Computer Science Department
2000 Examinations

D0a - MSc Basic Paper 1

Answer ALL questions

(The use of electronic calculators is not permitted in this examination)

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

a) i) What is a method and what is recursion?

[2 marks]

ii) Why is it not always recommended to use recursion?

[2 marks]

iii) What is method overloading and method overriding?

[2 marks]

iv) Explain how argument values of *primitive types* and *objects* are passed in Java. How do they differ?

[3 marks]

v) Consider the following Java method:

```
public static int search(int x, int[] v)
{
      int low, high, mid;
      low = 0;
      high = v.length – 1;
      while (low <= high) {
            mid = (low + high) / 2;
            if (x < v[mid])
                  high = mid – 1;
            else if (x > v[mid])
                  low = mid + 1;
            else
                  return mid;
      }
      return –1;
}
```

Assuming that v is an array of integers sorted in increasing order, what does the method `search` do and how?

[7 marks]

[Question 1 is continued on the next page]

[Question 1 continued]

b) i) What is inheritance?

[2 marks]

ii) What are the benefits of inheritance?

[2 marks]

iii) Explain how dynamic binding works in Java.

[2 marks]

iv) Employees in a company are either "*hourly paid*", "*sales commissioned*" or "*executive*" for the purpose of calculating their weekly wages or monthly salaries. Employment records are represented by four classes `Employee`, `HourlyPaid`, `SalesCommissioned` and `Executive`. The data to be maintained for the `Employee` and the `HourlyPaid` classes may be summarised as follows:

`Employee` class: name of employee.

`HourlyPaid` class: rate of pay and total weekly hours worked.

The methods used in those classes may be summarised as follows:

`Employee` class: `getName` and `computePay` (as an abstract method).

`HourlyPaid` class: `getRate`, `getHours` and `computePay`.

Write Java code to implement the classes `Employee` and `HourlyPaid`.

[10 marks]

[Question 1 is continued on the next page]

[Question 1 continued]

c) i) What is a Java exception?

[2 marks]

ii) The class `Calculator` represents a simple arithmetic calculator. It has one private data member `mem_store` of type `float` which is not accessible from outside the class and four public member methods that operate on the data: `calculate`, `store`, `recall`, and `clear`.

The member method `calculate` has parameters `(int a, int b, char op)`. `op` may be `'/'`, `'*'`, `'+'`, `'-'`. When called, the method should print on the screen the result of the appropriate expression and store it in `mem_store` by making a call to `store`. For example given:

```
Calculator  calc = new Calculator();
```

`calc.calculate(10, 20, '+')` should print "`The result is 30`".

`calc.store(78.4f)` should store 78.4 in `mem_store`.

`calc.recall()` should return the value stored in `mem_store`.

`calc.clear()` should "clear" `mem_store` to zero.

Given the above information, write Java code to define the class `Calculator` and its member methods. Note that your `calculate` method should use Java's exception handling mechanism to deal with an `ArithmeticException` type exception that will be thrown for division by zero. You are also required to use the `switch` statement as your selection mechanism in your calculate method and **not** an `if` or an `if-else` statement. *Minor errors in Java syntax will not be penalised*.

[10 marks]

iii) You are given the following user defined exception:

```
class DivisionByZeroException extends Exception {
   public DivisionByZeroException(String s) {
              super(s);
   }
}
```

Modify your `calculate` method so that when the `ArithmeticException` type exception is caught, a `DivisionByZeroException` type exception is thrown back at the calling method of `calculate`.

[4 marks]

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

a) *In this question you are asked to write fragments of M68000 assembly code. The meaning of any mnemonics and notations you use must be clear. However, you will not be penalised for minor syntax errors.*

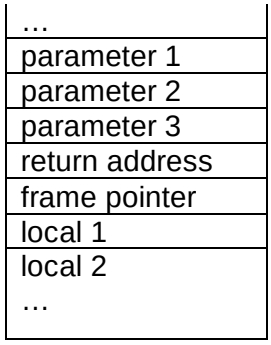
i) What is meant by a "*stack*". Explain how a stack may be implemented efficiently on an M68000-based system. Illustrate the operations you have defined by writing a sequence of assembly language instructions to replace the top two items on a stack by their sum.

[7 marks]

ii) What is "*relative addressing*". The M68000 uses relative addressing for most branch instructions; what is the advantage of this approach?

[4 marks]

iii) The diagram on the right shows a section of memory in a M68000-based system during a procedure call. Each memory location is 32-bits wide and memory addresses are decreasing down the page. Register `a6` stores the address of the memory location which holds the "*frame pointer*". Using *indexed addressing* and register `a6`, write two instructions which copy the value of *parameter 2* to register `d0` and then copy this value into *local 1*.



