1. True or False (20%)

(1) All problems have algorithmic solutions, but some of the solutions are impractical.
(2) A repeat-until loop can execute 0 times.
(3) The sequential search algorithm uses only sequential operations.
(4) Pseudo code is a precise, rigid language to insure uniformity among programmers.
(5) The key to measuring the efficiency of an algorithm is to find out how long it takes to run on a specific machine.
(6) The binary search algorithm is in the worst case O(n²)
(7) The binary search algorithm is a good algorithm for searching an unordered list.
(8) Correctness is the most important consideration when designing algorithms.
(9) \[ \neg[(x>y) \lor \neg(x<z)] \] (assume that x=1, y=2, z=3)
(10) \[(y=z) \land [(x<y) \lor (y<z)]\] (assume that x=1, y=3, z=3)

2. (a) State the formal definition of an algorithm. (5%)

A well-ordered collection of unambiguous and effectively computable operations that, when executed, produces a result and halts in a finite amount of time.

(b) List the necessary and desirable properties of an algorithm. (3%)

Necessary: Correctness,
Desirable: Efficiency, ease of understanding, elegance

(c) Is the following an algorithm? Explain your reasoning. (2%)

No. Step 1 might not be effectively computable. If it is, then the algorithm never comes to an end.

Step 1: Go to your bank and withdraw 1000 NT dollars
Step 2: Give the money to your Computer Science Professor
Step 3: Goto Step 1

3. Name the three classes of algorithmic operations (6%) and classify the following statements accordingly (4%): Sequential, conditional, and iterative operations.

(a) Print x if n=0 ; Print n if n != 0 conditional
(b) Replace i with i-1 sequential
(c) While the water is heating, stir the sauce once every two minutes iterative
(d) Repeat the following 5 operations until i<100 iterative

4. (a) Prove the two-variable DeMorgan’s theorem using the truth table: \((A+B)' = A' B'\) (3%)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A+B</th>
<th>(A+B)'</th>
<th>A'B'</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
(b) Use mathematical induction to show that \((A_1+A_2+\ldots+A_n)'=A_1' A_2' \ldots A_n'\). (3%)

Given that \((A_1+A_2+\ldots+A_k)'=A_1' A_2' \ldots A_k'\)

Now \((A_1+A_2+\ldots+A_k+A_{k+1})'= (A_1+A_2+\ldots+A_k)' A_{k+1}'\) (from part (a))

\[ = A_1' A_2' \ldots A_k' A_{k+1}' \]

(c) Show that \(x+x'y = x+y\). Construct and compare the circuit diagrams for \(x+x'y\) and \(x+y\) using AND, OR, NOT gates, then discuss why such simplification is desirable. (4%)

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>x+x’y</th>
<th>x+y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\(x+x’y\) uses three gates (one NOT gate, one AND gate and one OR gate), while \(x+y\) uses only one OR gate.

5. (a) The writer of the following algorithm wishes to print out whether a student (who is in 1st through 12th grade) is in the elementary (1-6th), middle (7-9th), or high school (10-12th). Does it wok correctly? If not, why? (4%) No. Step 3 should be: if (grade>6 ) AND (grade <=9) then

Step 1: If grade<=6 then
Step 2: Print “The student is in elementary school”
Step 3: If grade<=9 then
Step 4: “The student is middle school”
Step 5: Else Print “The student is in high school”

(b) Design an algorithm to compute the income tax of a taxpayer, assuming that the tax rate is:

- 10% if total income < 500000 NT, 20% if 500000 <= income < 1000000 NT.
- 30% if income >=1000000 NT (6%) 累進稅制

Step 1: Get value for the total_income
Step 2: If total_income < 500000 then
Step 3: tax = total_income*0.1
Step 4: else if (total_income >=1000000) AND (total_income <500000) then
Step 5: tax = 500000*0.1 + (total_income-500000)*0.2
Step 6: else tax = 500000*0.1 + 500000*0.2 + (total_income-1000000)*0.3
Step 7: Print the value of tax.
Step 8: Stop

6. The bubble sort algorithm is listed below. It makes multiple passes through the list from front to back, each time exchanging pairs of entries that are out of order.

Step 1: Give values for n and the n list items
Step 2: Set the marker U for the unsorted section at the end of the list
Step 3: Repeat steps 4 through 8 until the unsorted section has just one element
Step 4: Set the current marker \( C \) at the second element of the list.

Step 5: Repeat steps 6 and 7 until \( C \) is to the right of \( U \).

Step 6: If the item at position \( C \) is less than the item to the left then exchange these two items.

Step 7: Move \( C \) to the right one position.

Step 8: Move the marker for the unsorted section \( U \) forward one position.

Step 9: Stop

(a) Perform a bubble sort on the following two lists: 4,8,2,6 and 13,3,6,8,2. Show the list after each exchange. (4%) 4826,4286,4268,2468 and 13-3-6-8-2, 3-13-6-8-2, 3-6-13-8-2, 3-6-8-13-2,3-6-8-2,13,3-6-8-13,3-2-6-8-13

(b) Show that the computational complexity for the bubble sort algorithm is \( O(n^2) \). (4%) Value of \( U \) Number of passes through loop at step 5

\[
\begin{array}{|c|c|}
\hline
n & n-1 \text{ (C goes from 2 to } n+1) \\
\hline
n-1 & n-2 \text{ (C goes from 2 to } n) \\
\hline
2 & 1 \text{ (C goes from 2 to } 3) \\
\hline
\end{array}
\]

The loop at step 5 is therefore done \((n-1)+(n-2)+...+1\) times = \( O(n^2) \)

(c) Compute the number of exchanges that will be performed by the bubble sort algorithm assuming that the input list is already sorted. (2%) 0

7. (a) What decimal is 10110011 in ordinary binary notation? (2%) In sign-magnitude notation (2%) and in 2's complement notation (2%)? Ordinary binary: \( 2^7+2^5+2^4+2^1+2^0 = 179 \), Sign-magnitude: \( -(2^5+2^4+2^1+2^0) = -51 \), 2’s complement: \( -2^7+2^5+2^4+2^1+2^0 = -77 \)

(b) What does the letter \( G \) represent when used to describe the speed of a CPU (say, the clock rate is 1.5 GHz)? (2%) when used indicate the amount of storage (say 40Gbytes)? (2%) speed: \( 10^9 \), storage: \( 2^{30} \)

8. (a) Describe how a two-input multiplexor works. (2%) Select either of the two inputs based on the value of the SELECT line.

(b) Suppose that the two inputs are \( A \) and \( B \) and the selection line is \( S \). Construct the truth table for the two-input multiplexor. (4%)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>S</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
(c) Use the result in (b) to derive the Boolean expression of the two-input multiplexor using the circuit construction algorithm discussed in class. (4%) \( O = A'B+S + A'B'S'+AB'S'+AB 

Optional = BS(A+A') + AS'(B'+B) = AS'+BS 

9. (a) Give the truth table and the circuit diagram (using AND, OR, NOT gates) that corresponds to the Boolean expression: \( C = A'B + AB' \). (5%) 

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A'B + AB'</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

(b) Match each of the following algorithms with its worst-case time efficiency. (5%) 
- Searching (sequential) \( O(n) \) 
- Searching (binary) \( O(n^2) \) 
- Sorting (selection sort) \( O(1) \) 
- Data cleanup: converging pointers \( O(mn) \) 
- Data cleanup: shuffle left \( O(log\,n) \)