

THE AUSTRALIAN NATIONAL UNIVERSITY
First Semester Examination – June 1999

COMP1100
Section D Sample Questions

Study Period: 15 minutes

Time Allowed: 3 hours

Permitted Materials: None

These are sample questions only. You should remember that they do not define the scope of the exam in any way.

Question 1 [32 marks]

These questions involve random numbers and **simulation**.

- (a) One of the standard classes in Eiffel is `STD.RAND`. Objects of this class generate pseudo random numbers.
- When a random number generator is called, what does it do?
 - Are random numbers of type `REAL`, `DOUBLE` or `INTEGER`?
 - What is the significance of the adjective *pseudo*?
 - Why have a seed?

[6 marks]

- (b) Construct a class `COIN`, to model the use of coins where one is thrown in order to make a random binary decision.

Bearing in mind that routines should be either queries or commands, what features are appropriate for coins? State which routines and attributes should be *deferred*, *inherited* and/or *protected*.

What should the role of the Eiffel class `STD.RAND` (or its equivalent) be in this class?

[8 marks]

- (c) (Half baked ideas) Do something similar with playing cards (or Monopoly cards). **[8 marks]**
- (d) (Just an idea.) Do a little simulation of the weather given some statistical rules such as: Every day is like the previous one except there are probabilities associated with changes in temperature and precipitation. **[10 marks]**

Question 2 [32 marks]

These are questions on the **Syntax** of Eiffel.

- (a) Why do we have to know about the syntax of a programming language? Are the reasons different for English? [2 marks]
- (b) The following productions give the syntax for Eiffel conditional commands. Identify the syntactic categories, the literal text and the meta-symbols that are a part of the notation for specifying grammars.

```
Conditional
    if Then_part_list [Else_part] end
Then_part_list
    {Then_part elseif ...}+
Then_part
    Boolean_expression then Compound
Else_part
    else Compound
```

Translate the productions into informal English.

Write suitable production(s) for **Compound**. [10 marks]

- (c) The following productions give the syntax for Eiffel loop statements. Identify the syntactic categories, the literal text and the meta-symbols that are a part of the notation for specifying grammars.

```
Loop
    Initialization [Invariant] [Variant]
    Loop_body end
Initialization
    from Compound
Loop_body
    Exit loop Compound
Exit
    until Boolean_expression
```

Convince the examiner that you understand this fragment of the Eiffel grammar by arguing that the following piece of code is syntactically correct.

```
from index := 1
until index = 65536
loop
    index := index*2;
    io.put_integer(index);
    io.put_new_line
end
```

[10 marks]

- (d) The syntax for Eiffel loop statements is given below using the rail notation.

Question 4 [24 marks]

These are questions about structured programming and **top-down design**.

- (a) What is *structured programming*? What is its relationship to *GOTO statements*? What is *top-down design*? [3 marks]
- (b) Use top-down design to produce code for the following problem. “Read 10 real numbers from standard input and print out the deviations of each from their mean.” [7 marks]
- (c) Use top-down design to produce code for the following problem. “Given a function, f , which maps integers to integers, find the least positive integer, m , such that each of $f(m)$, $f(m+1)$ and $f(m+2)$ is zero.” [6 marks]
- (d) Use top-down design to produce code for the following problem. “Write a query, *perm*, which will say whether or not array, A , with bounds 1 and n has as elements, the integers 1 to n in some order.” [8 marks]

Question 5 [23 marks]

These questions are about **generics** and **reuse**.

- (a) Give at least three advantages of having generic classes for reuse in software engineering. [2 marks]
- (b) What is meant by the statement that the Eiffel’s ARRAY class is a *generic and polymorphic container class*? [3 marks]
- (c) Give three examples of *generic, polymorphic container classes*. [2 marks]
- (d) What are the defining features of the generic STACK class? [3 marks]
- (e) With the aid of a diagram, perhaps, compare and contrast the implementation of a STACK class based on a fixed array and that based on a linked list. [4 marks]
- (f) Suppose one was implementing class $STACK[T]$ using linked lists for objects of class T , and suppose that one of the attributes of the $STACK$ class was `first: NODE[T]`, where objects of class $NODE$ are used to hold objects of class T . Give a suitable definition of class $NODE$. [4 marks]
- (g) If one had a generic class $ORDERED_LIST[T]$ without duplicates, give a nontrivial list (at least half a dozen) of queries that should be implemented. Do the same for commands. [5 marks]