THE UNIVERSITY of LIVERPOOL

Section A

Give brief definitions for each term from the list below. Each definition is worth **3 marks**

1. Ω-notation
2. Worst-case running time of an algorithm.
3. Optimisation problem.
4. Spanning tree of a graph.
5. Integer Knapsack problem
6. Church-Turing hypothesis.
7. Divide-and-Conquer algorithm.
8. Undecidable problem.
9. *NP*-complete problem.
10. Polynomial Time complexity.

THE UNIVERSITY of LIVERPOOL

Section B

Answer the following 5 questions. Each question is worth **6 marks**.

1. Consider the following 2 Ada functions, each of which computes the value of x^n.

```
function power_a (x, n : integer) return integer is
  begin
    if n = 1 then
      return x;
    elsif n mod 2 = 0 then
      return power_a (x, n/2)*power_a (x, n/2);
    else
      return x*power_a(x,(n-1)/2)*power_a(x,(n-1)/2);
    end if;
  end power_a;
--*********************************************************
function power_b (x, n : integer) return integer is
  result : integer;
  begin
    if n = 1 then
      return x;
    elsif n mod 2 = 0 = then
      result:= power_b (x, n/2);
      return result*result;
    else
      result:= power_b (x, (n-1)/2);
      return x*result*result;
    end if;
  end power_b;
```

a. Assuming that *n* is an exact power of 2, $n = 2^k$, write down the recurrence relations that define the running times of the function *power_a* and the function *power_b*.

b. **State** the solutions to the recurrence relations you derived in part(a).

c. Briefly describe the *precise* reason why the two functions have different running times.

THE UNIVERSITY of LIVERPOOL

2. Show how a greedy approach may be used to find a minimal spanning tree in the graph below:

3. Give \fBthree\fR examples of intractable decision problems, including the form taken by instances and the decisions being made thereon.

4. Outline how the method known as \fBdiagonalisation\fR is used to prove the existence of undecidable problems.

5. Consider the following Ada function which transposes a given Boolean matrix if each row contains at least one \fBtrue\fR value.

```
.DS
\fBtype\fR \fImatrix\fR \fBis array\fR ($1..n,^1..n$) \fBof boolean\fR;
\fBfunction\fR \fItranspose_if_ok\fR ( \fIA\fR : \fBmatrix\fR ) \fBreturn matrix is\fR
 \fIresult\fR : \fBmatrix\fR;
 \fIall_rows_true\fR : \fBboolean\fR := \fBtrue\fR;
 \fIrow_ok\fR : \fBboolean\fR;
 \fItemp\fR : \fBboolean\fR;
 \fBbegin\fR
  \fBfor\fR $i$ \fBin\fR $1..n$ \fBloop\fR
   $row_ok^:=^$ \fBfalse\fR;
   \fBfor\fR $j$ \fBin\fR $1..n$ \fBloop\fR
    $row_ok^:=^row_ok$ \fBor\fR $A(i,j)$;
    \fBexit when\fR $row_ok$;
   \fBend loop\fR;
   $all_rows_true^:=^all_rows_true$ \fBand\fR $row_ok$;
   \fBexit when\fR \fBnot\fR $all_rows_true$;
  \fBend loop
;
  \fBif\fR $all_rows_true$ \fBthen\fR
   \fBfor\fR $i$ \fBin\fR $1..n$ \fBloop\fR
    \fBfor\fR $j$ \fBin\fR $i+1..n$ \fBloop\fR
```

