Department of Computer Science
University College London

Cover Sheet for Examination Paper to be sat in May 2001

COMPD0a/COMPC1: Basic Paper I

Time allowed 2.5 hours

Calculators are NOT allowed

Answer ALL questions

Checked by First Examiner:	Date:
Approved by External Examiner:	Date:

Answer ALL questions

Calculators are NOT allowed

1. Answer any **two** of parts a), b) and c)

   a) i) Under what circumstances should a `for` loop, as opposed to a `while` loop, be used in a Java program? When should a `while` loop be used?

   [4 marks]

   ii) Write a Java method that prints a ladder pattern (see below) to the standard output. Your method should take three integer parameters that define the size and shape of the ladder:

   - one to specify the number of rungs in the ladder,
   - one to specify the vertical separation of the rungs, and
   - one to specify the width of the ladder.

   Two examples are shown below:

Number of rungs = 4 Vertical rung separation = 3 Ladder width = 5	Number of rungs = 2 Vertical rung separation = 6 Ladder width = 8

Output:
`#   #`
`#####`
`#   #`
`#   #`
`#####`
`#   #`
`#   #`
`#####`
`#   #`
`#   #`
`#####`
`#   #`
	Output:
	`#      #`
	`#      #`
	`#      #`
	`########`
	`#      #`
	`#      #`
	`#      #`
	`#      #`
	`#      #`
	`########`
	`#      #`
	`#      #`

   Minor errors in Java syntax will not be penalised.

   [12 marks]

[Question 1 continued on next page]

[TURN OVER]

[Question 1 continued]

b) i) Explain briefly how the mechanisms differ in Java for passing primitive and non-primitive types as parameters to methods.

[4 marks]

ii) Write down the output of the following Java program:

```
class ExamProgram {

  public static void main(String[] args)
  {
    int[] a = {8, 4, 9, 3, -8, 13};
    int n = 2;
    printAAndN(a, n);
    incFromN(a, n);
    printAAndN(a, n);
  }

  public static void incFromN(int[] a, int n)
  {
    printAAndN(a, n);

    while(n < a.length) {
      a[n]++;
      n++;
    }

    printAAndN(a, n);
  }

  public static void printAAndN(int[] a, int n)
  {
    for(int i=0; i<a.length; i++) {
      System.out.print(a[i] + " ");
    }
    System.out.println();
    System.out.println(n);
    System.out.println();
  }
}
```

[6 marks]

iii) Explain in one sentence what the method `incFromN` does.

[2 marks]

[Question 1b continued on next page]

[Question 1b continued]

iv) Rewrite `incFromN` so that

- The size of the increment is a parameter to the method.
- Invalid values of parameter n are checked for and dealt with in a sensible way.
- The method no longer prints any output.
- The method takes into account a general rule of programming, which says that it is bad practice to alter the values of method parameters with primitive type inside the body of the method.

[4 marks]

c) You have been asked to implement a program to administer a collection of cooking recipes submitted by readers of an on-line magazine. Each recipe consists of the following pieces of information:

- Title of the dish
- A chunk of text describing the ingredients and method
- Name of the author
- A unique, integer valued ID code.

i) Each recipe in the system is to be represented by an object of user defined type `Recipe`. Assume that all four pieces of information will be known before a particular `Recipe` object is created. Further assume that once a `Recipe` object has been created, its contents never need to be changed, only retrieved. Write out in full a Java class that can be used to instantiate `Recipe` objects.

[8 marks]

ii) Write a second class `CookBook` that will be used to store a collection of `Recipe` objects. Your class should contain:

- A suitable container for the collection of recipes
- An appropriate constructor
- A public interface method, `addRecipe`, that adds a new recipe to the cookbook, given the three text parts of the recipe
- Another public interface method, `getRecipe`, that returns the `Recipe` object containing a specified ID code.

The unique ID codes can follow the order of insertion of the recipes into the cookbook.

[8 marks]

2. Answer any **two** of parts a), b) or c).

a) i) A computer uses *Direct Memory Access* (DMA) to read from its disc. The disc has 100 512-byte sectors per track. The disc rotation time is 16 ms. The bus is 16 bits wide and bus transfers take 0.5 μs each. What proportion of bus capacity can disc DMA consume?

[4 Marks]

ii) What is meant by file *fragmentation*? Why does fragmentation occur and what effect does it have on file-system performance?

[4 Marks]

iii) A proposed new file system has a special file called a *Master File Index* (MFI) on each disc. The MFI includes exactly one entry for each file stored on the disc. Entries have the following, fixed-length format:

File Name (256 bytes)	File Attributes (256 bytes)	List of disc cluster numbers (4096 bytes)

Here a *cluster* is a between 1 and 1024 disc blocks; the exact number of blocks per cluster being fixed for a particular disc. A unique 32-bit number identifies each cluster. The "*list of disc cluster numbers*" is an ordered list with one entry for each cluster belonging to the file. The maximum size of the list is 4096 bytes.

