Athabasca University

School of Computing & Information Systems

Final Grade Report

Start Date

Please use ink only

REV 1

COMPUTER SCIENCE	COMP 372	sample	**CRN**
Course designation and number		**Form no.**	**Office use only**

Student to complete *(please print)*

Name ______________________________ Student ID number ______________

Last First Middle

Address ______________________________ Tutor Name ______________

______________ Postal Code

Student signature ______________ **Date** ______________

Must be signed and dated to validate examination

Examination invigilator to complete

Note: Please place this form along with the examination and student response(s) in the self-addressed stamped envelope provided.
If there is a problem, please call Examination Services at: **1 - 800 - 788 – 9041 ext. 6579**

Name ______________________________ Telephone ______________

Date exam received by invigilator ______________ E-mail ______________

This is to certify that the above student wrote the examination under my supervision between the hours of ______ (am/pm) and ______ (am/pm)

on the specified date at ______________________________

Name of the examination centre or invigilator's address

Invigilator signature ______________ **Date** ______________

Must be signed and dated to validate examination

Marker to complete

Examination mark ______________
Comments:

Marker signature ______________ **Date** ______________

Final Grade ______________ Credits earned ______________ PASS/FAIL

(Note: Composite grade cannot be calculated until marks are received from the tutor)

Completed by: ______________ **Date** ______________

Students wishing to appeal an examination result must follow the instructions regarding Appeals regulations provided in the Athabasca University Calendar.

COMP 372

Design and Analysis of Algorithms

Sample Examination

(any questions about this, please contact Dr Oscar Lin at oscarl@athabascau.ca)

Instructions to the student:

1. You will be allowed **three (3) hours** to complete the examination.
2. It is a closed-book examination to be completed online without any printed material or electronic devices outside of the online invigilation system (e.g., notes, tapes, cell phone, smartwatch, tablet, etc.). You may NOT use your textbooks or consult with other people while writing this examination.
3. This examination consists of **two** Parts:
4. Part I: True/False questions (20 questions, 40%)
5. Part II: Multiple Choice questions (30 questions, 60%)
6. Legibility is important; an examination that cannot be read cannot be graded.
7. Complete the information requested on the **Final Grade Report** included in your examination package.
8. On completion of the examination, hand the entire examination package (examination questions, examination booklet, and Final Grade Report) to your examination supervisor.

COMP 372

Design and Analysis of Algorithms

Sample Examination

Version for Much Learning

Part I True/False questions (20 questions, 2% for each)

Write **T** for True or **F** for False in the blank before each question.

1. ___ An algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.
2. ___ $n^2 = o(n^2)$
3. ___ $f(n) = O(g(n))$ implies $g(n) = O(f(n))$.
4. ___ $n = O(n^2)$
5. ___ For two algorithms A and B, if $O(A) = n^{100}$ and $O(B) = 10^n$, then algorithm A is the better choice.
6. ___ The recurrence

$$T(n) = \begin{cases} \theta(1), & x = 1 \\ 2T(\frac{n}{2}) + \theta(n), & x \geq 1 \end{cases}$$

could be written equivalently as $T(n) = T(n/2) + \theta(n)$

7. ___ There is a significant difference in the running time f(n) = $\log_{100}(n)$ and the running time g(n) = lg n.
8. ___ F(n) = o(g(n)) means that f(n) is asymptotically smaller than g(n).
9. ___ All functions can be compared asymptotically.
10. ___ F(n) = Ω(g(n)) is like saying f(n) ≥ g(n)
11. ___ The master method can be used to solve all recurrence equations that have the form T(n) = aT(n/b) + f(n).
12. ___ A problem that generates completely new subproblems at each step is a good candidate for a dynamic programming solution.
13. ___ Dynamic programming has the potential to transform exponential-time algorithms to polynomial time.
14. ___ n is $O((\log n)^{\log n})$
15. ___ Every computational problem on input size n can be solved by an algorithm with running time polynomial in n.
16. ___ Given n integers x1, x2, …, Xn, the third smallest number among them can be computed in O(n) time.
17. ___The sum and composition of two polynomials are always polynomials.
18. ___A non-deterministic algorithm is said to be non-deterministic polynomial if the time-efficiency of its verification stage is polynomial.
19. ___ Dynamic programming makes use of previously calculated solution.
20. ___ Approximation algorithms provide bounds on the quality of the solution mathematically.

