| -PAPER CODE NO. COMP205 | EXAMINER : **Dr Frans Coenen, Dr Grant Malcolm**<br>DEPARTMENT : Computer Science Tel. No. **43698, 46794** |
|---|---|


# SUMMER 2001 EXAMINATIONS

Bachelor of Arts : Year 2
Bachelor of Arts : Year 3
Bachelor of Engineering : Year 2
Bachelor of Science : Year 1
Bachelor of Science : Year 2

## COMPARATIVE PROGRAMMING LANGUAGES

**TIME ALLOWED : Two Hours**

---

**INSTRUCTIONS TO CANDIDATES**

Candidates must answer all 12 questions in Section A (3 marks each)
In Section B credit will be given for the best FOUR answers (6 marks each)
In Section C credit will be given for the best TWO answers (20 marks each)

If you attempt to answer more than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

THE UNIVERSITY of LIVERPOOL

SECTION A (COMPULSORY SECTION)

1) In the context of the operation of imperative languages briefly distinguish between "interpretation" and "compilation".

2) Which of the following are usually considered to be "imperative" programming languages: (a) Java, (b) Ada, (c) C, (d) Miranda, (e) Pascal.

3) Some imperative languages (e.g. Pascal, Modula2) support a compound data type known as a "variant record", what is the distinguishing feature of such a record?

4) Given the following piece of code what would you expect the output to be:

```
procedure ENUMERATION_ATTRIBUTES is
   type DAYS_T is (MONDAY, TUESDAY, WEDNESDAY,
      THURSDAY, FRIDAY, SATURDAY, SUNDAY);
      package DAYS_INOUT is new enumeration_io(DAYS_T);
   NEWDAY: DAYS_T := TUESDAYS;

begin
   put(DAYS_T'FIRST); new_line;
   put(DAYS_T'LAST); new_line;
   put(DAYS_T'PRED(NEWDAY)); new_line;
   put(DAYS_T'SUCC(NEWDAY)); new_line;
end ENUMERATION_ATTRIBUTES;
```

5) Suggest, using any imperative language with which you are familiar, how you would define a post-test loop.

6) What is an Abstract Data Type (ADT)?

7) What is meant by "referential transparency"?

8) What is meant by "lazy evaluation"?

9) Using a variable *x* of type *A* and a variable f of type *A? B*, write a ?–term of type *(A? B) ? A? B*. What is the ?–reduet of this term?

10) What is meant by the term "constructor" in Miranda?

11) Briefly describe what is meant by "polymorphism" in Miranda, using the `map` function as illustration (recall that `map` takes a function `f` and a list `l` as arguments, and returns the list formed by applying `f` to each element of `l`).

12) In what way does graph – rewriting provide a more efficient implementation of functional programming languages than term rewriting?

SECTION B (ANSWER FOUR OUT OF SIX)

1) What would you expect the output to be, given the following C program?

```
void main(void)
{
char name1[] = "Charles";
char name2[15];

strcpy(name2,"Dickens");

printf("%s %s\\n",name1,name2)
printf("%c. %c.\\n",name1[0],name2[0]);
printf("%d\\n",sizeof(name1)/sizeof(name1[0]));
printf("%d\\n",&name1[0]);
printf("%d\\n",name1);
```

(6 marks)

2) Given the following context free grammar expressed in EBNF:

< sum > ::= < operand > < operator > < operand > = < number > ;

< operand> ::= < number > | ( < sum> < )

< operator> ::= + | -

< number > ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

which of the following statements are valid (legal statements) and which are not? In each case give a short explanation of your answer.

a) 2 + 4 = 6;
b) 7 - 3 = 2;
c) -4 + 4 = 0;
d) (6 + 7) - 5 = 8;
e) (2 - 4) + (4 - 6) = -4;
f) ((2 + 2) - 4) + 6) = 6;

(6 marks)

3) Given the following table of characters

| | | | |
|---|---|---|---|
| a | b | c | D |
| e | f | g | h |
| i | j | k | l |

a)Suggest a suitable data structure in which such a table may be stored. (2 marks)

b) Using any imperative language with which you are familiar define an appropriate dimensioned data type detailing the data structure suggested in (a).

(2 marks)

c) How would we go about accessing a particular element in the data structure suggested in (a). (2 marks)

4) a) Briefly describe a-conversion. (2 marks)

b) Give a step-by-step reduction of the following ?–term:

(?f.?x. f(fx)) (?f.?x.f(fx)),

indicating whether each step is an instance of a-conversion or ß-conversion. (4 marks)

5) a) State the induction principle for finite lists. (2 marks)

b) Complete the following Miranda definition of a function `append` that concatenates two lists:

```
append [] y = y
append (a:x) y =
```

i.e., `append x y` is equal to `x ++ y.` (1 mark)

c) Give an inductive proof that `append x [] = x` (3 marks)

6) Give a brief characterisation of each of the following kinds of semantics :

a) Operational semantics

b) Denotational semantics

c) Logical (or axiomatic) semantics

(6 marks)


SECTION C

1)a)Using C, define a structure (type name = \fRaddress_t\fR) comprising the following fields which can be used to form a linked list of such records.

| Type | Name |
|---|---|
| int | houseNumber |
| char[32] | streetName |
| char[32] | townName |
| address_t | nextPtr |

(7 marks).

b) Define a function (called `createAddressStruct`) that will dynamically create a structure of the form defined in part (a) given specific values for the first three fields and assuming the last fields takes the value `NULL`, and return a pointer to it.