THE UNIVERSITY of LIVERPOOL

$temp^:=^A(i,j)$; $A(i,j)^:=^A(j,i)$; $A(j,i)^:=^temp$;
\fBend loop\fR;
\fBend loop\fR;
\fBend if\fR;
\fBreturn\fR $A(1..n,1..n)$;
\fBend\fR \fItranspose_if_ok
;
.DE
.IP a)
Describe the form of the \fIbest case\fR input and the resulting \fIbest case\fR
number of operations executed by this function, expressing your answer using $O(..)$
notation.
.IP b)
Describe the form of the \fIworst-case\fR input and the resulting \fIworst-case\fR
running time of this function, again expressing your final answer using $O(..)$ notation.
.KE
.bp
.SH
Section C
.LP
Answer \fBtwo\fR of the three questions below. Each question is work \fB20 marks\fR.
.LP
.LP
\fB1)\fR
.IP a)
Describe the divide-and-conquer algorithm that multiplies two n-digit
integers, expressed in decimal, by recursively computing the products of \fBthree\fR
pairs of $n/2$-digit integers. (\fB12 marks\fR)
.IP b)
Suppose the algorithm could be modified so that the multiplication is performed
by recursively computing the products of \fBfive\fR pairs of $n/4$-digit integers.
.RS
.IP i)
Give the recurrence relation governing the running time of the modified algorithm and
state its solution.
(\fB3 marks\fR).
.IP ii)
Briefly describe what would need to be satisfied by a Divide-and-Conquer algorithm for
integer multiplication, of a similar style to the Karatsuba method,
to achieve a worst-case run-time of $O(^n sup {log sub 8 10}^)$. (
marks\fR)
.RE

```
.bp
.LP
\fB2)\fR
.IP a)
Consider the following two algorithms each of which can be shown to compute
the $n$th \fIFibonacci number\fR, i.e. the $n$th number in the sequence $F sub n$
defined by: $F sub 0^=^F sub 1^=^1$; $F sub n^=^F sub n-1^+^F sub n-2$ ($n^>=^2$).
.LP
.KS
.DS L
\fBfunction\fR \fIfibo_a\fR (\fIn\fR : \fBinteger\fR) \fBreturn integer is\fR
  \fBbegin\fR
    \fBif\fR $n^<=^1$ \fBthen\fR
      \fBreturn\fR $1$;
    \fBelse\fR
      \fBreturn\fR $fibo_a (^n-1^)^+^fibo_a (^n-2^)$;
    \fBend if\fR;
  \fBend\fR $fibo_a$;
.DE
.LP
.DS L
\fBfunction\fR \fIfibo_b\fR (\fIn\fR : \fBinteger\fR) \fBreturn integer is\fR
   $res,^t1,^t2$ : \fBinteger\fR;
   \fBbegin\fR
     $res^:=^1$; $t1^:=^1$; $t2^:=^1$;
     \fBif\fR $n^<=^1$ \fBthen\fR
       \fBreturn\fR $res$;
     \fBelse\fR
       \fBfor\fR \fIi\fR \fBin\fR $2..n$ \fBloop\fR
         $t2^:=^t1$; $t1^:=^res$;
         $res^:=^t1^+^t2$;
       \fBend loop\fR;
       \fBreturn\fR $res$;
     \fBend if\fR;
  \fBend\fR $fibo_b$;
.DE
.KE
.LP
i) Give a brief analysis of the \fIworst-case\fR
running times of both of these functions. (\fB4 marks\fR).
.LP
```

```
ii) What design paradigm is illustrated by the method employed in \fIfibo_b\fR? (\fB2
marks\fR)
.LP
.IP b)
If $A$ is the $2^times^2$ matrix
.EQ
left ( { lpile { {0~1} above {1~1} } } right )
.EN
then it can be shown that the $2^times^2$ matrix $A sup n$ (i.e. $A$ multiplied by itself $n$
times)
is equal to:
.EQ
A sup n~~=~~
left ( { lpile { {F sub n-2~F sub n-1} above {F sub n-1~F sub n} } } right )
.EN
Show how this relationship can be used to define a Divide-and-Conquer algorithm, $fibo_c$,
that computes the $n$th Fibonacci number $F sub n$, where the worst-case running time of
$fibo_c$ is $O(^log^n^)$. Your answer should not only include a description of
the algorithm but also a statement and solution of the recurrences describing its
run-time. (\fB14 marks\fR).
.LP
.bp
\fB3)\fR
A \fIk\fR-colouring of an $n$-node undirected graph $G(V,E)$ is
an assignment $chi^:^V^ra^ls^1,2^,...,^k^rs$ of colours to nodes such
that if $ls^v,w^rs$ is an edge of $G$ then $chi (^v^)^!=^chi (^w^)$.
The graph colouring problem is an optimisation problem defined by:
.LP
\fBInput:\fR Undirected $n$-node graph $G(V,E)$.
.br
\fBOutput:\fR A $k$-colouring of $G$ for which $k$ is minimal.
.LP
.IP a)
Using the observation that if $v$ and $w$ are nodes in $G$ which are
not joined by an edge, then $v$ and $w$ can receive the same colour, describe
an algorithm to find a colouring of $G(V,E)$ that proceeds by merging
non-adjacent nodes $v$ and $w$, i.e. forming a new graph $G sub 1$ from $G$ by
replacing the two nodes $v$ and $w$ with a new node $x$, and adding edges $ls^x,^y^rs$
for each edge $ls^v,y^rs$ or $ls^w,y^rs$ in $G$, and then finding a colouring of the
new $n-1$-node graph $G sub 1$. (\fB10 marks\fR)
.IP b)
If the algorithm terminates when a completely connected graph with $t$ nodes remains,
```

how many colours are used in the final colouring of G returned? (\fB2 marks\fR)
.IP c)
What is the \fIworst-case\fR running-time of this approach? (\fB3 marks\fR)
.IP d)
By considering the graph below and the result of merging nodes 1 and 2, followed
by merging nodes 3 and 4 show that this approach may fail to produce an
optimal colouring. Briefly explain why it is unlikely that such approaches
can be adapated to find an optimal colouring. (\fB5 marks\fR).

```
.KS
.PS
scale = 120
"\s14\fB6\s0\fR"  at 184, 120
"\s14\fB5\s0\fR"  at 24, 120
"\s14\fB4\s0\fR"  at 184, 24
"\s14\fB3\s0\fR"  at 184, 224
"\s14\fB2\s0\fR"  at 24, 24
"\s14\fB1\s0\fR"  at 24, 224
.ps 40
line   from 168, 208 \
         to 40, 136
line   from 24, 200 \
         to 24, 144
line   from 40, 208 \
         to 168, 136
line   from 48, 24 \
         to 160, 24
line   from 184, 96 \
         to 184, 48
line   from 24, 96 \
         to 24, 48
line   from 184, 200 \
         to 184, 144
line   from 48, 120 \
         to 160, 120
circle radius 24 at 184, 24
circle radius 24 at 24, 24
circle radius 24 at 184, 120
circle radius 24 at 24, 224
circle radius 24 at 24, 120
circle radius 24 at 184, 224
.PE
.KE
```