UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering and Systems Science
Department of Computer Science

B.A.(Mod.) Computer Science
Senior Freshman Examination

Trinity Term 2001

2BA2 - Programming Techniques

Saturday 26th May | Regent House | 14.00 –17.00

Dr. Hugh Gibbons

Attempt **FOUR** questions
(in presenting programs explain clearly the design of the Eiffel code)

Qs 1

i) Implement an Eiffel function that will find the square root of a number using the technique of binary search.

```
bin_sqrt(low, high, eps, x : REAL) : REAL is
        require
                low^2 <= x and x < high^2
        ensure
                Result^2 <= x and x < (Result + eps)^2
```

ii) Present an Eiffel routine that will search an array of unordered items. This can be achieved by searching in a linear way from the start of the array.
Assume an array, a, has N items with bounds such that
a.lower = 0 and a.upper = N-1.

If the item, x, is in the array then the index of the item should be returned, i.e. x = a.item(Result). The item may occur more than once in the array and if so then any index can be returned such that x = a.item(Result). But the item looked for may not occur at all in the array and in this case the index N should be returned.

```
linear_search(a:ARRAY[G]; x:G) : INTEGER is
        require
                a /= void
        ensure
                Result = N or else x = a.item(Result)
```

If it is known in advance that the item looked for is somewhere in the array, how would this simplify the linear_search program.

iii) Implement a routine, is_in, that checks if an item, x, is in the unordered array, a.

```
is_in(x : G;  a:ARRAY[G]) : BOOLEAN is
        require
                a /= void
        ensure
                -- Result = x ε a[0 .. n-1]
```

Qs 2

Assume we have an array, a, indexed from 1 to N with integer values.

a) Implement a boolean function,

isa_combination(a:ARRAY[INTEGER]; k, N: INTEGER) : BOOLEAN

that will determine if a sub-array indexed from 1 to k ($1 \le k \le N$) is a combination of k numbers from 1 to N.
For example:
The sequence of values [4,2,9,5] is a combination of 4 numbers from 1 to 10.
The sequences [0,9,4,6] and [4,7,7,1] are not valid combinations from 1 to 10.

b) Present an Eiffel class that will provide a routine that will generate all the combinations of choosing k numbers from 1 to N.

Qs 3

a) Present an Eiffel class that will provide a routine that will Heapsort an array.

b) Given an array, present a Boolean function

isa_heap(a:ARRAY[STRING]) : BOOLEAN

that will determine if the array, a, satisfies the *heap property* that for each index, k,

a.item(k) ≥ a.item(2k) and a.item(k) ≥ a.item(2k+1),

given that these items are within bounds.

Qs 4

Assume we have classes, LIST_BAG and NODE with short forms

```
class interface LIST_BAG [G]
    add(x : G)
    -- Add x, maybe again
    count : INTEGER
    empty : BOOLEAN
    has (x : G) : BOOLEAN
    remove (x : G)
    is_equal(s : LIST_BAG[G])
end -- class LIST_BAG
```

```
class interface NODE[G]
    item : G;
    next : NODE[G];
    set_item(x :G)
    set_next(n : NODE[G])
end -- NODE
```

Implement the routines in the class, LIST_BAG, using linked nodes. Use linked list diagrams in explaining the routines.

Qs 5

a) Explain why there is no Knight's tour (cyclic journey) on an NxN board when N is odd.

b) If N is odd, it may be possible to find a Knight's tour on an NxN board if one square is not used. Present an Eiffel class that will provide a routine that will find a Knight's tour, if any, on NxN board when the bottom right square is not used.
For example: if N = 7 and the top left square is (1,1), then the bottom right square at (7, 7) is not used.

Qs 6

Present an Eiffel class that will provide:

a) A *non-recursive* Eiffel routine using an *explicit stack* that will inorder traverse a binary tree.

b) A *non-recursive* Eiffel routine *without* using an explicit stack that will inorder traverse a binary tree.