[3 marks]

iv) Using the scheme of part iii), write an instruction which will copy the value of the frame pointer from the stack into register `a6`.

[2 marks]

[Question 2 continued on next page]

[Question 2 continued]

b) i) "*The Java language is compiled into bytecodes which are then interpreted by the Java Virtual Machine (JVM)*". Explain what this sentence means making it clear how the Java approach differs from that used to implement other mainstream, general-purpose high-level languages. What are the advantages and disadvantages of the Java approach?

[9 marks]

ii) A Java method and the bytecodes generated from it are illustrated below.

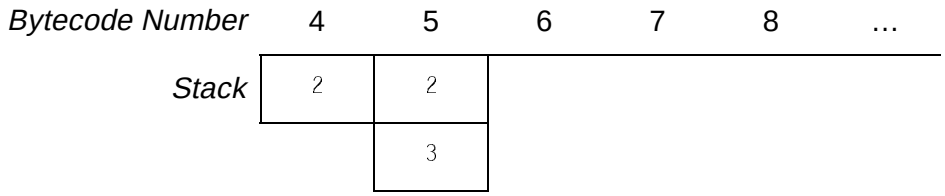
```
public int silly(){
    int a = 2;
    int b = 3;
    if (a + b > 2) return 1;
    return 0;
  }
```

```
Method int silly()
   0 iconst_2
   1 istore_1
   2 iconst_3
   3 istore_2
   4 iload_1
   5 iload_2
   6 iadd
   7 iconst_2
   8 if_icmple 13
  11 iconst_1
  12 ireturn
  13 iconst_0
  14 ireturn
```

What is the purpose of bytecodes 0-3?

[2 marks]

iii) The diagram below illustrates a portion of the JVM stack during the execution of bytecodes 4 and 5 of the method in part ii). (The stack grows downwards.)



Complete this illustration up to but not including the execution of an `ireturn` instruction.

[5 marks]

[Question 2 continued on next page]

[Question 2 continued]

c) i) What is a processor cache? Briefly explain why using such a cache can be a cost-effective way of improving processor performance. Your answer should mention the importance of "locality" in cache performance.

[8 marks]

ii) The diagram shows a cache-line from a direct-mapped system with a 1M byte cache. The system has a 32-bit address space and the line is entry number 4.

Valid	Tag	Data (32 bytes)															
1	2	31	30	29	28	27	26	25	24	23	22	21	20	19	(bytes 2-18 not shown)	1	0

Calculate the memory address (in hexadecimal) of the shaded byte in the diagram. Show your working clearly.

[6 marks]

iii) Using the scheme of part ii), decide whether the addresses `0x00904a94` and `0x01a04a84` collide. Justify your answer.

[2 bytes]

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

a) i) Explain the difference between *polled* and *interrupt-driven* input/output (I/O) and state the advantages and disadvantages of each.

[4 marks]

ii) In the context of general purpose operating systems, **briefly** explain the following terms:

"User process"; "scheduler"; "system call"; "synchronous read".

[4 marks]

iii) A user process executes a system call which performs a synchronous read from the keyboard. Outline the scheduling events and process state changes that are likely to occur before the original process is scheduled once more. (You do not need to describe the fine detail of interrupt handling nor any character-level processing that the operating system may perform.)

[8 marks]

[Question 3 continued on next page]

[Question 3 continued]

b) i) In the context of memory management, **briefly** explain the following terms:

"Virtual memory"; "swapping"; "memory management unit".

[6 marks]

ii) A paged virtual memory system is designed for a processor having a 32-bit address space. The page size is 4K bytes. How many bits should be allocated to the page numbers?

[1 mark]

iii) Explain the role of an associative cache inside a *Memory Management Unit* (MMU). Illustrate, possibly with the aid of a diagram, the operation of an MMU cache in the case of the system in part ii).

[5 marks]

iv) The Intel Pentium processor uses a "*two-level page table*". Explain what this means and why such an arrangement is necessary.

[4 marks]

c) i) "*Disc performance may gradually deteriorate due to file fragmentation*". Explain what is meant by "*file fragmentation*" and why it affects performance.

[2 marks]

ii) The performance of a disc system may be improved by cacheing disc blocks in main memory. Explain why this is so and discuss the strategies which may be employed when a cached block is written to.

[7 marks]

iii) One version of the Unix filestore bases its organisation on the use of "*i-nodes*". Describe the organisation and operation of such a system. Your answer should mention the implementation of *directories*, *file attributes* and *links*.

[7 marks]

[END OF PAPER]