Part II Multiple-choice questions (30 questions, 2% for each)

1. Suppose $T_1(n) = O(f(n))$ and $T_2(n) = O(f(n))$. Which of the following is UNTRUE?
   a. T1(n) + T2(n) = O(f(n))
   b. T1(n) – T2(n) = O(f(n))
   c. T1(n)/T2(n) = O(1)
   d. T1(n)*T2(n) = O(f(n)2)
   e. None of the above
2. Problems that can be solved in polynomial time are known as?
   a. Intractable
   b. Tractable
   c. Decision
   d. complete
3. ______ is the class of decision problems that can be solved by non-deterministic polynomial algorithms?
   a. NP
   b. P
   c. Hard
   d. Complete

4. Halting problem is an example for?
    a. decidable problem
    b. undecidable problem
    c. complete problem
    d. trackable problem
5. Problems that cannot be solved by any algorithm are called?
    a. tractable problems
    b. intractable problems
    c. undecidable problems
    d. decidable problems
6. How many stages of procedure does a non-deterministic algorithm consist of?
    a. 1
    b. 2
    c. 3
    d. 4
7. When considering different types of algorithms, which one of the following statements is true?
    a. Divide and conquer algorithms are always faster than greedy algorithms for the same problem.
    b. Greedy algorithms are always faster than divide and conquer algorithms for the same problem.
    c. Greedy algorithms give good approximate answers to problems, but never the best possible answer.
    d. Brute-force algorithms can never be faster than a well-designed greedy algorithm for the same problem.
8. Strassen's algorithm for matrix multiplication uses which of the following?
    a. divide and conquer
    b. dynamic programming
    c. greedy method
    d. backtracking
9. Which of the following is NOT a parallel computer? Select one:
    a. A relatively inexpensive desktop or laptop that contains a single multicore integrated-circuit chip that houses multiple processing "cores," each of which is a full-fledged processor that can access a common memory.
    b. A cluster that is built from individual computers—often simple PC-class machines—with a dedicated network interconnecting them.
    c. A supercomputer uses a combination of custom architectures and custom networks to deliver the highest performance in terms of instructions executed per second.
    d. A uniprocessor computer.
10. The ratio $T1/T\infty$ of the work to the span defines the parallelism of the multithreaded computation. What of the following statement is NOT true? Select one:

a. As a ratio, the parallelism denotes the average amount of work that can be performed in parallel for each along the critical path.

b. As an upper bound, the parallelism gives the maximum possible speedup that can be achieved on any number of processors.

c. The parallelism is the total time to execute the entire computation on one computer.

d. The parallelism provides a limit on the possibility of attaining perfect linear speedup.

11. The actual running time of a multithread computation depends on

    Select one:

    a. its work, that is the running time on a single processor
    b. its span
    c. how many processors are available and how the scheduler allocates strands to processors.
    d. not only its work and its span, but also how many processors are available and how the scheduler allocates strands to processors.

12. What is the correct result for gcd(54, 30)? Select one:

    a. 2
    b. 4
    c. 10
    d. 6

13. Euler's totient function or Euler's phi function $\Phi(n)$ is the number of integers in $Z(n)$ that are relatively prime to *n*. What is the correct result of $\Phi(15)$? Select one:

    a. 7
    b. 8
    c. 13
    d. 3.

14. The RSA public-key cryptosystem relies on the dramatic difference between

    Select one:

    a. two prime numbers.
    b. public key and private key.
    c. the ease of finding large prime numbers and the difficulty of factoring the product of two large prime numbers.
    d. ciphertext and digital signature.

