NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

SEMESTER II: 2006–2007

EXAMINATION FOR

GEM 1501 – Problem Solving for Computing

Monday 23 April 2007 Afternoon – Time Allowed 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This examination paper consists of TEN (10) questions and comprises ELEVEN (11) printed pages.
2. Answer **ALL** questions.
3. This is an **Closed Book** examination.
4. Every question counts FIVE (5) Marks which is distributed equally on sub-questions in the case that there are any. The maximum possible marks are 50.
5. Please write your Matriculation Number below:

   MATRICULATION NO: Master Solution

This portion is for examiner's use only

Qestion	Marks	Remarks	Qestion	Marks	Remarks
Q01:			Q06:		
Q02:			Q07:		
Q03:			Q08:		
Q04:			Q09:		
Q05:			Q10:		
			Total:		

Question 1 [5 marks] **GEM 1501**

Answer the following questions to the history of computing.

(a) Who is known for presenting the first non-trivial algorithm in his text-book?
☐ Al-Khowârizmî; ☒ Euclid; ☐ Euler; ☐ Pascal; ☐ Turing.

(b) What is Ada Lovelace be famous for?
☐ Inventing the weaving loom;
☒ Writing programs for Charles' Babbage Analytical Engine;
☐ Working in the office of the first American Census.

(c) What does the abbreviation Fortran stand for?

Formula Translator .

(d) Which is the most famous mathematical theorem where computers played an essential part in finding its proof?
☐ Fermat's Last Theorem;
☐ Existence of NP-Complete sets;
☒ The Four Colour Theorem.

(e) When were the first full-programmable computers available?
☐ Around 1930; ☒ around 1945; ☐ around 1960; ☐ around 1975.

Question 2 [5 marks] **GEM 1501**

Solve the following dynamic programming task. Find the cheapest voyage to go to goal 9 from nodes 1,2,3,4,5, respectively. Here the table of costs.

```
 Costs of direct route from node x to node y,
 only voyage from x to y with x<y possible.

price   1  2  3  4  5  6  7  8  9
 1:        10 17 30 65 77 99 99 99
 2:           10 16 33 69 49 99 99
 3:              10 20 30 30 99 99
 4:                 10 20 19 37 51
 5:                    12 20 42 54
 6:                       10 19 35
 7:                          10 14
 8:                             10
 9:
```

For example, one can go from 6 to 9 best through 7 giving the price of 10 for going from 6 to 7 and 14 for going from 7 to 9. So the costs (total price) is 10+14, that is, 24. Complete this table.

- From 1, the route is 1-2-4-7-9 and costs 59;
- From 2, the route is 2-4-7-9 and costs 49;
- From 3, the route is 3-4-7-9 and costs 43;
- From 4, the route is 4-7-9 and costs 33;
- From 5, the route is 5-7-9 and costs 34;
- From 6, the route is 6-7-9 and costs 24;
- From 7, the route is 7-9 and costs 14;
- From 8, the route is 8-9 and costs 10.

Question 3 [5 marks] **GEM 1501**

Nick's Class (NC) consists of problems which can be computed in parallel fast, RP consists of problems for which a randomized algorithm can find a solution in polynomial time with high probability (provided that the solution exists), NP are those for which a solution can be verified in polynomial time. What are the known inclusions between these problems and which problems are known to be in these classes.

Which are the known inclusions? Check exactly one box.

☐ $NC \subseteq NP \subseteq RP$; ☒ $NC \subseteq RP \subseteq NP$; ☐ $NP \subseteq NC \subseteq RP$;
☐ $NP \subseteq RP \subseteq NC$; ☐ $RP \subseteq NC \subseteq NP$; ☐ $RP \subseteq NP \subseteq NC$.

Which **two** of these classes are known to contain the class P of all polynomial time computable problems?

☐ NC; ☒ NP; ☒ RP.

In which classes are the following problems known to be in, check one box only.

Satisfiability of formulas: does a given formula φ have a satisfying assignment?

☐ NC; ☒ NP-complete.

Determining the maximum of n numbers.

☒ NC; ☐ NP-complete.

Sorting of n numbers.

