1. (20%) Multiple-choice questions (單選題).

1.1. (5%) Consider the following C++ code.

```
#include <iostream>
using namespace std;

class Copied {
public:
  int* num;
  int len;

  Copied (int length) {
    num = new int[100];
    for (int i = 0; i < 100; i++) num[i] = i;
    len = 100;
  }
  Copied (const Copied& c) : // Line A
    num(c.num), len(c.len) { }

  virtual Copied operator=(const Copied& c) {return Copied(c);}
};

int main( ) {
  Copied n1(5);
  Copied n2(n1);
  n2.num[0] = 50;
  cout << "n2.num[0] = " << n2.num[0];
  cout << ", n1.num[0] = " << n1.num[0];
  cout << endl;
}
```

What is printed by the `cout` statements in main?

(a) n2.num[0] =50, n1.num[0] = 50
(b) n2.num[0] = 50, n1.num[0] = 0
(c) n2.num[0] = 0, n1.num[0] = 50
(d) n2.num[0] = 0, n1.num[0] = 0
(e) There is an error because << is not overloaded by the copied class.

1.2. (5%) Continue the problem 1.3. Replace the constructor `Copied (const Copied& c)` at (Line A) above with the code below. With this new constructor, what is printed by the `cout` statement?

```
Copied (const Copied& c) : len(c.len) {
  num = new int[len];
  for (int i = 0; i < len; i++) { num[i] = c.num[i]; }
}
```

(a) n2.num[0] = 50, n1.num[0] = 50
(b) n2.num[0] = 50, n1.num[0] = 0
(c) n2.num[0] = 0, n1.num[0] = 50
(d) n2.num[0] = 0, n1.num[0] = 0
(e) n2.num[0] = 50, n1.num[0] = 50
(f) There is an error because << is not overloaded by the copied class.

1.3. (5%) A function prototype does not have to _____.
(a) include argument names
(b) end with a semi-colon
(c) agree with the function header
(d) match with all calls of the function

1.4. (5%) In C++, to exhibit polymorphism it is necessary to _____.
(a) declare all methods that will exhibit polymorphic behavior as virtual
(b) access the methods by a pointer or reference
(c) declare the class as abstract
(d) declare the class as polymorphic
(e) (a) and (b)
(f) (a) and (c)
(g) (a) and (e)

2. (10%) The GNU C compiler supports an extension to ANSI C called "statement expressions," which allows any compound statement to be treated an expression by enclosing it in parentheses. For example, we can write

```
#define maxint (a, b) ({int a1 = (a), b1 = (b); a1>b1 ? a1 : b1;})
```

Compare this to the more usual C definition

```
#define max (a, b) ((a) > (b) ? (a) : (b))
```

Consider passing expressions `max(i++, j++)` and `maxint(i++, j++)`, where these two macros behave differently, which one is probably a better definition, and explain why.

3. (6%) The following C function converts a string of lowercase characters into uppercase characters. For example, it will convert "abc" into "ABC". The parameter `str` is the string. Fill in the missing expressions in (1) and (2).

```
void lower2upper(char str[]) {
   int i = 0;
   while (      (1)      ) {
      str[i]= str[i] +      (2)      ;
      i++;
   }
}
```

4. (8%) The following C function transposes a 3 × 3 array of integers. For example, it will trnaspose {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}} into {{1, 4, 7}, {2, 5, 8}, {3, 6, 9}}. The parameter `m` is the array. Fill in the missing expressions in (1) ~ (8).

```
void transpose(int (*m)[3]) {
   int i, j, t;
   for (i =    (1)    ; i <    (2)    ; i++) {
      for (j =    (3)    ; j <    (4)    ; j++) {
         t = m[i][j];
         m[ (5) ][ (6) ] = m[ (7) ][ (8) ];
         m[j][i] = t;
      }
   }
}
```

5. (6%) Complete the following C function that sums the elements of a one-dimensional array of integers. The parameter `m` is the array and the parameter `n` is the size of the array. You need to use the pointer notation to access the array.

```
int sum(int *m, int n) { ... }
```

6. (5%) Suppose that the postorder traversal of a binary tree T is

   A D B C F E

   and the inorder traversal of T is

   A D B E F C

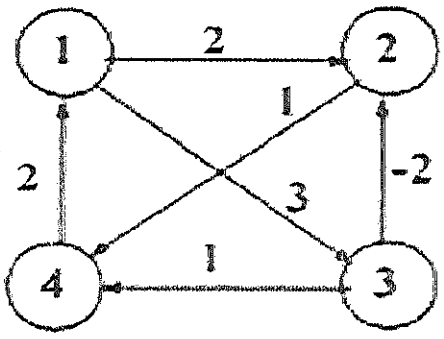
   What is the preorder traversal of T?

7. (7%) Quicksort:
   (a) (5%) Please write a C function to do quicksort on an array of integers.
   (b) (1%) What is the worst case time complexity of quicksort algorithm?
   (c) (1%) Please describe a method to avoid the worst case.

8. (8%) Please write a C function to search on a Binary Search Tree. Each node of the tree has a field named 'key' which is a string with at most 32 bytes. The return value is a pointer to the matched node if there is a match. The return value is NULL if there is no match.

9. (7%) Bellman-Ford and Dijkstra algorithms have been well-known for finding the Single-Source Shortest paths in a graph. Briefly answer the following questions.
   (a) (2%) Which algorithm(s) can be applied on graphs with negative edge?
   (b) (2%) Which algorithm(s) is able to detect negative cycle?
   (c) (3%) Given a directed acyclic graph, is there any way to run faster than the above two algorithms?

10. (8%) All-Pairs Shortest Paths



   (a) (4%) Compute the lengths of all-pairs shortest paths in this graph using the Floyd-Warshall algorithm.
   (b) (4%) Briefly describe the transformation process in the Johnson's algorithm which converts the negative edge weights into positive weights without affecting the shortest path solution between any two nodes.

11. (5%) Consider the following procedure for sorting an array of numbers. Give a loop invariant which can be used to prove its correctness.

```
procedure Sort(A)
1.  for j=2 to length(A)
2.      do key=A[j]
3.         i=j-1
4.         while i>0 and A[i]>key
5.            do A[i+1]=A[i]
6.               i--
7.         A[i+1]=key
8.  return
```

12.(10%) Consider a chain of n matrix $A_1, A_2, \cdots, A_n$, where the dimension of A_i is $d_{i-1} \times d_i$. Assume the number of multiplications to compute $A_i \times A_{i+1}$ is $d_{i-1} \cdot d_i \cdot d_{i+1}$. Let $m[i, j]$ denote the minimum number of multiplications to compute the product of the subchain $A_i \times A_{i+1} \times \cdots A_j$.

(a) (5%) Give the recursive formula for $m[i, j]$.

(b) (5%) Show, step by step, how to use your recursive formula to compute $m[1,6]$ from the following tables for computing $m[i, j]$.

matrix	A_1	A_2	A_3	A_4	A_5	A_6
dimension	30×35	35×15	15×5	5×10	10×20	20×25

$m[i, j]$	$j = 1$	2	3	4	5	6
$i = 1$	0	15,750	7,875	9,375	11,875	?
2	—	0	2,625	4,375	7,125	10,500
3	—	—	0	750	2,500	5,375
4	—	—	—	0	1,000	3,500
5	—	—	—	—	0	5,000
6	—	—	—	—	—	0