15. Assume a = b (mod n) and c is any integer. Which of the following is UNTRUE?

    Select one:

    a. a+c = b+c (mod n)
    b. a*c = b*c (mod n)
    c. ac = bc (mod n)
    d. a+b = b+c (mod n)

16. Complete the following algorithm to find the greatest common divisor of n and m, where n≤m.

    algorithm GCD(n, m)

    if n= 0 then return m

m=m−n
if n≤m then return GCD(n, m)
else return _______

Which of the following should be in blank ______?

a. GCD(n, m-1)
b. GCD(m, n)
c. 1 + GCD(n, m)
d. m/n.
e. None of the above.

17. The time complexity of the algorithm in the last question is Not
    a. O(m).
    b. Ω(m).
    c. Ω (m*n)
    d. O(m*n).
    e. None of the above.
18. We define an abstract problem Q to be
    a. a set I of problem instances.
    b. a set of S of problem solutions.
    c. a binary relation on a set I of problem instances and a set of S of problem solutions.
    d. a problem that has a yes/no solution
19. A decision problem is
    a. a problem with only two possible solutions.
    b. a problem with more than two possible solutions.
    c. a problem for which all known algorithms do not terminate.
    d. undecidable according to the Church-Turing thesis because it does not terminate on a Turing machine.
20. We can view an abstract decision problem as

    Select one:
    a. A function that maps the instances set I to the solution set {0, 1}.
    b. An optimization problem.
    c. A shortest-path problem.
    d. A NP-complete problem.
21. Which of the following strings does not belong to {0, 1}* ?

    Select one:
    a. 1
    b. 1011
    c. 10_10
    d. 11, 11
22. SAT is the decision problem that takes as input a Boolean formula and returns YES if the formula can be satisfied, NO if it cannot. Assume P != NP, which of the following about SAT us UNTRUE?
    a. It's in P

b. It's in NP
c. It's NP-complete
d. It's NP-hard

23. Which of the following does NOT describe the greedy-choice property?
    a. We can assemble a globally optimal solution by making locally optimal (greedy) choices.
    b. When we are considering which choice to make, we make the choice that looks best in the current problem without considering results from sub-problems.
    c. We make a choice at each step, but the choice usually depends on the solution to sub-problems.
    d. We make whatever choice seems best at the moment and then solve the sub-problem that remains.
24. Let X be a sequence {A, B, C, D, D, A, B}. Let Y be another sequence {B, D, C, A, B, A, E}. Which of the following is a LCS of X and Y.
    a. {B, D, A, B}
    b. {A, B}
    c. {A, B, C, D, E}
    d. {D, D}
25. A problem L is NP-Complete if and only if
    a. L is NP
    b. L is NP-Hard
    c. L is NP and NP-Hard
    d. L is non-polynomial

Questions 26-28 are based on the following:

There is a river which flows horizontally through a country. There are N cities on the north side of the river and N cities on the south side of the river. The X coordinates of the N cities on the north side of the river are $n_1, n_2, \ldots, n_N$, and the X coordinates of the N cities on the south side of the river are $s_1, s_2, \ldots, s_N$. Assume that we can only build bridges between cities with the same number; that is, we can only build bridges between cities with coordinates n_i and s_i, where $1 \leq i \leq N$. In this problem, we ask you to determine the maximum number of bridges we can build without any bridges crossing each other. Note that n_1 through n_N and s_1 through s_N are both not sorted.

- Describe your definition of a subproblem. Use that definition, prove that this problem exhibits optimal substructure.
- Describe a dynamic-programming algorithm to solve the problem.
- What is the time complexity of your algorithm?

First we sort the coordinates of N cities on the north side in non-decreasing order. Assume that $a_1, a_2, \ldots, a_N$ is the sorted sequence and b_j is the coordinate of the city at the south side which corresponds to the city located at a_j in the north, that is, for each $1 \leq i \leq N$, if $a_i = n_j \ then \ b_i = s_j, where$ $1 \leq j \leq N$ and $a_1 \leq a_2 \leq \cdots \leq a_n$. In this case, the problem of finding the maximum number of bridges we can build without any bridges crossing with each other is equivalent to finding the longest increasing subsequence of $b_1, \ldots, b_N \ of \ length \ k$, and the sequence formed by the corresponding a_i of that increasing subsequence is also non-decreasing by definition.

