Data Structures and Algorithms
Final Exam

1. Please fill in the blanks with your choices:

(1) Suppose that n integers are pushed into and popped out of a stack. The input sequence is 1, 2, 3, …, n and the output sequence is $p_1, p_2, p_3, \dots, p_n$. If $p_1 = n$, then $p_i =$ ___________. (2 points)

① i ; ② $n - i$; ③ $n - i + 1$; ④ Cannot be determined

(2) A circular queue is represented by an array with m elements. Given the values of *front* and *rear*, the number of elements in the queue is _______________. (2 points)

① (*rear* – *front* + *m*) MOD *m* ; ② *rear* – *front* + 1 ; ③ *rear* – *front* – 1 ; ④ *rear* – *front*

(3) Given a connected graph with n vertices, the minimum number of edges in the graph must be _________. (2 points)

① n ; ② $n - 1$; ③ $n + 1$; ④ $n(n - 1)/2$

(4) Given the preorder traversal sequence ABDCEFHG and the inorder traversal sequence BDAFHEGC, the corresponding postorder traversal sequence is ________________. (3 points)

① DBAHFGCE; ② BDHFGECA; ③ DBHFGECA; ④ DBCFHEGA

(5) Given a list of integers {1, 2, 3, 4, 5, 9, 8, 6, 7}. To sort them in increasing order, _________ is the fastest, and _________ takes the most extra spaces. (4 points)

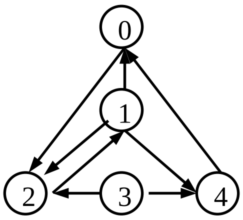
① insertion sort; ② quick sort; ③ merge sort; ④ heap sort; ⑤ selection sort

(6) Given a segment of text “BB’S_NOT_ON_BBS”, __________bits are required to store the string as 1-byte characters, and __________bits are need to store its Huffman code. (5 points)

① 120; ② 112; ③ 88; ④ 40; ⑤ 21

2. For the digraph given by the figure, obtain:

(1) the in-degree and out-degree of each vertex (5 points)
(2) its adjacency matrix (3 points)
(3) its adjacency list representation (3 points)
(4) its inverse adjacency list representation (3 points)
(5) its strongly connected components (5 points)



3. Given a list of integers {40, 30, 20, 50, 60, 45, 25, 55, 35, 38}. Let us insert these integers according to the given order into an AVL tree which is originally empty. Please draw the resulting AVL tree.（12 points）

4. Please read the following two programs. Specify their functions and fill in the blank lines of code.
（14 points）

```
（1）void test(element item, list_pointer ht[])
        {
            int hash_value = hash( item.key) ;
            list_pointer ptr, trail = NULL, lead = ht[hash_value];
            for ( ; lead; trail = lead, lead = lead->next) {
                    if (!strcmp(lead->item.key, item.key)) {
                            fprintf(stderr, "The key is in the table\n");
                            exit(1);
                    }
             }
            ptr = (list_pointer)malloc(sizeof(list));
            if (IS_FULL(ptr)) {
                    fprintf(stderr, "The memory is full\n);
                    exit(1);
            }
            ptr->item = item;
            ptr->next = ___________;
            if (trail)
                ____________________;
            else
                _______________________;
        }
```

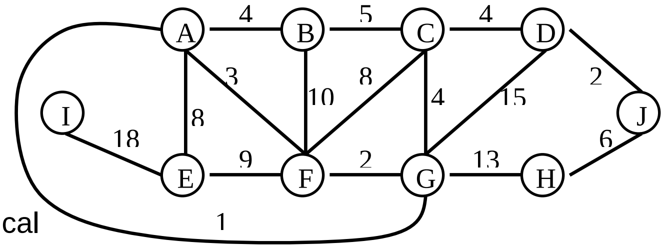
The function is:

```
（2）void test ( int cost [ ] [ MAX_VERTICES ], int distance [ ] [ MAX_VERTICES ], int n )
        {     int    i, j, k ;
            for ( i = 0; i < n; i++ )
                for ( j = 0; j < n; j++ )
                    distance [ i ] [ j ] =  _________________  ;
             for ( k = 0; k < n; k++ )
                for ( i = 0; i < n; i++ )
                    for ( j = 0; j < n; j++ )
                        if ( distance [ i ] [ k ] + distance [ k ] [ j ] < distance [ i ] [ j ] )
                            ____________________________________________;
        }
```

The function is:

5. For the given graph, please draw
   (1) the depth-first spanning tree; (5 points)
   (2) the breath-first spanning tree; (5 points) and
   (3) the minimum cost spanning tree. (9 points)

Note: In (1) and (2), the searches both start from A.
The vertices will be visited according to alphabetical order if there are several options.



6. Please write a C program that deletes an arbitrary element with index *i* from a min heap. The resulting heap must satisfy the min heap definition. The time complexity of your function must be O(log$_2$ *n*) where *n* is the number of elements in the heap. (18 points)

void delete_arbitrary_min_heap(element heap[ ], int i, int *n)