

THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

UNIVERSITY OF LONDON

291 0211 ZB

BSc Examination
for External Students

COMPUTING AND INFORMATION SYSTEMS

Computer Programming Paradigms

Date: Monday 11 May 2009 : 2.30 – 5.30 pm

Duration: 3 hours

There are ten questions on this paper. Candidates should answer **SIX** questions. Full marks will be awarded for complete answers to **SIX** questions.

Candidates must answer **THREE** questions from **Section A** and **THREE** questions from **Section B**. Candidates must answer at least **ONE** question on Prolog in **Section B**. There are 150 marks on this paper.

A hand held calculator may be used when answering questions on this paper but it must not be pre-programmed or able to display graphics, texts or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

© University of London 2009

UL09/891

Page 1 of 12

Section A

Question 1

- (a) Consider the Java program below. Write down what will be displayed on the screen on completion of the execution of class C. [5]

```
class A {
    void red() {
        System.out.print("Red");
    }

    void yellow() {
        System.out.print("Yellow");
    }
}

class B extends A {
    void blue() {
        System.out.print("Blue");
        super.yellow();
    }

    void red() {
        System.out.print("Pink");
        super.red();
    }
}

class C {
    public static void main (String [] args) {
        A a = new A ();
        B b = new B ();
        a.red();
        b.red();
        System.out.print("\n");
    }
}
```

- (b) Using the classes given in part (a) above as an example, discuss the hierarchical relationship between class A and class B and describe how one method may *override* another. [5]

2910211 2009

(question continues on next page)

- (c) Using overloading approach, write two methods to print out a date. The first method takes *two* arguments namely month and year, and displays on the screen a simplified date such as "May, 2004". The second method takes *three* arguments namely day, month and year, and prints out a full date such as "15, May, 2004". [5]
- (d) Explain the term *applet* in the context of Java. There is a syntax error in the following HTML statement calling an applet. Identify and correct the error. [5]

```
<html>  
<applet code="appletText1.java" width=300 height=200></applet>  
</html>
```

- (e) Write in Java a boolean type method `negative` which takes an array of integers as a single argument and determines if all the integers in the array are negative. [5]

Question 2

- (a) Write a class `Sphere` that takes (x, y) and displays a sphere on the screen. [5]
- (b) Write a class `Bubble` that extends `Sphere` in part (a) above that takes a radius of value r , and returns a double sized sphere with a radius of $2r$. [5]
- (c) Discuss how `Bubble` inherits from `Sphere` in terms of functionalities. Explain what is meant by *overriding* and how overriding is used in your program. [5]
- (d) The mean of a set of real numbers $x_i, i = 1, 2, \dots, n$ is defined as

$$\frac{1}{n} \sum_{i=1}^n x_i$$

Write a method in Java that takes an array of the real numbers and n as the input and return their mean. [5]

- (e) Consider the following two methods in Java. Indicate whether or not the use of the same variable name 'k' twice is a problem. Give your reason. [5]

```
class testClashName1 {
    private void methodOne(int y, int a) {
        int k = 0;
        // more code here ...
    }

    private void methodTwo(int b, int x) {
        int k=1;
        // more code here ...
    }

    public static void main(String[] args) {
        testClashName1 t = new testClashName1();
        t.methodOne(1,2);
        t.methodTwo(1,2);
    }
}
```

Question 3

- (a) Analyse the following Java classes. What would be displayed on the screen after execution of the program? Draw a diagram to show the details of your answer. [10]

```
import java.awt.*;
import java.awt.event.*;

class FlagMaker1 extends Frame {
    FlagMaker1 () {
        add ("Center", new Flag());
        setTitle("A Flag");
        setSize(300,200);
        setVisible (true);
    }

    public static void main (String [] args) {
        new FlagMaker1 ();
    }
}

class Flag extends Canvas {
    public void paint (Graphics g) {
        g.setColor (Color.black);
        g.fillRect (40, 40, 200, 40);
        g.setColor (Color.red);
        g.fillRect (40,80,200,40);
        g.setColor (Color.yellow);
        g.fillRect (40, 120, 200, 40);

        g.setColor (Color.black);
        g.drawString ("Germany", 100, 180);
    }
}
```

- (b) The program above cannot close the display window easily. Modify the program so the display window can be closed on request of the user. [5]
- (c) Explain briefly what an *exception* is. Further, describe the *four* essential things that have to be done in a Java program in order to handle exceptions. [10]

2910211 2009

Question 4

- (a) Write a while statement that is equivalent to the following for statement, where max is a constant. [5]

```
for (int i=0; i<max; i++)
    System.out.println(i);
```

- (b) Write a Java method `triangle` that takes the number of lines as an input and displays a triangle of odd numbers. An example of a triangle of 5 lines is given by: [5]

```
1
3 3
5 5 5
7 7 7 7
9 9 9 9 9
```

- (c) Explain the following two terms in the context of Java programming:
- (i) accessibility modifier
 - (ii) typed (or classed) method

Provide an example for each term which can be identified from the method below. [10]

```
private boolean isPositive (int x) {
    boolean flag;
    if (x>0) {flag = true;}
    else {flag = false;}
    return flag;
}
```

- (d) Write a method `FirstN` that takes an integer `n` and an array of height values as arguments and displays on the screen the first `n` elements of the array of height values and their indices. [5]

2910211 2009

Question 5

- (a) Consider the inflexibility of the Java program below. Modify the program by defining a *static* method so that the program can compute the sum of any two integers. [5]

```
class Test {
    public static void main(String[] argus) {
        int a=2; int b=3;
        int sum=a+b;
        System.out.println("sum(2,3)=" + sum);
    }
}
```

