招生學年度	101	招生類別	轉學招生考試
系所班別	資訊工程學系三年級		
科目	資料結構		
注意事項	禁止使用掌上型計算機		

1. (10%) Please complete the following code snippets by writing a statement or partial statement for ❶.

```
// Return position of largest value in "A" of size "n"
int largest(int A[], int n) {
    int currlarge = 0; // Holds largest element position
    for (int i=1; i<n; i++) // For each array element
        if (A[currlarge] < A[i]) // if A[i] is larger
            currlarge = ____❶____; // remember its position
    return currlarge; // Return largest position
}
```

2. (15%) Assume $T(n) = c_1n^2 + c_2n$ for c_1 and $c_2 > 0$. Then prove that the lower bound of $T(n)$ is $\Omega(n^2)$.
3. (15%) Please complete the Implementations in the following code snippets for the doubly linked list remove method.

```
// Doubly linked list link node
template <typename E> class Link {
    ...
public:
    E element; // Value for this node
    Link* next; // Pointer to next node in list
    Link* prev; // Pointer to previous node
};

// Remove and return current element
E remove() {
    if (curr->next == tail) // Nothing to remove
        return NULL;
    E   it = curr->next->element; // Remember value
    Link<E>* ltemp = curr->next; // Remember link node
    curr->next->____❶____ = curr;
    curr->next = ____❷____; // Remove from list
    delete ____❸____; // Reclaim space
    cnt--; // Decrement cnt
    return it;
}
```

招生學年度	101	招生類別	轉學招生考試
系所班別	資訊工程學系三年級		
科目	資料結構		
注意事項	禁止使用掌上型計算機		

4. (15%)Please complete the following functions, enqueue() and dequeue(), for implementation of Arryay-based queue.

```
// Array-based queue implementation
template <typename E> class AQueue: public Queue<E> {
private:
    int maxSize; // Maximum size of queue
    int front; // Index of front element
    int rear; // Index of rear element
    E *listArray; // Array holding queue elements
public:
    AQueue(int size =defaultSize) { // Constructor
    // Make list array one position larger for empty slot
        maxSize = size+1;
        rear = 0; front = 1;
        listArray = new E[maxSize];
    }
    ~AQueue() { delete [] listArray; } // Destructor
    void clear() { rear = 0; front = 1; } // Reinitialize
    void enqueue(const E& it) { // Put "it" in queue
        Assert(( ❶ ) != front, "Queue is full");
        rear = ❷ ; // Circular increment
        listArray[rear] = it;
    }
    E dequeue() { // Take element out
        Assert(length() != 0, "Queue is empty");
        E it = listArray[front];
        front = ❸ ; // Circular increment
        return it;
    }
    const E& frontValue() const { // Get front value
        Assert(length() != 0, "Queue is empty");
        return listArray[front];
    }
    virtual int length() const // Return length
    { return ((rear+maxSize) - front + 1) % maxSize; }
};
```

招生學年度	101	招生類別	轉學招生考試
系所班別	資訊工程學系三年級		
科目	資料結構		
注意事項	禁止使用掌上型計算機		

5. (10%) Please write down the preorder enumeration for the tree of Figure

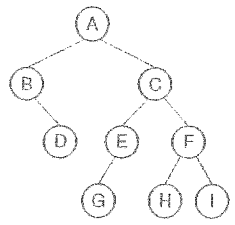

Figure 1 A binary tree.

6. (15%)What is a heap and max-heap? Why heaps are nearly always implemented using the array representation?
7. (20%) The following C++ ???() function is for graph traversal. What kind of graph traversal algorithm ???() function is?

```
    void moveToStart(); // Set the current position to the start of the list
    void prev(); // Move the current position one step left.
    void next();     // Move the current position one step right.
    int length();    // Return: The number of elements in the list
    const E& getValue()       // Return: The current element.

    int first(int v); // Return first neighbor of "v"
    int last(int v); // Return last neighbor of "v"
    int next(int v, int w); // Get v's next neighbor after w
    int getMark(int v) { return mark[v]; }
    void setMark(int v, int val) { mark[v] = val; }
};

void ???(Graph* G, int v) { // ??? search
    // PreVisit(G, v); // Take appropriate action
    G->setMark(v, VISITED);
    for (int w=G->first(v); w<G->last(v); w = G->next(v,w))
        if (G->getMark(w) == UNVISITED)  ???(G, w);
    // PostVisit(G, v); // Take appropriate action
}
```

#