國立成功大學
111學年度碩士班招生考試試題

編　號：178

系　所：電機工程學系

科　目：資料結構

日　期：0219

節　次：第 2 節

備　註：不可使用計算機

國立成功大學111學年度碩士班招生考試試題

系　所：電機工程學系

考試科目：資料結構　　　　考試日期：0219，節次：2

※ 考生請注意：本試題不可使用計算機。 請於答案卷(卡)作答，於本試題紙上作答者，不予計分。

(甲) 80% 單選題, 共 20 題 (每題 4 分)

一定要按照答題**順序** 並填上答案 (可以留空) 並如題目選項以**英文大寫**作答, 其他表示法不予計分)

1. _____ form of access is used to add and remove nodes from a stack
A. LIFO　B. FIFO　C. Both (A) and (B)
D. None of these

2. A linked list index is ____representing the position of a node in a linked list.
A. an Integer　B. a variable　C. a character　D. a Boolean

3. Which of the following statement is false?
A. Arrays are dense lists and static data structure
B. data elements in a linked list need not be stored in adjacent space in memory
C. pointers store the next data element of a list
D. linked lists are a collection of the nodes that contain the information part and next pointer

4. A _____ is a data structure that organizes data similar to a line in the supermarket, where the first one in line is the first one out.
A. queue linked list　B. stacks linked list C. both of them D. neither of them

5. Which of the following data structure is linear type?
A. Strings　B. Lists　C. Queues　D. All of above

6. Which of the following data structure is a non-linear data structure?
A. Arrays　B. Linked lists　C. Trees　D. None of above

7. Identify the data structure which allows deletions at both ends of the list but insertion at only one end
A. Input-restricted dequeuer　B. Output-restricted dequeue
C. Priority queues　D. None of above

8. The situation when in a linked list START=NULL is
A. underflow　B. overflow　C. saturated　D. None of above

9. Which of the following data structure can't store the heterogeneous data elements?
A. Arrays　B. Records　C. Pointers　D. None

10. The five items: A, B, C, D, and E are pushed in a stack, one after the other, starting from A. The stack is popped four times, and each element is inserted in a queue. Then two elements are removed from the queue and pushed back on the stack. Now one item is popped from the stack. The popped item is.
A. A　B. B　C. C　D. D

11. In a graph, if e=[u, v], Then u and v are called
A. endpoints of e　B. adjacent nodes　C. neighbors　D. all of above

12. A variable P is called a pointer if
A. P contains the address of an element in DATA
B. P points to the address of the first element in DATA
C. P can store only memory addresses
D. P contain the DATA and the address of DATA

13. The post-order traversal of a binary tree is CDBFGEA. Find out the pre-order traversal.
A. CBDAFEG　B. ADBFECG　C. CBDGAEF　D. CDBFGEA

14. Elements of an array are stored ______ in memory
A. periodical　B. sequentially　C. in Parallel　D. None of the above

15. Which of the following case does not exist in complexity theory
A. Best case　B. Worst case　C. Average case　D. Null case

16. The operation of processing each element in the list is known as
A. Sorting　B. Merging　C. Inserting　D. Traversal

17. To represent the hierarchical relationship between elements, which data structure is suitable?
A. Dequeue　B. Priority　C. Tree　D.All of above

18. An algorithm that calls itself directly or indirectly is known as
A. Sub algorithm　B. Recursion　C. Polish notation　D. Traversal algorithm

19. Linked lists are best suited
A. for relatively permanent collections of data
B. for the size of the structure and the data in the structure are constantly changing
C. for both of above situation
D. for none of above situation

20. The reason for using a pointer is ...　(Choose the false option from the following sentences)
A. Accessing arrays or string elements
B. Dynamic memory allocation
C. Implementing linked lists, trees, graphs, and many other data structures
D. All are false

(乙) 20% 短答題, 共 4 題 (每題 5 分)

以下共四個 function, 請簡單回答他們的功能

1. **What is the usage of the following function?**
   (You can use Input: nums = [2,7,8,13], target = 10 to explain)

```
vector<int> twoSum(vector<int>& nums, int target) {
    for(int i=0;i<nums.size();i++)      {
        for(int j=i+1;j<nums.size();j++)        {
            if(nums[i]+nums[j]==target)   {
                return {i,j};
            }
        }
    }
    return {};
}
```

2. **What is the usage of the following function?**

```
string ans="";      int count=0;
int ans_number=0;      int first=500;
string Function X (vector<string>& strs) {
    if(strs[0]=="")                     return "";
    if(strs.size()==1) {
        return strs[0];
    for(int i=1;i<strs.size();i++){
            if(strs[0].size()>=strs[i].size()){
                count=strs[i].size();
            }
            else count=strs[0].size();
        for(int j=0 ;j<count;j++){
            if(strs[0][j]==strs[i][j]){
                ans_number++;
            }
            else break;
        }
        if(ans_number<first){
            first=ans_number;
        }
            ans_number=0;
    }
    strs[0].resize(first);
    return strs[0];
}
```

3. What is the usage of the following function?

```
bool isValid(string s) {
    stack<char> save;
    char current;
    if (s.size()%2==1) return false;
    if(s[s.size()-1]=='('||s[s.size()-1]=='{'||s[s.size()-1]=='[')   return false;
    for(int i=0;i<s.size();i++) {
            current=s[i];
            switch (current) {
            case '(':
                    save.push(current);   break;
            case '{':
                    save.push(current);   break;
            case '[':
                    save.push(current);   break;
            case ')':
                    if(save.size()==0||save.top()!='(')   return false;
                    save.pop(); break;
            case '}':
                    if(save.size()==0||save.top()!='{')   return false;
                    save.pop(); break;
            case ']':
                    if(save.size()==0||save.top()!='[')   return false;
                    save.pop(); break;
            default :
                    return false; break;
        } // end switch
    } // end for
    if(save.size()!=0) {
        if(save.top()=='('||save.top()=='{'||save.top()=='[')   return false;
    }
    return true;
}
```

4. What is the usage of the following function?

```
// struct ListNode {
//     int val;
//     ListNode *next;
//     ListNode() : val(0), next(nullptr) {}
//     ListNode(int x) : val(x), next(nullptr) {}
//     ListNode(int x, ListNode *next) : val(x), next(next) {}
//};
   ListNode* MM(ListNode* L1, ListNode* L2) {
       if(L1 == NULL)            return L2;
       else if(L2 == NULL)       return L1;

       if(L1->val <= L2->val){
           L1->next = MM(L1->next, L2);            return L1;
       }
       if(L1->val > L2->val){
           L2->next = MM(L1, L2->next);            return L2;
       }
       return 0;
```