26. Let $L(k)$ denote the length of the longest increasing subsequence of $b_1, \ldots, b_k$. Since the increasing subsequence can be extended from the subsequence that ends with bi if $b_k > b_i$, for $i = 1, 2, \ldots, k-1$, the recurrence relation for the optimal solution is:
    a. $L(k) = \max_{1 \le i \le k-1} \{L(i) - 1 | b_i < b_k\}$.
    b. $L(k) = \max_{1 \le i \le k-1} \{L(i) + 1 | b_i > b_k\}$.
    c. $L(k) = \max_{1 \le i \le k-1} \{L(i) + 1 | b_i < b_k\}$.
    d. $L(k) = \max_{1 \le i \le k-1} \{L(i) + 1 | b_i = b_k\}$.
27. **Algorithm 1** is a dynamic programming algorithm for solving the longest increasing subsequence, which returns the length of the longest increasing subsequence of the input string. Please fill the blank in it.

Algorithm 1 LIS(A[1..n])

```
for i =1 to n do
    L[i] = 1
for i = 1 to n do
  for j = 1 to i-1 do
    if A[i] > A[j] then
        L(i) = max{L[i], L[j]+1}
LIS_length = 0
For i = 1 to n do
  If ___(27)___ then
  LIS_length = L[i]
return LIS_length
```

Which of the following should be in Blank (27)?
    a. L[i] > LIS_length
    b. L[i] = LIS_length
    c. L[i] < LIS_length
    d. i < n

28. Which of the following is correct?
    a. It takes $O(N^2)$ for sorting and $O(N \log N)$ for computing the longest increasing subsequence. Therefore, the time complexity of the algorithm is $O(N^2)$.
    b. A. It takes $O(N \log N)$ for sorting and $O(N^2)$ for computing the longest increasing subsequence. Therefore, the time complexity of the algorithm is $O(N \log N)$.
    c. A. It takes $O(N \log N)$ for sorting and $O(N^2)$ for computing the longest increasing subsequence. Therefore, the time complexity of the algorithm is $O(N^2)$.
    d. A. It takes $O(\log N)$ for sorting and $O(N^2)$ for computing the longest increasing subsequence. Therefore, the time complexity of the algorithm is $O(N^2)$.
29. Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n) = T(n/2) + n^2$. Use the substitution method to verify your answer.
    The following is an incomplete proof:

Suppose that $T(i) \le cn^2, for\ all\ i < n.$

$$Then\ T(n) = T\left(\frac{n}{2}\right) + n^2 \le c\left(\frac{n}{2}\right)^2 + n^2 = \left(\frac{c}{4} + 1\right)n^2 \le$$

_ (29) ___ $, as\ long\ as\ c \ge \frac{4}{3}. Hence\ T(n) = O(n^2).$

The blank (29) should be filled with:

a. cn

b. cn^2

c. n/2

d. $T(n/2) + n^2$

30. Given an unweighted graph $G = (V, E)$, we need to find the shortest path from u to v, where $u, v \in V$. Show that the problem of finding the shortest path from u to v exhibits the optimal substructure property. The following is an incomplete proof. Please complete it by filling the blank(s).

Proof: We denote the shortest path of u to v as P, and one vertex in the path P as w. The path between u and w in the P is P_{uw}. And the path between w and v in the P denoted as P_{wv}. We want to prove that the P_{uw} is the shortest path between u and w.

If the P_{uw} is not the shortest path from u and w, then we can find another path P_{uw^*} shorter than P_{uw}. Then _(30)______ and this is contradiction with the definition that P is the shortest path of u and v. So the path from u and w in the P is a shortest path too. Then this problem has the optimal substructure.

Which of the following should be in Blank (30)?

a. $P_{uw} + P_{wv} < P_{uw^*} + P_{wv}$

b. $P_{uw^*} < P_{uw}$,

c. $P_{uw^*} < P_{uw} + P_{wv}$,

d. $P_{uw^*} + P_{wv} < P_{uw} + P_{wv}$,