☒ NC; ☐ NP-complete.

Question 4 [5 marks] **GEM 1501**

Determine the complexity of these sets with respect to the hierarchy of recursively enumerable sets. The entries are the following:

- "decidable" so that an algorithm can decide whether any given element is in the set or not;
- "not r.e." which means not recursively enumerable;
- "r.e. incomplete" which means not decidable, not complete and recursively enumerable;
- "r.e. complete" which means recursively enumerable and complete for the within the recursively enumerable sets.

Now the following problems should be classified with respect to which level of the hierarchy they go. Check one box per row.

The set of all formulas true in Presburger Arithmetic. These are formulas which consist of quantifiers over integer variables and a body containing comparisons between additive terms of variables and integer constants. For example, $\forall x \exists y\,(x+y > 10)$ and $\forall x \exists y\,(x = y + y)$. The first formula is true, the second not as there are odd numbers x. All variables are always quantified.

[x] Decidable; [ ] not r.e.; [ ] r.e. incomplete; [ ] r.e. complete.

Unbounded Monkey Puzzles, that is, the set of all finite tiles such that every area can be covered by tiles of this type.

[ ] Decidable; [x] not r.e.; [ ] r.e. incomplete; [ ] r.e. complete.

The halting problem, that is, the set of all pairs (P, T) such that the Java Script program P halts on input T.

[ ] Decidable; [ ] not r.e.; [ ] r.e. incomplete; [x] r.e. complete.

Satisfiability, that is, the set of all Boolean formulas in logical variables which can be satisfied for some choice of the truth-values of these variables.

[x] Decidable; [ ] not r.e.; [ ] r.e. incomplete; [ ] r.e. complete.

The set of all compressible texts, that is, the set of all texts T such that the Java Script program P shorter than T outputs T without requesting any input.

[ ] Decidable; [ ] not r.e.; [x] r.e. incomplete; [ ] r.e. complete.

Consider the following finite automaton:

```
List of all states: A,B,C;
Starting state: A;
Accepting states: B,C;

Transition-Table of Automaton:

Old State  |   Input  | New State
-----------+----------+----------
   A       |    0     |   A
   A       |    1     |   B
   B       |    0     |   C
   B       |    1     |   A
   C       |    0     |   B
   C       |    1     |   C
```

Which of the following words are accepted by the automaton?

00110 ☐ accept; ☒ reject.

11000 ☐ accept; ☒ reject.

00000 ☐ accept; ☒ reject.

10100 ☒ accept; ☐ reject.

The language accepted of this automaton are numbers represented in binary form. Leading zeroes are irrelevant for acceptance. Which numbers are accepted?

- ☐ Those which are multiples of 2;
- ☐ Those which are not multiples of 2;
- ☐ Those which are multiples of 3;
- ☒ Those which are not multiples of 3;
- ☐ Those which are multiples of 6;
- ☐ Those which are not multiples of 6.

Question 6 [5 marks] **GEM 1501**

Write a Java Script function which does the following: it receives an array of numbers and sums up the squares of its fields. So fsum(a) is $a[0] * a[0] + a[1] * a[1] + a[2] * a[2]$ in the case that the array has three elements.

```
function fsum(a)
  { var n; var b;
    b=0;
    for (n=0;n<a.length;n++)
      { b += a[n]*a[n]; }
    return(b); }
```

Question 7 [**5 marks**] **GEM 1501**

What is the complexity to compute the following function? State it in dependence of the two parameters n,m.

```
function fone(a,b)
  { var n = a.length; var m = b.length;
    var i; var j; var t = a[0]+b[0];
    for (i=0;i<n;i=i+1)
      { for (j=0;j<m;j=j+1)
          { if (a[i]+b[j] < t) { t = a[i]+b[j]; } } }
    return(t); }

function ftwo(a,b)
  { var n = a.length; var m = b.length;
    var i; var j; var t = a[0]+b[0]; var u = 1;
    do { i = u%n; j = u%m; u=u+1;
         if (a[i]+b[j] < t) { t = a[i]+b[j]; } }
      while ((i > 0) || (j > 0))
    return(t); }

function fthree(a,b)
  { var n = a.length; var m = b.length;
    var i; var j; var t = a[0]; var s = b[0];
    for (i=0;i<n;i=i+1) { if (a[i] < t) { t = a[i]; } }
    for (j=0;j<m;j=j+1) { if (b[j] < s) { s = b[j]; } }
    return(s+t); }
```

Let $LCM(n,m)$ be the "least common multiple" of n and m. State the runtime-complexity of fone, ftwo and fthree in dependence of m, n.