(7 marks).

c) Suggest a sequence of calls to the `createAddressStruct` function defined in (b) and pointer re-assignments that will result in a linked list comprising the following "records".

24 Peach Street
Liverpool

16 High Street
Manchester

120 Church Street
Birkenhead

(6 marks)

2) One dimensional arrays are typically processed using loop constructs such that the loop variable (counter) doubles up as the array index. Using any imperative language with which you are familiar describe how you would carry out the following:

a) Mappings. For example given the array `{1,2,3,4,5}` multiply each element by `2` to give `{2,4,6,8,10}`. (5 marks)

b) Filtering. For example given the array {1,2,3,4,5} output only does elements which are odd numbers.

(5 marks)

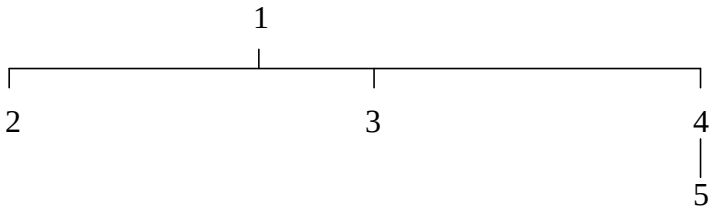
c) Folding. For example given the array {1,2,3,4,5} calculate and output the sum of the element values.

(5 marks)

d)Zipping. For example given the arrays `{0,2,4,6,8}` and `{1,3,5,7,9}` produce the array `{0,1,2,3,4,5,6,7,8,9}`.

(5 marks)

3) "Multi-trees" are trees with a variable branching factor; whereas a binary tree is either a leaf or has two subtrees, a multi-tree can have any number of subtrees (leaves are mulit-trees with zero subtrees). More concretely, a multi-tree consists of an internal label and a list of subtrees. For example, consider multi-trees with internal labels of type `num`, a multi-tree with internal label 1 and three subtrees, the first of which has internal label 2 and no subtrees, the second of which has internal label 3 and no subtrees, the third of which has internal label 4 and one subtree which has internal label 5 and no subtrees, can be pictured as follows:



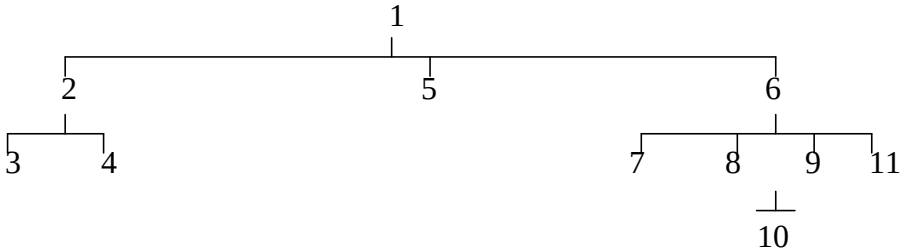

a) Give a Miranda definition of a type `MultiTree*`, where the internal notes are elements of the parameter type *. (6 marks)

b) Give a recessive definition in Miranda of the catamorphism for multi-trees (analogous to `foldr` for lists). Include the types of the catamorphism in your answer. (6 marks)

c) Give a Miranda definition of a function

```
sum Nodes: multiTree num ? num
```

that sums all the internal labels in a multi-tree. For example, given the following multi-tree




`Sum Nodes` returns 1+2+3+4+5+6+7+8+9+10+11 = 66 (8 marks)


4) Given a list `l`, a *segments* of `l` is a list `y` such that `l = x++y++z` for some lists `y` and `z` (`x`, `y` and `z` can be the empty list)

a) Give a Miranda definition of a function

```
segments:: [*]? [[*]]
```

that returns all the segments of a given list. For example, the segments of a list [1,2,3] are: [],[1], [1,2], [1,2,3], [2], [2,3], [3]. (12 marks)

b) Give a Miranda definition of a function

```
Sums Of Segments :: [num] ? [num]
```

That returns the list of sums of all the segments of a list, i.e., each element of the result list is the sum of all the elements in a segment of the argument list. (3 marks)

c) A segment is "flat" if all its elements are the same.
For example, the list [1,3,3] has four flat segments:

[], [1], [3] and [3,3]

give a Miranda definition of a function

```
longest Flat Segment :: [num] ? num
```

that returns the length of the longest flat segment in a list. (5 marks)