- (b) Draw a diagram to show the data structure of a doubly-linked-list node. [2]
- (c) Write a method to construct a doubly linked list with n nodes. You should first define a node class for the linked list. Suppose each node of the list consists of one *integer data* part and two *link* parts to the *previous* and *next* node. You may then define n as a formal parameter of the method, and the data may be input from the keyboard. [13]
- (d) With the aid of the statements below as an example, explain briefly what is meant by *actual parameter*. [5]

```
private short difference(short x, short y) {
    return x-y;
}
...
System.out.println(different(2.1, 3.2));
```

Section B

Question 6

- a)
- i) What is an *infix* function? Give one example of a predefined infix function in SML. [3]
- ii) Describe, giving examples, the concepts of *association* and *precedence* that need to be considered when defining a function as *infix*. [4]
- iii) Define a function q that given two integers, (x and y , say) returns the difference between their squares so that $q(1, 2)$ results in -3 (that is $1-4$). [2]
- iv) Describe the syntax required to convert function q in iii) above into an *infix* one, explaining clearly how issues of *association* are dealt with. [4]
- v) Explain how two different results might be expected of the evaluation of $1\ q\ 2\ q\ 3$ and the circumstances under which they might be obtained. [2]
- b)
- i) Explain the concept of currying a function, giving examples and explaining the notation used in SML. [6]
- ii) Consider the SML functions defined by:
 $fun\ h(x, y) = x*x + y*y;$
 $fun\ g\ x\ y = x*x + y*y;$

Give, with explanations, the results (values, types and errors returned) of evaluating the following:

$h;$
 $g;$
 $h\ 4;$
 $g\ 4;$
 $h\ 2\ 3;$
 $g\ 2\ 3;$
 $h(3, 4);$
 $g(3, 4);$

[4]

Question 7

a) SML is said to be a *strongly typed language*. Giving suitable examples, explain what this means and give an example of a feature of SML that breaks the ideal of strong typing.

[4]

b) Consider the SML code:

```
abstype bag = emptybag | add of int * bag
with
  val empty1 = emptybag;
  fun member(x, emptybag) = false |
      member(x, add(element, bag)) = x = element orelse member(x, bag);
  fun empty(emptybag) = true | empty(bag) = false;
  fun next(add(element, bag)) = element;
  fun add1(element, bag) = add(element, bag);
end;
```

i) Explain the meaning and use of each line of the SML code above.

[5]

ii) Give the output produced by the following evaluations:

```
empty1;
val a1 = add1(2, empty1);
val a1 = add1(1, a1);
val a1 = add1(2, a1);
val a1 = add1(1, a1);
if not (empty(a1)) then (a1, next(a1)) else (empty1, 0) ;
```

[5]

iii) Write an SML function *setit(bag)* which, given a *bag* returns a list of the contents of *bag* in increasing order and with duplicates left out. For example *setit(add1(2,add1(1,add1(2,empty1))))* should return *[1, 2]*. You may need to write helper functions to do this.

[11]

Question 8

a) Distinguish between the SML concepts of List, Tuple and Record, giving examples of their definition and use.

[10]

b) Define a record type 'student' that holds the following pieces of information: *student number, name, date of birth, courses taken and results obtained*. (You will need to decide on the type (and possibly structure) of each of these.)

[6]

c) Define functions that extract each component.

[2]

d) Define a function *average* that, given such a record, produces the average result of examinations taken by the student and give an explanation of how it works.

[7]

Question 9

a)

- i) Give the rules that determine when two Prolog terms match. [4]
- ii) State which, if any, of the following pairs match, giving your reasoning fully for each pair: [6]
- $h(i,j(K))$ with $h(K, L)$
 - $m([n|p],q)$ with $m([R],S)$.
 - $t(u(V))$ with $t(W)$.
 - $a(b,[c, D],e(F,g))$ with $a(b,D,e(f,C))$

b) Consider the following Prolog predicate:

```
insert(H, [], [H]).
insert(H, L, [H|L]).
insert(H, [_|T], [_|R]):-insert(H, T, R).
```

- i) Describe the relationship between the three parameters of *insert* above. [3]
- ii) Trace the execution of the following query if ‘;’ is typed after each solution is presented: $insert(l,[a,b,c], L)$. [6]
- iii) Modify *insert* so that duplicate solutions are not found for this query. [2]
- iv) Write a Prolog predicate *permute(L, P)* which takes two lists as parameters and returns list *P* as a permutation of list *L* in such a way that if there are no duplicates in *L* there are none in the set of solutions returned if ‘;’ is typed as each solution is presented.

So query $permute([a,b,c], P)$ results in the following exchange:

```
?- permute([a,b,c],P).
P = [a, b, c] ;
P = [b, a, c] ;
P = [b, c, a] ;
P = [a, c, b] ;
P = [c, a, b] ;
P = [c, b, a] ;
no
?-
```

[4]

Question 10

Prolog is claimed to be a 'a declarative language based upon predicate logic' (guide page 79).

a) Justify this statement by giving those features that are in accord with this description.

[8]

b) On the other hand there are aspects of Prolog that do not fit well with this statement.

i) Give 3 of these aspects together with a description of why you think they do not fit the statement.

[6]

ii) For each of the three aspects that you gave in i) above, describe why that aspect is important to Prolog and how it is used by programmers.

[6]

c) Distinguish between the declarative and the procedural readings of a Prolog program, and explain why a programmer needs to be able to use both readings when writing or debugging Prolog programs.

[5]

END OF EXAMINATION