A particular disc has 512-byte blocks and clusters consist of 64 blocks. What is the maximum possible file size? [You may give your answer as a power of 2]

[2 marks]

iv) Comment on the effectiveness of the scheme in iii) for managing large discs. How would you implement *directories* in a hierarchical file system? How easy would it be to implement multiple *links* to files?

[6 marks]

[Question 2 continued on next page]

b) A virtual memory has 64 virtual pages of 1024 bytes implemented on a physical memory of 32 Kbytes.

i) How many bits must be allocated to page numbers and page offsets?

[2 marks]

ii) The page table for a process is currently as follows:

Virtual page number	**Physical page frame**
0	3
1	4
2	Not in memory
3	Not in memory
4	5

Give, where possible, the physical addresses (in hex) corresponding to the following virtual addresses:

`0x1234, 0x0ba6, 0x0538`

[3 marks]

iii) Explain the role which might be played by an *associative memory cache* in the implementation of this paging scheme. Why is such a cache desirable?

[5 Marks]

iv) The Intel Pentium processor employs a two-level page table scheme. Explain how this scheme works and what advantage it brings.

[6 marks]

[Question 2 continued on next page]

[TURN OVER]

[Question 2 continued]

c) i) What is a *system call* and how is one normally invoked?

[2 marks]

ii) Two processes share a processor and are managed by a pre-emptive scheduler. Process A is copying data between two files in 512-byte blocks, process B is attempting to factorise a very large number. Which process should be awarded the highest priority and why?

[3 marks]

iii) Draw a time-sequence diagram to illustrate likely state changes that take place in processes A and B from ii) during a typical period of processing. Indicate on the diagram the events which trigger state changes.

[5 marks]

iv) A third process, process C is added to the processes in ii). Process C is continually reading blocks from a small file, performing some processing on the blocks and writing them back to the **same** file. Discuss the likely performance characteristics of process C compared with those of processes A and B.

[6 marks]

3. Answer any **two** of parts a), b) or c).

a) i) What is the purpose of the *local variable array* (LVA) in the *Java virtual machine* (JVM)? When is storage for the LVA allocated and recovered? How do Java bytecodes reference local variables?

[3 marks]

ii) In the context of the JVM, what is a "*just in time*" (JIT) compiler? Explain how a JIT compiler can improve the performance of the JVM.

[4 marks]

iii) An implementation of the JVM on the Motorola M68000 employs a stack which grows downwards in memory. Register a7 is used as a stack pointer. Register a6 points at the base of the LVA (which also grows downwards in memory). The stack and the LVA are both implemented in terms of 32-bit words. Write a **single** M68000 assembler instruction which pushes the third item of the LVA onto the stack. [You will not be penalised for minor syntax errors. Explain any novel notations you introduce]

[3 Marks]

iv) Assuming that registers a6 and a7 are used as described in iii), outline the stack manipulations that must be done when one Java method calls another. [You do not need to give precise details of any particular implementation nor do you need to write code. However, you should explain what needs to be stored on the stack and why]

[6 marks]

[Question 3 continued on next page]

[Question 3 continued]

b) i) A processor executes instructions in three phases instruction fetch (I), instruction execution (E), and data movement to/from memory (D). The diagram below shows a non-pipelined execution of a sequence of instructions. [In each case the register on the right is the one modified by the instruction.]

`mov.l aaa, d0`	I	E	D							
`mov.l bbb, d1`				I	E	D				
`add.l d1,  d2`							I	E		
`add.l d0,  d2`									I	E

A pipelined processor has separate I, E and D units and can overlap the I, E and D phases. Show how the instructions above would be executed on the pipelined processor indicating and explaining any stalls that take place. Is it possible to optimise this code by changing the order of the instructions?

[6 marks]

ii) “*The average execution time for instructions on processor A is 8 nanoseconds whilst that on processor B is 12 nanoseconds. It is clear that processor A is the faster processor.*”

Discuss the statement above.

[5 marks]

iii) Explain what is meant by *microprogramming*. What are its strengths and weaknesses?

[5 marks]

c) i) Why is it useful for processors to support *interrupts*?

[3 marks]

ii) How do interrupting devices identify themselves to the processor?

[2 marks]

iii) Why is it useful to assign priorities to interrupts?

[2 marks]

iv) Discuss the different approaches to implementing interrupt priorities.

[9 marks]

[END OF PAPER]