UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

B. Sc. Examination 2013

DEPARTMENT OF COMPUTING

IS52028B Principles and Applications of Programming

Duration: $1\frac{1}{2}$ **hours**

There are two questions in this paper. You should answer BOTH questions. Each question carries 25 marks. The marks for each part of a question are indicated at the end of the part in [.] *brackets.*

There are 50 marks available on this paper.

No calculators should be used.

THIS PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

Question 1 Procedural programming and simple objects

(a) Some programming languages have an exponentiation operator $\uparrow$ which evaluates a^b i.e. $2 \uparrow 3$ evaluates to 8 $(= 2^3)$. Suppose that $\uparrow$ is right associative.

The value of $2 \uparrow 3 \uparrow 2$ is

i. 2^9

ii. 6^2

iii. 2^{32}

iv. undefined - the expression is ambiguous

v. none of the above [5]

(b) Consider the following algorithm:

```
POWER
Input: non-negative integers a, b
Output: a raised to the power b
1. set c to the value of a
2. if b is 1
3.    return c
4. subtract 1 from b and save the result in b
5. multiply c by a and save the result in c
6. go to 2
```

Write a snapshot sequence for `POWER` in the case that immediately before instruction 1, a, b and c have the values $a = 3$, $b = 2$ and $c = 1$ (the first snapshot is `1. a=3 b=2 c=1`). [5]

(c) Write a non-recursive procedural Java method that implements `POWER`. Do not use `Math.pow(double a, double b)`. [5]

(d) Now implement `POWER` with a *recursive* procedural Java method. Do not use `Math.pow(double a, double b)`. [5]

(e) Write a class `MyMaths` that has an implementation of `POWER` as an instance method. Do not use `Math.pow(double a, double b)`. Include test code in `main` to demonstrate how your instance method is called. [5]

Question 2 Advanced objects

(a) A Java class `Book` is defined as follows:

```
public class Book{

    private String title;
    private String[] authorName;
    private boolean isForSale;

    public Book(String t, String[] name, boolean b){
        title = t;
        authorName = name;
        isForSale = b;
    }

    public String getTitle(){
        return title;
    }

    public void setTitle(String t){
        title = t;
    }

    public String[] getAuthor(){
        return authorName;
    }

    public void setAuthorName(String[] name){
        authorName = name;
    }
}
```

Which one of the following statements best describes `Book`?

i. `Book` is correctly encapsulated.

ii. `Book` is not correctly encapsulated because the variable `isForSale` does not have a getter or a setter method.

iii. `Book` is not correctly encapsulated because `Book` does not implement the `Encapsulate` interface.

iv. `Book` is not correctly encapsulated because it lacks a default constructor.

v. None of the above. [5]

(b) Consider this subclass of `Book`:

```
public class Novel extends Book {

    private String summary;

    public Novel(String t, String[] name, boolean b, String s) {
        super(t, name, b);
        summary = s;
    }

    public String getSummary() {
        return summary;
    }

    public void setSummary(String s) {
        summary = s;
    }

    public static void main(String[] args) {

        Book book = new Novel("Java in a peapod",
                new String[] { "A", "Student" }, true,
                "This book...");
        System.out.println(book);
    }
}
```

When `Novel` is executed, this line (or something similar) is printed at the command line:

```
exam.Novel@35960f05
```

The programmer actually intended that the program printed a meaningful representation of the object referred to by `book`. Add code to either `Book` or `Author` so that the programmer's aim is accomplished. [5]

(c) Provide code to demonstrate how `Book` and its subclasses would implement the following interface:

```
public interface Printable {
    public void print();
}
```

[5]

(d) Explain how the JVM uses stacks and stack frames in order to organise its computations. [5]

(e) This program

```
public class Bad {

    public static void main(String[] args) {

        Integer[] buff = new Integer[5000000];

        int i = 0;
        while (true) {

            i++;
            if (i == buff.length)
                i = 0;

            Integer obj = new Integer(i); // line 14
            buff[i] = obj;
            // do something useful with buff[i];
        }
    }
}
```

terminated unexpectedly after several seconds and the following message was printed at the command line:

```
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
at exam.Bad.main(Bad.java:14)
```

Explain what went wrong, and provide code to fix the problem. [5]