The order of the runtime of fone is, for some constant $c > 0$, approximately
☐ $c(n+m)$ ☐ $c \cdot LCM(n,m)$ ☒ $c \cdot n \cdot m$ ☐ $c \cdot n^2 \cdot m^2$ ☐ 2^{c+n+m}.

The order of the runtime of ftwo is, for some constant $c > 0$, approximately
☐ $c(n+m)$ ☒ $c \cdot LCM(n,m)$ ☐ $c \cdot n \cdot m$ ☐ $c \cdot n^2 \cdot m^2$ ☐ 2^{c+n+m}.

The order of the runtime of fthree is, for some constant $c > 0$, approximately
☒ $c(n+m)$ ☐ $c \cdot LCM(n,m)$ ☐ $c \cdot n \cdot m$ ☐ $c \cdot n^2 \cdot m^2$ ☐ 2^{c+n+m}.

Which functions __fone__ and __fthree__ of the functions fone, ftwo and fthree produce the same output on all possible inputs?

Give two values, __3__ for n and __4__ for m, such that all three functions have the same output on input a and b of the respective length.

Question 8 [5 marks] **GEM 1501**

Complete the following counter programs to compute addition, multiplication, remainder, division and exponentiation. Note that counter programs can only add or subtract one. Functions can call each other. Comparisons between variables or variables and numbers are permitted. Each correct function is one mark.

```
Note, inputs x and y are always positive or zero.

function add(x,y)
  { var v = x; var w = y;
    while (v != 0) { w = w+1; v = v-1; }
    return(w); }

function mult(x,y)
  { var v = x; var w = 0;
    while (v != 0) { w = add(w,y); v = v-1; }
    return(w); }

function remai(x,y)
  { var z=x; var u=0;
    while (z!=0)
      { z = z-1;
        u = u+1;
        if (u == y)
          { u = 0; } }
    return(u); }

function div(x,y)
  { var z=x; var u=0; var d = 0;
    while (z!=0)
      { z = z-1;
        u = u+1;
        if (u == y)
          { u = 0;
            d = d+1;  } }
    return(d); }

function expo(x,y)
  { var v = y; var w = 1;
    while (v!=0) { v=v-1; w = mult(w,x); }
    return(w); }
```

Question 9 [5 marks] **GEM 1501**

Complete the following function such that it uses an array to store once computed values. It is called several times. fibonacci(0) is 0, fibonacci(1) and fibonacci(2) are both 1, fibonacci(n+2) is the sum of fibonacci(n) and fibonacci(n+1). The input n is always a natural number. The programming language is Java Script. Every correct line gives a mark.

```
var a = new Array(0,1,2);

function fibonacci(n)

  { if (n<2) { return(n); }

    if (a.length > n) { return(a[n]); }

    if (a.length <= n)

      { fibonacci(n-1);

        a.push(a[n-1]+a[n-2]); }

    return(a[n]);  }
```

Question 10 [5 marks] **GEM 1501**

The body of the following function has several syntax errors. Write a syntactically correct function producing the intended output. Each adequately corrected line gives one mark.

```
function ff(x,y)
  { variable z := 0;
    for (z := 0 to 828 inclusively step +2)
      { if (z=x or z=y) then return(z); }
    if x < y then z := x+y else z := x*y;
    if z>828 then return(z) else return(828); }
```

The program should be written in Java Script.

```
function ff(x,y)
  { var z=0;
    for (z=0;z<=828;z+=2)
      { if ((x==z)||(y==z)) { return(z); } }
    if (x<y) { z = x+y; } else { z = z*y; }
    if (z>828) { return(z); } else {return(828); } }
```

END OF PAPER