x + y;  
    }  
  
    public static void main(String[] args){  
        k = add(5, 19);  
    }  
}
```

(A) missing return statement

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

(C) variable k has not been initialised

(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 ("Camelot");`
 - (ii) `Integer.parseInt(10);`
 - (iii) `int a = Math.abs ("Camelot".length());`
 - (iv) `Integer.parseInt ("350");`
 - (v) `"threelfifty".compareTo ("350");`
 - (vi) `int z = "threelfifty".compareTo ("350");`
 - (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:

<u>Parameter</u>	<u>Returns</u>
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 Boolex{
    static int x; static int y;
    public static void main(String[] args){
        if x > y System.out.print("error");
    }
}
```

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

```
public class Bool2{
    public static void main(String[] args){
        if (true) 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(!(!false));
    }
}
```

- (A) true
- (B) false
- (C) No output – the program will not compile.
- (D) None of the above.

- (iv) Consider class *Bool3*:

```

import java.util.Scanner;

public class Bool3{

    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 *Bool3* [2 marks] when the user enters 3?

- (A) The program will output hello
- (B) The program will output nothing at all.
- (C) The program will output 0
- (D) The program will not compile.
- (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("1+1");
        System.out.println(1+1);
        System.out.println("1"+"1");
    }
}

```

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

```

class Division{
    public static void main(String[] args){
        System.out.println(3/2);
        System.out.println(3.0/2);
        System.out.println(3/2.0);
    }
}

```

- (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(5*2+1);  
        /*2*/     System.out.println(5*(2+1));  
        /*3*/     System.out.println((5*2)+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 file closes.
 - (C) When the *bling()* method reads in the end of file character.
 - (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 output 0
 - (C) The program will output the text contained in the text file 'filey.java'.
 - (D) The program will stop with a run-time error.
 - (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) The program would end with an *InputMismatchException*
(C) There would be no output as the condition *t != -1* would be false at the beginning.
(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) *FileNotFoundException*
(C) *NumberFormatException*
(D) *IOException*
(E) None of the above.

(b)

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

```
public class Ex{  
    public static void main(String[] args){  
        try{  
            Integer.parseInt("alpha");  
            System.out.println("beta");  
        }  
        catch(Exception e){  
            System.out.println("gamma");  
        }  
    }  
}
```

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

```

public class Yex{

    public static void main(String[] args){
        int x = 5; int y = 2;
        try{
            int z = x/y;
            System.out.println("Proceed");
        }
        catch(Exception e){
            System.out.println("divide by zero");
        }
    }
}

```

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

```

class ArrayTest{
    public static void main(String[] args){
        int k=6;
        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 *ArrayTest* class [2 marks] is correct?

- (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 and output:

```

0
1
2
3
4
5
6

```

- (D) 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("d1;fkas;lkf");
    }
}
```

- (c) The file *num1.txt* has the following contents:

10
200
450
100
90

Write a method with the heading

[8 marks]

```
public static void divideFileContents()
```

that does the following:

- Opens *num1.txt* to read from.
- Opens a file called *num2.txt* to write to.
- Reads each number from *num1.txt*, divides it by 10, and writes it to *num2.txt*.
- Writes numbers to *num2.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 primitive variables: [2 marks]

Boolean
char
String
int

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

- (A) When you make an assignment to a reference variable, the variable does not hold the value assigned, but instead points to it in memory.
- (B) Java allows one byte (=8 bits) to store characters. This means that in Java we can represent 2^8 different characters.

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

```
class StringParam{
    public static void main(String [ ] args) {
        String s="hello";
        p(s);
        System.out.println(s);
    }

    static void p(String m) {
        m="goodbye";
    }
}
```

- (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]=7;
    }
}

```

(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);
    }
}

```

If the output of *Try1* is 107, say which one of the following [2 marks] statements is true:

- (A) The user entered 107.
- (B) 107 is the unicode value of the first char the user entered.
- (C) 107 is the unicode value of the last char the user entered.
- (D) None of the above.

(iii) Write a simple class that will display the char that has the [2 marks] unicode value 37.

(iv) Which one of the following is the most likely output of the [3 marks] *Chars1* class below?

```
public class Chars1{  
  
    public static void main(String [ ] args){  
        System.out.println('8'+'A');  
        System.out.println((char)('8'+'A'));  
    }  
}
```

- (A) 8A
Y
- (B) 121
Y
- (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 have return types.
 - (C) A constructor is typically used to give values to the object's instance variables.
 - (D) A class can have arbitrarily many constructors.
 - (E) The keyword `this` can be used in a constructor to distinguish the formal parameter from the field name.
 - (F) We can only use an instance method in conjunction with an instance of the class.
 - (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 String toString(){
        String s = ("Action: "+item+" with ");
        s = s+(importance+" importance");
        s = s+("\nAction taken status: "+done);
```
3.

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

```

4. public class ToDoList {

    private String item;
    private String importance;

5.     return s;
}

6.     public boolean isDone() {

7.         return done;
}

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

9.         private boolean done;

10.        this.item = item;
11.        this.importance = i;
12.        this.done = 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(String[] x, String y);
}

```

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

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

END OF PAPER